Java Script do Curso em Video

No VsCode > escreva > html:5 > enter > Faz o html normal

No final do body adiciona <script></scricpt>

Aula 4: ex001:

<script>

        window.alert('Minha primeira mensagem.')

</script>



O conteúdo some

<script>

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

</script>



<script>

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

</script>

****

<script>

        window.alert('Minha primeira mensagem.')

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

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

</script>

Variáveis: serve para guardar dados

Comentários: // - serve para uma única linha e /\*\*/ - mais de uma linha

##### Identificadores das variáveis:

* Podem começar com letra, $ ou \_
* Não podem começar com números
* É possível usar letras ou números
* É possível usar acentos e símbolos
* Não podem conter espaços
* Não podem ser palavras reservadas (comandos do JS)

No Node.js ou no VsCode > Terminal > New terminal



Para limpar a tela: ctrl + L

Para sair do programa: .exit

Para mostrar qual o tipo da variável no terminal ou Node.js se coloca typeof

Diferença entre VAR  X  LET  X CONST Aula html > udemy > Var.Let.Const.js

//Usando Var - pouco usado

var value; //pode apenas declarar sem descrição de primeira

value = '1232';

//Usando Let

let color = 'Red'; //pode apenas declarar sem descrição de primeira

color = 'Blue'

//Usando Const - NÂO pode alterar, redefinir, da erro - mais usado

const fullName = 'Larytgs' //sempre tem q ter uma descrição



console.log(value);

console.log(color);

console.log(fullName);

#### Tipos Primitivos Primordiais

|  |  |
| --- | --- |
| Number | 5 18 -12 0.5 -15.9 3.14 8.0  Infinity NaN(not a number) |
| String | “Google” ‘JavaScript’ `Maria` “098-97.09” |
| Boolean | True e False |
| Null | Valor vazio |
| Undefined | Quando não tá recebendo nenhum valor |
| Object | [ ] { } Array(vetor) |
| Function | Funções |

## Uso de caracteres especiais em string

|  |  |
| --- | --- |
| \0 | Byte nulo |
| \b | Backspace |
| \f | Alimentador de formulário |
| \n | Nova linha |
| \” | Aspas duplas |
| \’ | Apóstrofo ou aspas simples |

Array – serve para fazer uma lista Aula html > udemy > array.js

*const* times = ['Flamengo', 'Coxa', 'Botafogo', 'Atlético', 'Cruzeiro'];

console.log('Acessando os times: ', times) //mostrar em lista

console.log('O segundo time é: ', times [1]) //mostrar o segundo time

console.log('Sao',times.length, 'arrays.') // tem 5

console.table(times) //mostrar em tabela

//Operaçoes com o array

times.unshift('Corinthians') //add no começo do array

times.push('USA') //add no final do array

console.log('\nAdicionando mais dois times no inicio e no final:')

console.table(times)

console.log('Agora o tamanho da array tem:' ,times.length , 'times.') // tem 7

times.shift() //remove o primeiro elemento

times.pop() //remove o ultimo elemento

console.log('\nRemovendo o primeiro e o ultimo:')

console.table(times)

console.log('\nAdicionando um novo time na posição 2:')

times.splice(2, 1, 'Palmeiras') //add um elemento em uma posição específica, e removendo o antigo. 1º coloca o indice q quer remover, e quantos elementos quer remover(ate 2), e add ate 2

console.table(times)

console.log('\nOutro exemplo, adicionando 2 times na posição 1')

times.splice(1, 1, 'USA', 'Botafogo') //outro exemplo, removendo 1, add 2

console.log(times)

[Video explicando sobre o splice( )](https://www.google.com/search?q=usando+o+splice%28%29+no+javascript&sca_esv=2ee0018a8b8908ea&sca_upv=1&sxsrf=ACQVn08MNfXSHjLs3_sCgFh6fJxHR7xg4Q%3A1712942062144&ei=7msZZvi1CKyChuMP7MykKA&udm=&oq=usando+o+splice%28%29&gs_lp=-LAQ&sclient=gws-wiz-serp#fpstate=ive&vld=cid:83e966bc,vid:SyuCjQCn05U,st:0)

## Conversão de string para numero: (parse-conversao) Aula06-Ex003

* Number.parseInt(n)
* Number.parseFloat(n)
* Number(n)

<script>

        var n1 = Number.parseFloat(window.prompt("Digite um numero:"))

        var n2 = Number(window.prompt("Digite outro numero: "))

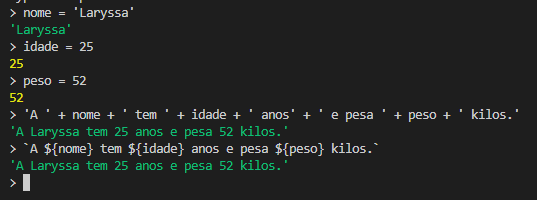
        var s = n1 + n2

        window.alert('A soma dos dois numero é ' + s)

</script>

#### Conversão de numero para string:

* Sting(n)
* n.toSting( )

Template de String – Formatação de strings – utilizando `crases` e ${n} – no node

## Aula06-Ex004:

* s.length : quantos caracteres a string tem
* s.toUpperCase() : tudo em letra ‘MAIUSCULAS’
* s.toLowerCas() : tudo em letra ‘minusculas’

<script>

        var nome = window.prompt("Qual é oseu nome? ")

        document.write(`Ola ${nome}. Seu nome ${nome.length} tem letras. <br>`)

        document.write(`Seu nome em maiusculas fica ${nome.toUpperCase()} <br>`)

        document.write(`Seu nome em minusculas fica ${nome. toLowerCase()}`)

</script>

#### Formatação de números – no node

* n1.toFixed(2) : para colocar duas casas depois do ponto
* n1.toFixed(2).replace(‘.’ , ‘,’) : para substituir oponto por virgula
* n1.toLocalesString(‘pt=BR’, {style: ‘currency’,

currency: ‘BRL’}) : colocando R$

* n1.toLocalesString(‘pt=BR’, {style: ‘currency’,

currency: ‘USD’}) : colocando US$

* n1.toLocalesString(‘pt=BR’, {style: ‘currency’,

currency: ‘EUR’}) : colocando €

## **Operadores de Java Script**

|  |  |  |
| --- | --- | --- |
| * Aritméticos     Ordem de precedência:  ( )  \*\*  \* / %  + -  .  5 % 2 = é o resto da divisão  .  5 \*\* 2 = é a potencia | * Atribuição     Atribuiçao simples <- | |
| * Relacionais     Identidade:  .  5 é igual a 5  5 é iguala ‘5’  5 não é idêntico a ‘5’  5 é idêntico a 5 | * Lógicos     && significa “e”  || significa “ou”   * Ordem de precedência:   1º Aritméticos  2º Relacionais  3º Lógicos | |
| * Ternário     O teste logico que pode dar verdadeiro ou falso    Vai aparecer aprovado ou Reprovado dependendo do teste  Ex1:  A média é 5.5  A média é maior que 7?  Se sim, Aprovado, se não Reprovado  Ex2:  X é 8  8 % 2 == 0  Resto de 8 dividido por 2 é 0 então 0 é igual a 0  Se a resposta for True = 5 se for False = 9  Res = 5  Ex3: | | |

## **DOM document Object Model**

Aula 9 ex 05:

Arvore do DOM

3 dos objetos:

Location : URL

Document: o doc atual

History: de onde vim p onde vou

**Ver mais sobre na Aula 9 – Introdução ao DOM**

* Por marca

|  |
| --- |
| <script>  *var* p1 = window.document.getElementsByTagName('p')[1] //seleciona o 1º paragrafo          window.document.write("<hr>Esta escrito assim: &nbsp" + p1.innerText)  //innerHTML: mostra com as formataçoes  </script> |

* Por ID : colocando: id=”msg” la na <div>

|  |
| --- |
| <script>  *var* d = window.document.getElementById('msg')          window.document.write('<hr>' + d.innerHTML)          d.style.backgroundColor = 'green'          d.style.color = 'white'          d.style.fontSize = '2em'          d.innerText = 'Mudando o texto da div com ID'  </script> |

* Por nome colocando: name=”msg” la na <div>

|  |
| --- |
| <script>  *var* d = window.document.getElementsByName('msg')[0]       d.style.textAlign = 'left'       d.style.color = 'green'       d.innerText = 'Agora com o Name, deixando na esquerda'  </script> |

* Por classe colocando: class=”msg” la na <div>

|  |
| --- |
| <script>  *var* d = window.document.getElementsByClassName('msg')[0]          d.style.background = 'orange'          d.style.color = 'blue'          d.style.fontSize = '2em'          d.innerText = 'Mudando agora com o Class'  </script> |

* Por seletor colocando por id=”msg” la na <div>

|  |
| --- |
| <script>  *var* d = window.document.querySelector('div#msg')          d.style.color = 'blue'          d.innerText = 'Mudando agora por QuerySelector'  </script>  Por class=”msg”  Apenas mudar no var no <div>:  ('div.msg')  querySelector : singular  querySelectorAll : plural |

## **Eventos DOM – o que pode acontecer com as tags Aula 10 ex 06 e udemy > exerciciosSimples**

* Mouseenter – entrar com o mouse em cima da tag
* Mouseout – tirar o mouse de cima
* Mousemove – mexer o mouse dentro da tag
* Mousedown – pressionar o mouse
* Mouseup – soltar o mouse
* Click – clicar em cima

|  |
| --- |
| <div id="area" onmouseenter="entrar()" onmouseout="sair()">          Interaja...  </div>  <script>  *var* a = window.document.getElementById('area') //colocando p fora, serve para todos          a.addEventListener('click', clicar)          a.addEventListener('mouseenter', entrar)          a.addEventListener('mouseout', sair)    *function* clicar() {              a.innerText = 'Voce clicou!!'              a.style.background = 'green'              a.style.color = 'yellow'          }  *function* entrar() {              a.innerText = 'entrou'          }  *function* sair() {              a.innerText = 'saiu'              a.style.background = 'purple'          }  </script> |

## Método **addEventListener Aula 10 ex 06**

Na div pode tirar tudo aquilo pra não ficar poluído, e adicionar no script

a.addEventListener('click', clicar)       //addEventListener recebe 2 parametros:

a.addEventListener('mouseenter', entrar)  //o evento e a function q vai executar

a.addEventListener('mouseout', sair)

## **udemy > exerciciosSimples:**

<form id="form"> //primeiro exemplo

            <label>Informe sua idade: </label>

            <input type="number" id="age">

            <br><br>

            <button type="submit">Enviar</button> //colocando botao tipo submit, para funcionar

            <button id="btn">Diga ola</button> //segundo exemplo

            <script>

*const* form = document.getElementById("form")

              form.addEventListener('submit', enviar) //o evento e a function q vai executar

*function* enviar(*e*){

*e*.preventDefault() //para nao carregar a pagina sozinho

*const* idade = document.getElementById('age')

                    if (idade.value < 18){

                        alert('Voce é menor de idade')

                    }else if(idade.value >= 18){

                        alert('Voce é maior de idade')

                    }else if(idade.value > 60){

                        alert('Voce é um velhote')

                    }}

            </script>

</form>

Mais tipos acessa 🡪 <https://developer.mozilla.org/en-US/docs/Web/Events>

## Nova função

preventDefault()

Serve para nao carregar a página automaticamente

Cancela o comportamento que o elemento geralmente tem na página, então se o comportamento padrão de um link é abrir um site, nós vamos cancelar isso.

## **Condições**

* Condições Simples aula11 ex08

|  |
| --- |
| var velocidade = 80.5  console.log (`A velocidade do seu carro é ${velocidade}`)  if (velocidade > 70){      console.log (`Voce ultrapassou o limite de velocidade! Multado!`)  }  console.log(`Use sempre o cinto de segurança!`) |

* Condições Compostas aula11 ex09

|  |
| --- |
| aula11 ex09  var pais = "EUA"  console.log(`Vivendo em ${pais}`)  if (pais == "França"){      console.log('Voce é Brasileiro');  }else{      console.log('Voce é Estrangeiro')  }  aula11 ex10 em html  <body>      <h1>Paises</h1>      Em que país voce vive? <input type="text" name="nome" id="txtnome" >      <input type="button" value="Verificar" onclick="verificar()">      <div id="vive"></div>      <script>          function verificar(){              var pais = window.document.getElementById('txtnome')              var vive = window.document.querySelector('div#vive')              var nac = String(pais.value)              vive.innerHTML = `Vivendo em ${nac}`              if (nac == 'Brasil' || nac == 'brasil'){                  vive.innerHTML += "<p>Voce é Brasileiro.</p>"              }else{                  vive.innerHTML += "<p>Voce é Estrangeiro.</p>"              }          }      </script>  </body> |

## Elemento: propriedade: ClassList aula11 ex10.html

Usar classList: uma alternativa conveniente para acessar a lista de classes de um elemento como uma string delimitada por espaço via [element.className](https://developer.mozilla.org/en-US/docs/Web/API/Element/className).

Embora a classListpropriedade em si seja somente leitura, você pode modificá-la DOMTokenListusando os métodos [add()](https://developer.mozilla.org/en-US/docs/Web/API/DOMTokenList/add" \o "adicionar()), [remove()](https://developer.mozilla.org/en-US/docs/Web/API/DOMTokenList/remove), [replace()](https://developer.mozilla.org/en-US/docs/Web/API/DOMTokenList/replace" \o "substituir())e [toggle()](https://developer.mozilla.org/en-US/docs/Web/API/DOMTokenList/toggle" \o "alternar()).

<style>

        .blue{

*color*: blue; }

        .red{

*color*: red; }

</style>

<body>

<p id="text" class="texto">

        Lorem ipsum dolor sit amet consectetur adipisicing elit. Natus quis nobis.

</p>

<script>

        const texto = document.getElementById('text')

        texto.classList.add('blue') 

texto.classList.remove('texto') //vai remover a claas texto

      texto.classList.remove('blue') //vai tirar a cor azul

        texto.classList.add('red') // adicionando a cor vermelha



</script>

</body>

* Condições. Aninhadas

|  |
| --- |
| Aula12 ex12  var idade = 89  console.log(`Sua idade é ${idade}`)  if (idade < 16){      console.log('Voce NÂO VOTA.')  }else{      if (idade < 18 || idade >= 60){          console.log("Votação OPCIONAL")}      else{          console.log("Voto OBRIGATÓRIO.")} }  Aula12 ex13  var agora = new Date //para mostrar as hora  var hora = agora.getHours() //exata q estiver executando  console.log(`Agora sao ${hora} horas`)  if (hora < 12){      console.log("Bom diaaa")  }else if (hora <= 17){      console.log("Boa tardee")  }else{      console.log("Boa noiteee") } |

* Condições Múltiplas

|  |
| --- |
| Obrigatório o break  var agora = new Date() // dia atual  var diaSem = agora.getDay()  switch(diaSem) {      case 0:          console.log("Domingo")          break      case 1:          console.log("Segunda")          break        case 2:          console.log("Terça")          break      case 3:          console.log("Quarta")          break      case 4:          console.log("Quinta")          break      case 5:          console.log("Sexta")          break      case 6:          console.log("Sabado")          break  } |

Exercício de idades - aula12ex1

Para carregar o ano atual:

var data = new.data()

var ano = data.getFullYear() // ano atual

Exercício do horário - aula12ex2 horario.html

Dicas:

Para adicionar imagens no HTML pelo JS:

<body onload="carregar()">

    <section>

        <div id="msg">

            Aqui vai aparecera a hora

        </div>

    <script src="script.js"></script> <!--script separado tb-->

</body>

<script> img.src= ("fotos/manha.png")</script>

ou

<script> foto.innerHTML= '<img src="fotos/manha.png">'</script>

Para carregar a hora atual:

var agora = new Date //para mostrar as hora

var hora = agora.getHours() //exata q estiver executando

Para mostrar o dia atual:

var agora = new Date() // dia atual

var diaSem = agora.getDay()

Para que o código JS execute assim que a página é carregada, usa-se Onload no HTML:

onload=" ()"

aula12ex2 fotonoJS.html

Dicas:

Para adicionar imagens e titulo pelo JS :

<body>

    <div id="conteiner"></div>

    <script>

*let* conteiner = document.getElementById('conteiner')

*let* manha = document.createElement('img') //criando um elemento img

*let* titulo = document.createElement('h1') //criando um elemento h1

        titulo.innerHTML = 'Imagem de manhã' //add o caminho para o titulo

        manha.src = './fotos/manha.png' //add o caminho para a foto

        conteiner.appendChild(titulo) //add no html, para aparecer o titulo no site

        conteiner.appendChild(manha)  //add no html, para aparecer a imagem no site

        manha.style.display = 'flex'

        manha.style.width = '400px'

        manha.style.height = '400px'

        //conteiner.removeChild(titulo) //removendo do html

    </script>

</body>

Atributos aula12ex2 atributos.html

<body>

    <img src="fotos/noite.png" alt="noite" width="400px" id="imagem">

    <script>

*const* img = document.getElementById('imagem')

*const* atributo = img.getAttribute('alt') //pegar o valor do atributo



        console.log(atributo) //mostrando no console ==

img.setAttribute('width', '500px') //modifica o valor de um atributo existente num elemento específico

        img.removeAttribute('width') //vai remover o atributo escolhido

    </script>

</body>

## Evento **onSubmit** udemy > exercicios simples

É um evento que acontece quando tenta "submter" o formulário, ou seja, por exemplo quando clica num botão submit . Então você pode executar uma ou mais functions, pode fazer todas as validações que deseja.

<body>

    <main>

        <form onsubmit="enviar()">

            <label>Informe sua idade: </label>

            <input type="number" id="age">

            <br><br>

            <button type="submit">Enviar</button> //tem que ter um botao tipo submit, para funcionar

            <script>

*function* enviar(){

*const* idade = document.getElementById('age')

                    if (idade.value < 18){

                        alert('Voce é menor de idade')

                    }else if(idade.value >= 18){

                        alert('Voce é maior de idade')

                    }else if(idade.value > 60){

                        alert('Voce é um velhote')

                    }

                }

            </script>

        </form>

    </main>

</body>

## Evento **onLoad** aula12ex2 horario.html

Para que o código JS execute assim que a página é carregada, usa-se Onload no HTML:

onload=" ()"

## Colocando o nome do function do JS

function carregar()

Foi colocado no body: HTML:

<body onload="carregar()">

    <header>

        <h1>Hora do Dia</h1>

    </header>

    <section>

        <div id="msg">

            Aqui vai aparecera a hora

        </div>

        <div id="imagem">

            <img src="fotos/manha.png" alt="foto do dia">

        </div>

    </section>

    <footer>

        <p>&copy;Curso em Video</p>

    </footer>

    <script src="script.js"></script> <!--script separado tb-->

</body>

SCRIPT.JS :

function carregar() { //para carregar, coloquei no html(body)

    var msg = window.document.getElementById('msg')

    var foto = document.getElementById('imagem')

    var data = new Date()  //data atual

    var hora = data.getHours() //hora atual

    //var hora = 8

    if (hora >= 7 && hora < 12){

        msg.innerText = `Bom dia! Agora sao ${hora} horas.`

        img.src= ("fotos/manha.png")

        document.body.style.background = '#FEDC5A'

    } else if (hora >= 12 && hora <= 18){

        msg.innerHTML = `Boa tarde! Agora sao ${hora} horas.`

        foto.innerHTML= '<img src="fotos/tarde.png">'

        document.body.style.background = '#f59629'

    } else{

        msg.innerHTML = `Boa noite! Agora sao ${hora} horas.`

        foto.innerHTML= '<img src="fotos/noite.png">'

        document.body.style.background = '#1e4c5c'

    }

}

### **Repetições**

* Estrutura de while Aula13

|  |
| --- |
| .  //esse faz o teste antes e executa o comando  var c = 1  while (c <= 6) { //enquanto o contador der 6..      console.log(`Passo ${c}`)      c++ //c = c + 1  } |

* Estrutura Do while Aula13

|  |
| --- |
| .  //esse executa o comando e  depois faz o teste  var s = 1  do {      console.log(`Passo ${s}`)      c++  } while (s <= 6) |

* ForEach Aula13 🡪 tem mais 2 exercicios la no VsCode

|  |
| --- |
| //repetição com array  *const* cidades = ['São Paulo', 'Rio de Janeiro', 'Curitiba', 'Joinville']    cidades.forEach( (*elemento*) *=>* { //add uma variavele      console.log(*elemento*)        //e uma função anônima 🡪  })    //mostrando o indice junto  cidades.forEach( (*elemento*, *index*) *=>* { 🡪      console.log('Executando o:', *index*)      console.log(*elemento*)  })  console.log('') |

* Estrutura de for Aula14

|  |
| --- |
| /\*var c = 1  while(c <= 5) {      console.log(c)      c++  }\*/  for (var c = 1 ; c<=10; c++){      console.log(c) } |

Aula 14 ex 2

function tabuada() { //para carregar, coloquei no html

    var num = document.getElementById('txtn')

    var tab = document.getElementById('seltab')

    if (num.value.length == 0) {

        window.alert('Por favor, digite um numero!')

    } else {

        var n = Number(num.value)

        var c = 1

        tab.innerHTML = '' //antes de mostrar a tabuada, limpe a tabela antes

        while (c <= 10){

            var item = document.createElement('option') //criar um elemento de opçoes

            item.text = `${n} x ${c} = ${n\*c}`

            tab.appendChild(item) //adicionar os itens

            c++

        }    }}

## **Aula 15 – Variáveis Compostas usando colchetes []**

Uma variável composta tem vários elementos, cada elemento é composto por um valor e por uma chave de identificação.



let num = [8, 4, 3, 1, 9]

num [3] = 6 //adicionar na posição 3 o nº 6

num.push(7) //adicionar no final

console.log(num)

console.log(`Nosso vetor é ${num}`)

console.log(`O vetor tem ${num.length} elementos`) //para ver quantos elementos tem

console.log(`Deixando em ordem crescente ${num.sort()}`)

console.log(`O primeiro valor agora é ${num[0]}`)

/\*for (let pos = 0; pos < num.length; pos++){ //enquanto ele n chegar no final do vetor..

    console.log(`A posição ${pos} = ${num[pos]}`)

}\*/

// Mais simplificado:

for(let pos in num) { //para cada posição dentro de num

    console.log(`A posição ${pos} tem valor ${num[pos]}`)

}

var pos = num.indexOf(7) //vai mostrar em que posição esta

console.log(`O valor 7 esta na posição ${pos}`)

## **Aula 16 – Funções**

* São ações executadas assim que são chamadas ou em decorrência de algum evento.
* Uma função pode receber parâmetros e pode retornar um resultado.

function parimp(n) {

    if (n % 2 == 0) {

        return `O numero ${n} é PAR` EX1

    } else {

        return `O numero ${n} é IMPAR`

    }

}

let res = parimp(1127)

console.log(res)

*function* soma(*n1*=0, *n2*=0) { //se o n1 ou n2 n for chamado é = 0

    return *n1*+*n2* EX2

}

console.log(soma(4))

let v = function(x) { //uma variavel recebe uma função EX3

    return x\*2 //o dobro

}

console.log('O dobro de 5 é', v(5))

console.log('O dobro de 7 é', v(7))

function fatorial(n){

    let fat = 1

    for (let c = n; c > 1; c--) {

        fat \*= c EX4

    }

    return fat

}

console.log(fatorial(5))

let msg = ('O fatorial de 5 é '+ fatorial(5))

console.log(msg)

//RECURSIVA

*function* fatorial(*n*) {

    if (*n*==1) {

        return 1

    } else {

        return *n* \* fatorial(*n*-1)

    }

}

console.log(fatorial(5))

## Colocando tabela pelo JS e aparecer no HTML **Ex: Documents\Projetos HTML e CSS\Musculacao**

var container = document.getElementById("container");

container.innerHTML = [

'<table>',

'<thead>',

'<tr>',

'<th>id</th>',

'<th>col1</th>',

'<th>col2</th>',

'<th>col3</th>',

'</tr>',

'</thead>',

'<tbody>',

'<tr>',

'<td>1</td>',

'<td>data</td>',

'<td>data</td>',

'<td>data</td>',

'</tr>',

'<tr>',

'<td>2</td>',

'<td>data</td>',

'<td>data</td>',

'<td>data</td>',

'</tr>',

'<tr>',

'<td>3</td>',

'<td>data</td>',

'<td>data</td>',

'<td>data</td>',

'</tr>',

'</tbody>',

'</table>'

].join("\n");

### [Utilizando metodos tostring e join em listas](https://www.dio.me/articles/utilizando-metodos-tostring-e-join-em-listas-com-javascript)

## Manipulando Tabelas com Javascript

<https://www.youtube.com/watch?v=ImkWbhXnISM>

Onde o usuário insere os dados solicitados no

formulário da pagina em HTML. Mostrando na tabela

quando clicar em adicionar.

## Busca em tabela com Javascript

<https://www.youtube.com/watch?v=HWZfqk6gSfI>

Onde o usuário quer fazer uma buscar de nome

na tabela

## [Clicar no botão que leva para uma outra parte da mesma página](https://pt.stackoverflow.com/questions/175871/click-no-link-que-leva-para-uma-outra-parte-da-mesma-pagina) 🡪 Projeto Meu\_site\_de\_sites

Usa href="#idDoElement" com o elemento da ID. Assim quando clicares na ancora a página muda o scroll para mostrar esse elemento.

<a href="#texto5"><h1>Serviços</h1></a>

<a href="#contatos"><h1>Contatos</h1></a>

Com scroll suave:

Colocando:  scroll-behavior: smooth no html

html{

*scroll-behavior*: smooth; /\*rolagem suave\*/

}

Só fica uma pequena questão: a tela vai rolar para o exato ponto em que o título (ou o conteúdo que seja) começa sendo bastante comum o “corte” do texto desse título — o que não é bom para a experiência de quem acessa o site.

Felizmente, agora é possível contar com a propriedade scroll-margin-top.

html{

*scroll-behavior*: smooth;

}

:target {

*scroll-margin-top*: 5em;

}

main #responsivo:target{

*scroll-margin-top*: 10em;

}

Perceba que :target pode ser usado em elementos específicos, mas, assim como muitos outros pseudo em CSS, se usado “solto”, representa a estilização em tudo o que é afetado pelo estilo em questão.

## [Como adicionar em um botão para voltar ao topo da página](https://pt.stackoverflow.com/questions/245814/como-adicionar-em-um-bot%c3%a3o-a-anima%c3%a7%c3%a3o-para-voltar-ao-topo-da-p%c3%a1gina-usando-a-bib) 🡪 Projeto Vet São Lourenço

HTML:

<div class="scroll-top on">

        <a id="botao-top" onclick="scrollToTop()"><img src="imagens/setapcima.png"

alt=""></a>

</div>

CSS:

.scroll-top {

*position*: fixed;

*bottom*: 35px;

*right*: 30px;

}

#botao-top {

*cursor*: pointer;

}

O nome que eu coloquei na função foi ScrollToTop. No nosso JavaScript iremos fazer uma const para armazenar a nossa arrow function, dentro dela colocamos o método window.scrollTo() que vai fazer a rolagem para determinado ponto da página.

JS:

*const* scrollToTop = () *=>* {

    window.scrollTo({

   top: 0,

   behavior: 'smooth'

    })

}

Dentro do window.scrollTop() vamos passar, entre chaves, o ponto que no caso seria o topo da nossa página, o topo inicial de uma página é zero.

Por fim, adicionamos o comportamento dessa rolagem para que ela fique suave aos olhos do usuário.

## Colocando a opçao que fica “invisível” na tela e aparece apenas quando o usuário estiver rolando a página **ainda não achei o codigo certo**

## [Quando clicar no botão ir para fazer o cadastro](https://pt.stackoverflow.com/questions/464160/quando-clicar-no-botao-ir-para-fazer-o-cadastro) por JS

Tem 2formas:

1. Redirecionando com onclick= window.location

<div class="inscrever">

        <input class="bnt st-button" type="button" value ="Inscrever-se"/>

        <input class="bnt es-button" type="button" value ="Entrar">

</div>

obs: location é sinonimo de location.href

<input class="bnt st-button" type="button" value ="Inscrever-se" onclick="window.location='https://pt.stackoverflow.com/questions';" />

<input class="bnt st-button" type="button" value ="Entrar" onclick="window.location='https://pt.stackoverflow.com/';" />

1. função javascript

<input class="bnt st-button" type="button" value ="Inscrever-se" onclick="novaPagina(0)">

<input class="bnt es-button" type="button" value ="Entrar" onclick="novaPagina(1)">

*function* novaPagina(*num*) {

*var* url=new *Array*();

    url[0]="https://pt.stackoverflow.com/questions";

    url[1]="https://pt.stackoverflow.com/";

    window.location=url[*num*];

}

Usando Bootstrap

## Ex Documents\Faculdade TI\Pós Front-end\Meus\_exercicios\_da\_Facul\LinguagensEPadroesWeb\Tema4\Bloco3

[getbootstrap.com](https://getbootstrap.com/docs/4.0/getting-started/introduction/)

*let* input = document.querySelector('input[name=tarefa') /\*busque para mim um input que tem o nome tarefa \*/

*let* btn = document.querySelector('#botao');

*let* lista = document.querySelector('#lista')

*let* card = document.querySelector('.card')

*let* tarefas = JSON.parse(localStorage.getItem('tarefas')) || [] /\*buscar o local de armazenamento, ele vai buscar tds os elementos da tarefa, ou se ele nao encontrar o localStorage, vai me mostrar vazio\*/

/\*getItm: buscar a informação

  setItem: utilizada para armazenar um valor no local storage\*/

O código anterior cria as variáveis que serão utilizadas no decorrer do código. Temos os querySelector, que buscam elementos pelo texto inserido entre parênteses e como parâmetros. A variável que se difere das demais é a “tarefas”, que cria um vetor com as tarefas registradas. Note que o conceito de seletores em JavaScript são o que criam a integração entre os elementos HTML e as funções programadas em tal linguagem.

Na sequência, temos a função renderizarTarefas(), que cria uma lista de itens que serão exibidos. Essa lista receberá todas as tarefas inseridas. Temos, no corpo desta função, um for que percorre todas as tarefas, povoando a lista e criando-a.

*function* rendenizarTarefas(){

    lista.innerHTML = ''

    for(tarefa of tarefas){ /\*vai percorrer toda lista que tenho, criando uma lista\*/

*let* itemLista = document.createElement('li')

        itemLista.setAttribute('class', 'list-group-item list-group-item-action') /\*lista do bootStrap\*/

        itemLista.onclick = *function*(){

            deletarTarefa(this)

        }

        /\*a partir que eu excluir um item da lista, ele vaideslocar todo os valores q eu tinha armazenado p um nivel anterior, para nao ficar nenhum "buraco" no registro que eu removi\*/

*let* itemTexto = document.createElement(tarefa)

        itemLista.appendChild(itemTexto) /\*insirir um texto na minha lista\*/

        lista.appendChild(itemLista)

    }

}

rendenizarTarefas();

A seguir, temos a variável btn, que referencia o botão e que, no momento em que este for “clicado” pelo usuário, ativará a função que, por meio de uma estrutura condicional if, verifica se a novaTarefa não está vazia e adiciona no LocalStorage, por meio da função salvarDadosNoStorage().

A chamada da função removerSpans() cria um alerta com possíveis problemas, caso o texto da tarefa esteja vazio, exibindo ainda uma mensagem para que o usuário entre com tal texto.

btn.onclick = *function*(){ /\*coloquei a função direta\*/

*let* novaTarefa = input.value; /\*pega o valor do input, joga para uma variavellocal, para verificar \*/

    if(novaTarefa !== ""){ /\*caso nao esteja vazio\*/

        tarefas.push(novaTarefa)/\*vai inserir tds as tarefas\*/

        rendenizarTarefas() /\*função para mostrar as tarefas\*/

        input.value = '';

        removerSpans() /\*função p remover os espaços\*/

        salvarDadosNoStorage() /\*função p salvar os dados\*/

    }else{ /\*se tiver tarefas\*/

        removerSpans()

*let* span = document.createElement('span')

        span.setAttribute('class', 'alert alert-warning')/\*alerta do bootStrap\*/

*let* msg = document.createTextNode('Voce precisa digitar a terafa que deseja registrar!')

        /\*caso o usuario nao digitar nenhuma tarefa, vai mostrar a mensagem de alerta\*/

        span.appendChild(msg)/\*vai inserir a msg dentro do span\*/

        card.appendChild(span)/\*dps vai inserir o span dentro do card\*/

    }

}

*function* removerSpans(){

*let* spans = document.querySelectorAll('span')

    for(*let* i = 0; i < spans.length; i++){

        card.removeChild(spans[i])

    }

}

A função deletarTarefa() é adicionada à tarefa que foi acrescentada, por meio da função renderizarTarefas(). Faz com que a tarefa da lista, ao ser “clicada” pelo usuário, seja excluída.

*function* deletarTarefa(*tar*){

    tarefas.splice(tarefas.indexOf(*tar*.textContent), 1) /\*splice: busca um indice de um determinado conteudo, e remove aquele conteudo\*/

    /\*indexOf(): utilizado para retornar o índice da primeira ocorrência

    de uma determinada palavra ou caractere em uma string,

    lembrando que a contagem das posições começa sempre do zero.\*/

    rendenizarTarefas()

    salvarDadosNoStorage()

}

Finalmente, a última função coloca, no localStorage, a tarefa digitada pelo usuário. O localStorage, ou armazenamento na web, também conhecido como armazenamento DOM, fornece ao desenvolvedor a possibilidade de armazenar dados no lado do cliente. É como se tivéssemos um banco de dados local, inserido no próprio navegador, que mantém as informações armazenadas até serem totalmente removidas pelo usuário. Assim, já temos um projeto inicial que trabalha, inclusive, com o localStorage.

*function* salvarDadosNoStorage(){

    localStorage.setItem('tarefas', JSON.stringify(tarefas))

    /\*setItem: utilizada para armazenar um valor no local storage\*/

}

O framework do Bootstrap adiciona uma folha de estilos CSS específica, com suas próprias classes. Com sua documentação e a comunidade que a mantém atualizada, temos uma ótima alternativa de estilização sem muitos esforços,. bastando apenas, sabermos as classes a serem utilizadas, o que pode ser consultado diretamente na documentação do Bootstrap (BOOTSTRAP, 2021).

Agora, para finalizar o projeto, utilize seus conhecimentos para alterar as páginas e também ajustar a integração da folha de estilos criada para a página inicial com o CSS do Bootstrap.