**JavaScript**

Conteúdo = Html (tecnologia de construção de site)

Design = CSS (tecnologia de construção de site)

Interatividade = JavaScript (linguagem de programação)

ECMAScript = Linguagem JavaScript padronizada

Versão ES6 é a mais popular e a que será estudada

Microsoft code

**Bibliografia:**

JavaScript, O Guia definitivo (David R)

JavaScript (Mauricio SS - MauJor)

Grátis - Guia de referência da Mozilla e da ECMA

**Programas para ajudar no estudo:**

Visual Studio Code – plugin liver server e live server web extension

NodeJs – plugin node.js Exec

1. **JavaScript (Comandos):**
   1. **uma única linha 🡪** //
   2. **mais de uma linha 🡪** /\* \*/
   3. **recebe 🡪 =**
   4. **nulo 🡪** null
   5. **variável 🡪** var
   6. **cadeia de caracteres 🡪** string
   7. **nome de variável 🡪** identificadores:
      1. **podem começar com 🡪** letra, $ ou \_
      2. **não podem começar com números**
      3. **é possível usar letras e números**
      4. **é possível usar acentos e números**
      5. **não pode ter espaço**
      6. **não podem ser palavras reservadas**
   8. **dicas 🡪** identificadores**:**
      1. maiúsculas e minúsculas fazem diferença
      2. usar nomes coerentes
   9. **Dados // tipos primitivos // typeof // comando**
      1. number// JS não diferencia números inteiros e reais)
         1. infinity
         2. NaN // Not-a-Number
      2. string // cadeia de caracteres, que podem ser letras, números etc
      3. boolean // true false
      4. null
      5. undefined
      6. object
         1. array
      7. function
2. **Manipulação de Dados**
   1. **Comentário em html** 🡪 <!-- isso é um comentário em html -->
   2. **Comentário em CSS** 🡪 /\* isso é um comentário em CSS \*/
   3. **Concatenação ou adição 🡪** +
   4. Conversão de **string** em **number**
      1. Number.parseInt(n) 🡪 número inteiro
      2. Number.parseFloat(n) 🡪 número real
      3. Number 🡪 ele se vira
   5. Conversão de **number** em **string**
      1. String(n)
      2. n.toString()
   6. Template String

var s = ‘JavaScript’

* + 1. ‘Eu estou aprendendo s’ // não faz interpolação
    2. ‘Eu estou aprendendo’ + s // usa concatenação
    3. `Eu estou aprendendo ${s}` // usa template string
    4. s.lenght // quantos caracteres a string tem
    5. s.toUpperCase() // para MAIÚSCULAS
    6. s.toLowerCase() // para minúsculas
    7. s.toLocaleString('pt-br', {style: 'currency', currency: 'BRL'}) // localizar
    8. s.toFixed(2).replace('.', ',') // concertar

1. **Operadores** 
   1. **Aritméticos**
      1. **Operadores Binários**
         1. 5 + 2 🡪 7
         2. 5 – 2 🡪 3
         3. 5 \* 2 🡪 10
         4. 5 / 2 🡪 2.5
         5. 5 % 2 🡪 1 // resto da divisão inteira
         6. 5 \*\* 2 🡪 25 // ao quadrado
      2. **Ordem de Precedência de Operadores Aritméticos**
         1. ()
         2. \*\*
         3. \* / %
         4. + -
   2. **Atribuição**
      1. var a = 5 + 3 🡪 8
      2. var b = a % 5 🡪 3
      3. var c = 5\*b\*\*2 🡪 45
      4. var d = 10 - a/2 🡪 6
      5. var e = 6\*2/d 🡪 2
      6. var f = b % e + 4 / e 🡪 3
         1. **Operadores de Atribuições (Simplificando):**
            1. +=
            2. -=
            3. \*=
            4. /=
            5. \*\*=
            6. %=
         2. **Operadores de Incremento:**
            1. ++
            2. \_ \_
   3. **Relacionais**
      1. 5 > 2 🡪 true
      2. 7 < 4 🡪 false
      3. 8 >= 8 🡪 true
      4. 9 <= 7 🡪 false
      5. 5 == 5 🡪 true
         1. 5==’5’ 🡪 true para JavaScript
         2. **Operador de Identidade** 
            1. Igualdade Restrita

5 === ‘5’ 🡪 false

5 === 5 🡪 true

* + - * 1. Desigualdade Restrita

5 !== ‘5’ 🡪 true

5 !== 5 🡪 false

* + 1. 4 != 4 🡪 false // != é diferente
  1. **Lógicos** 
     1. ! 🡪 **negação (não)**
        1. True
        2. False
     2. && 🡪 lê como “e comercial” // **conjunção** **(e)**
        1. True && True 🡪 true
        2. True && False 🡪 false
        3. False && True 🡪 false
        4. False && False 🡪 false
     3. || 🡪 lê como pipe // **disjunção (ou)**
        1. True || True 🡪 true
        2. True || False 🡪 true
        3. False || True 🡪 true
        4. False || False 🡪 false
     4. **Ordem de execução** (em caso de estarem presentes Não, E, Ou)
        1. Não
        2. E
        3. Ou
  2. **Ternário**
     1. Tem esse nome porque é composto de 3 partes:
        1. Teste ? True : False
        2. ex.: Média>=7 ? ”aprovado” : “reprovado”
     2. ?
     3. :

1. **Ordem de Precedência (Resumão!)**
   * + 1. **Operadores aritméticos**
          1. ()
          2. \*\*
          3. \* / %
          4. + -
       2. **Operadores Relacionais**
          1. >
          2. <
          3. >=
          4. <=
          5. ==
       3. **Operadores Lógicos** 
          1. Não 🡪 !
          2. E 🡪 &&
          3. Ou 🡪 ||
2. **DOM (Document Object Model)**
   1. O que é?
      1. Vertente web para o JavaScript
   2. Árvore DOM
      1. Window
         1. Location
         2. Document
            1. Html

Head

Meta

Title

Body

h1

p

p

strong

div

* + - 1. History
      2. ...
    1. Técnicas para Selecionar o DOM
       1. Por marca
          1. getElementsByTagName()
       2. Por ID
          1. getElementsById ()
       3. Por nome
          1. getElementsByName()
       4. Por classe
          1. getElementsByClassName()

classe representada por 🡪 **.**

* + - 1. Por seletor
         1. querySelector()
         2. querySelectorAll()

query representada por 🡪 #

* 1. Eventos DOM
     1. O que é:
        1. Tudo que pode acontecer com uma <div>
     2. Procurar “event reference” no site developer mozilla
        1. Exemplo de eventos no Mouse:
           1. mouseenter
           2. mousemove
           3. mousedown
           4. mouseup
           5. click
           6. mouseout
  2. Funções em JS
     1. O que são?
        1. Conjunto de códigos que serão executados quando o evento ocorrer
     2. Bloco
        1. Conjunto de linhas (que serão disparadas somente quando o evento ocorrer)
        2. Bloco é delimitado por { }
        3. Bloco é nomeado com a palavra 🡪 function
        4. A function também tem que ser nomeada 🡪 function ação()
        5. Dentro pode haver parâmetros 🡪function ação(param)

1. **Condições Simples em JS**
   * + 1. extensão .js (diferente de html)
          1. console.log()
2. if(condição) {

true

}

1. **Condições Compostas em JS**
   * 1. if(condição) {

true

} else {

false

}

1. **Condições Aninhadas em JS**
   * 1. if(cond1) {

bloco1

} else {

If(cond2){

Bloco2

} else {

Bloco3

}

1. **Condições Múltiplas em JS**
   * 1. switch(expressão) {

case valor 1:

break

case valor 2:

break

case valor 3:

break

default:

}

1. **Modelo**
   1. Html
      1. Body
         1. Header (Título)<h1> </h1>
         2. Section
            1. Div (divisões) <div> </div>
            2. Div (divisões) <div> </div>
         3. Footer <p> </p>
2. **Laço / Looping / Repetições**
   1. Estrutura de repetição com teste lógico no **início** (mais comum)
      1. while(condição) {

}

* + - 1. Ex.: function comerPizza(){

while(temFatia()) {

}

}

* 1. Estrutura de repetição com teste lógico no **final** (mais rara)
     1. Do {

} while (condição)

* 1. Estrutura da Repetição com variável de controle
     1. for(inicio ; teste logico ; incremento) {

}

* 1. Depuração
     1. Na aba “Run”, clicar em “Starting Debugging”

1. **Variáveis compostas / Arrays**
2. **Uso de funções e eventos**
3. **Passagem de parâmetros**