**JavaScript**

O que é o JavaScript – é uma linguagem de programação de alto nível ou seja: uma linguagem que assemelha a linguagem humana, linguagem que integra o desenvolvimento de apps e paginas web.

Manipulando um arquivo:

**\*atalhos para comentários: CTRL + /**

Variáveis e constantes são uma operação de primeiro grau, quando o **X** pode ter seu valor alterado mas **A** e **B** não. Exemplo: **ax + b = a\*x + b**, **X** pode ter seu valor alterado **a e b** não. **X neste caso é a variável.**

**Tipos de Variáveis:**

**Var** – **escopo global e local** pode ter o valor alterado, se não tiver um valor inicial será tratada como null (nulo). Variável pode: redeclarar, reatribuir, hoisting.

**Let – escopo local e bloco**, pode ter o valor alterado, se não tiver um valor inicial será tratada como null (nulo). Let pode: reatribuir.

**Const – escopo local bloco**, somente leitura o valor inicial é obrigatório e não pode ser alterado. **NÃO PODE**: redeclarar e reatribuir

**Hoisting** - **escopo global e bloco**,

Exemplos de escopo global e local

obs: o console vai mostrar a variável no

inspecionar pois ela esta fora do bloco (a amostra)

var escopoGlobal = ‘global’;

Console.log(escopoGlobal);

function escopoLocal () {

obs: só será exibido no inspecionar

se o console.log estiver dentro do

bloco de código (que neste caso foi a funcion {} )

let escopoLocalInterno = ‘local’;

Console.log(escopoLocalInterno);

}

**O sinal de “ = ” em JavaScript, significa ATRIBUIÇÃO**

**O sinal de “ = = ” em JavaScript, significa COMPARAÇÃO**

Exemplo:

**Como declarar:** 0 = = “0”;

**Como ler:** “0” tem o valor igual a 0?

**//nesse caso retorna true**

Exemplo:

**Como declarar:** var nome = “meu nome”;

**Como ler:** variavel nome recebe o valor meuome

number

string

**O sinal de “ == = ” em JavaScript, significa COMPARAÇÃO IDÊNTICA**

Exemplo:

**Como declarar: “**0” = == 0;;

**Como ler:** “0” tem o valor e o tipo idêntico a 0?

**//nesse caso retorna false (false pq são de valores igual mas de tipos diferentes)**

**EXEMPLO DE VARIAVEL:**

**Variável:**

Var preco = 2;

Var desconto = 0.2;

Preco = preco – desconto;

EXEMPLO DE CONSTANTE:

**Constante:**

Var desconto = 0.2;

Const PRECO = 2; \*por ser uma constante o valor “preço não pode ser alterado”

**Var total = PRECO – desconto;**

Esses parênteses é um espaço para ser declarados “atributos” parâmetros da sua função

**FUNÇÃO:**

function soma ( ) { }

Declarando uma função

ex:

function soma ( a, b ) {

Return = informa o que essa função faz Retornar o valor de a +b

return a + b;

}

Para chamar a function:

Chamando uma função

soma ( 2, 5 );

Console.log (a+b) 🡪 comando para verificar se a função esta sendo chamada, se o valor esta realmente funcionando e etc.

**DOM**: Document Object Model

Basicamente são os elementos que existem na pagina

Addeventlistner 🡪 pesquisar sobre.

**FRAMEWORKS E BIBLIOTECAS para desenvolvimento Web em JavaScript**

* VueJS
* Angular
* React
* jQuery

**Sites para estudos e pesquisar DEv**

* W3C
* Stackoverflow
* Youtube
* MDN web docs moz://a
* Github
* Linkeding
* Twitter

**VARIAVEIS E TIPOS:**

**Estrutura de Dados**

**Strings 🡪** comumente utilizada para textos 🡪 valores declarados entre aspas ou crases;

Metodos importantes da Instring:

* Concatenação; juntar mais de uma string
* Propriedade length; com ela vc pode ver o tamanho da sua string
* Iterabilidade: voce pode checar o que tem em cada indice / caractere da string
* Formatação; pode muidar tudo para maiusculo ou minusculo, separar por espaço etc.
* Index das letras.

**Number 🡪** Numeros inteiros ou decimais **🡪** é declarado sem aspas

**Booleans 🡪** VERDADEIRO OU FALSO

**Arrays 🡪** Listas iteraveis de elementos 🡪 representrado dentro de [ ]

Array.push – adicional elemento no final

Array.pop – tira o elemento do final

Array.shift – tira o primeiro elemento

Array.anshift – adiciona elemento no inicio

**Objetos 🡪** começa declarando a variavel seguido do sinal de = e {}

Estrutura tipo “chave e valor”

**Empty, null e undefined 🡪**

**Empty:**

**Null:**

**Undefined:**

**FUNÇÕES**

**Obs:** Variáveis criadas dentro de uma função apenas podem ser utilizadas dentro dela.

**Tipos de função:**

**Estrutura:**

Function nome (parametros ) {

//instruções

}

Function nome (parametros ) {

//instruções

return; //valor de rertorno

}

**FUNÇÃO ANÔNIMA:**

Const soma = function (a,b) {

Return a + b;

}

**Soma(1, 2 )** //3

**Soma(5, 4 )** //9

**FUNÇÃO AUTOINVOCÁVEL**

(

function( ) {

**Obs** uma função anônima entre parênteses, seguida por outro par de parênteses, que representa sua chamada.

let name = “Digital Innovetion One”

return name;

}

) ( );

//Digital Innovation One

**OU**

(

OU

Const soma3 = (

Function( ) {

Return a + b;

}

) (1, 2);

Console.log(soma3) //3

function(a, b) {

**Obs:**  também podem ser utilizadas com parâmetros ou armazenada em uma variável

return a + b;

}

)(1, 2); //3

**FUNÇÃO CALLBACKS**

**Obs:** Utilizando call-backs, você tem maior controle da ordem de chamadas.

Const calc = function (operação, num1, num2){

return operação(num1, num2);

}

Const soma = function (num1, num2){

return num1 + num2;

}

Const sub = function (num1, num2){

return num1 - num2;

}

Const resultSoma = calc(soma, 1, 2);

Const resultSub = calc(sub, 1, 2);

Console.log(resultSub); // -1

Console.log(resultSoma); // 3

Async / await 🡪

API 🡪 É uma forma de intermediar os resultados do **back-end** com o que é apresentado no **front-end.** Você consegue acessá-la por meio de **URLs.**

É muito comum que a **API**s retornem seus dados no formato **.json**, portanto presamos tratar esses dados quando recebemos.