# Antes de mais nada...

Diferente de outras linguagens, JavaScript não necessita de ponto-e-vírgula para encerrar linhas de códigos, salve algumas situações que você verá mais a frente.

# Console.log

Para escrever no console, exibir resultados… este é o Console.log.

*console.log(“Olá!”)*

Pode exibir variáveis, objetos, funções, etc.

*console.log(numero1)*

**TENTE VOCÊ MESMO**

Tente escrever no console a seguinte frase: Olá, sou [SEU NOME]

# Comentários

Comentários são anotações que não serão consideradas como funções e ignoradas durante a compilação. Muito bom para explicar funções no seu código.

**Comentários de 1 linha:**

*//Isso é um comentário*

**Comentário de várias linhas:**

*/\**

*Isso é*

*um*

*comentário*

*\*/*

# HTML + JavaScript

dentro da tag *<body>* no HTML, inserir no final e antes do seu fechamento a tag <script>, referenciando a localização do arquivo Js.

*<script src=”index.js”></script>*

# Variáveis com let

Variável é o conceito de salvar algo na memória em tempo de execução do código.

As variáveis armazenam qualquer tipo de dado, seja números ou palavras, etc.

*let mensagem = “Olá!”*

*let ano= 2022*

As variáveis podem mudar de valor a todo momento.

*mensagem = “Olá Mundo!”*

# Constantes com const

Semelhante às variáveis, mas não podem ter o valor alterado.

*const pi = 3.14*

# Tipos de Dados

Existem vários tipos de dados em JS, sendo eles:

* string: textos
* number: inteiros, reais
* undefined: Não aponta para nenhum lugar na memória e não tem valor algum
* null: também não aponta para nenhum lugar na memória, mas tem um valor nulo
* boolean: verdadeiro ou falso

# Operadores

## Aritméticos:

* + : soma
* - : subtração
* \* : multiplicação
* / : divisão
* \*\* : potenciação
* % : resto da divisão

## Atribuição:

* = : atribui um valor

## Incremento:

* ++ : soma mais 1
* – : decrementa 1

# Strings

O JavaScript possui várias funções para manipulação de string, sendo alguma delas:

* minhaString.charAt(x) : retorna o caractere na posição x da string
* minhaString.indexOf(‘A’) : retornará o index da letra A, caso tenha. Se não, -1
* minhaString.search(‘x’) : retorna o index do que deseja ser procurado
  + - minhaString.match(‘x’) : uma busca mais detalhada
    - minhaString.match(/[a-z]/g) : ainda mais detalhado, retornando todas as minúsculas
* minhaString.replace(/Ola/, ‘Oi’) : retornará a string com a troca da palavra Ola por Oi
  + minhaString.replace(/Ola/g, ‘Oi’) : trocará todos os Ola por Oi
* minhaString.length : retorna o tamanho da string
* minhaString.slice(x,y) : retornará os caracteres de x até y
* minhaString.split(‘ ’) : ira dividir a string, no caso, a cada espaço, dividirá as palavras

# Numbers

O JavaScript possui várias funções para manipulação de numbers, sendo alguma delas:

* num1.toString() : transforma o número em string
  + num1.toString(2) : retornará a representação binária do número
* num1.toFixed(x) : retornará o número com apenas x casas decimais
* Number.isInteger(num1) : retornará true ou false, caso o número seja inteiro
* parseFloat(num1) : transformará o valor em número Real

# Objeto Math

O *Math* é um objeto com várias funções matemáticas que podemos utilizar, sendo alguma delas:

* Math.floor(num1) : arredonda para baixo
* Math.ceil(num1) : arredonda para cima
* Math.round(num1) : arredondamento comum
* Math.max(1,2,3,4) : retorna o maior número da sequência (no caso, o 4)
* Math.min(1,2,3,4) : retorna o menor número da sequência (no caso, o 1)
* Math.random() : retorna um numero aleatório entre 0 e 1
  + Math.random() \* (x-y) + y : número aleatório entre x e y
* Math.pow(x,y) : retorna o resultado de x elevado a y

# Arrays – Básico

De primeiro momento, pense que um array como se fosse uma lista, ou seja, uma coleção de coisas

*const alunos = [‘Ana’, ‘Marcos’, ‘Maria’]*

Os arrays são criados a partir da atribuição de um colchetes [ ]. Cada elemento dentro do array, equivale a um índice, ou seja, no índice *0*, encontramos o nome *Ana*.

Para mudar um nome presente dentro do array, basta:

*alunos[0] = ‘João’*

*Assim, mudamos o valor que estava no índice 0*

Para adicionar um novo elemento no final do array, basta:

*alunos.push(‘Letícia’)*

Para adicionar um novo elemento no início do array, basta:

alunos.unshift(‘Gustavo’)

Para remover o elemento final do array, basta:

*alunos.pop()*

obs.: quando essa função é executada, é capaz de armazenar o valor que foi removido

*const removido = alunos.pop()*

Para remover o elemento inicial do array, basta:

*alunos.shift()*

Para remover um elemento, mas sem alterar os índices do array, basta:

*delete alunos[1]*

*assim, o índice 1 fica com um espaço vazio, mas não altera os demais elementos*

Para dividir o array, basta

*alunos.slice(x,y)*

*assim, retorna os valores do índice x até o índice y-1*

# Funções – Básico

Para declarar uma função, basta:

*function minhaFuncao() { }*

Para executa-lá, basta:

*minhaFuncao()*

É possível enviar variáveis como parâmetro, para serem usadas dentro de uma função:

*minhaFuncao(nome) {*

*console.log(`Olá ${nome}`)*

*}*

*minhaFuncao(‘Silas’)*

É possível fazer com que uma função retorne algum valor, para ser atribuído em uma variável, basta:

*function somar(num1, num2){*

*return num1 + num2*

*}*

*let resultado = somar(2, 3)*

Obs.: ao inserir um *return*, nenhum código abaixo dele na função será executado.

É possível deixar um parâmetro pré-definido na função, para evitar um resultado não desejado, veja:

*function somar(num1 = 0, num2 = 0) {*

*return num1 + num2*

*}*

*let resultado = somar(2)*

*Nesse caso, o JavaScript atribuirá o valor de 2, para a variável no parâmetro num1 e fará a conta normalmente*

## Outras formas de declarar funções

* *const funcao = function ( ){ };*
* *const funcaoo = ( ) => { };*
* *const funcao = n => ... ;* :para funções de 1 linha, podem ser declaradas assim

obs.: com essas declarações, é **obrigatório** o fechamento da linha com ponto e virgula ( ; )

# Objetos – Básico

Para evitar uma criação de várias variáveis sobre a mesma coisa (exemplo: nome1, nome2, idade1, idade2), criamos um objeto, para agrupar variáveis e funções que podem ser reutilizadas, sem que seja necessário a declaração de novas variáveis e funções.

Observe o exemplo abaixo:

*let nome1 = ‘silas’*

*let idade1 = 21*

*let nome2 = ‘maria’*

*let idade2 = 20*

Para isso, utilizaremos um objeto:

*const pessoa = {*

*nome: ‘silas’,*

*idade: 21,*

*fala(){*

*console.log(`ola, sou o ${this.nome}`)*

*}*

*}*

Podemos também, ter uma função criadora de objetos:

*function criaPessoa(nome, idade) {*

*return {*

*nome: nome,*

*idade: idade,*

*fala(){*

*console.log(`ola, sou o ${this.nome} e tenho ${this.idade}`)*

*}*

*}*

*}*

*const pessoa1 = criaPessoa(‘Silas’, 21)*

# Protegendo seu Script

Durante a execução de uma página Web, suas funções e variáveis podem entrar em conflito com funções e variáveis já declaradas por uma biblioteca ou afins. Para evitar esse conflito, uma forma de proteger nosso script é criar um código JavaScript, proteja-o dentro de uma função:

*function meuEscopo ( ) {*

*...SEU CÓDIGO...*

*}*

*meuEscopo( )*