**O que é o JS?**

* O JavaScript é uma linguagem de programação de alto nível, que integra o desenvolvimento de apps e páginas web. É uma das linguagens mais usadas no mundo do desenvolvimento;
* Atualmente o JavaScript pode ser executado por diversos meios, console de navegadores, IDEs como Visual Studio Code e NetBeans, terminais, entre outros, graças a tecnologia V8 do Chrome e ao Node.js.

**Função**

* Cria scripts dinâmicos que realizam a interação de apps ou páginas web. Imagine que o HTML é a estrutura, o CSS é o design e o JavaScript é a parte interativa do desenvolvimento.

**Tipagem**

* A tipagem funciona como uma regra de uso de dados, quanto mais forte for, mais obrigatório é a declaração do tipo de dado. A tipagem em JavaScript é fraca, a declaração dos dados acontece de modo dinâmico;
* Ex: ao criar uma variável com valor entre aspas (“valor”) o JavaScript já converte o dado para o tipo string.

**Tipos Primitivos**

* As variáveis em JavaScript podem guardar tipos de dados que chamamos de tipos primitivos. Eles são: boolean, null, undefined, number, string, array, object, function.

**Declaração de Variáveis**

* Existem 3 modos de declarar as variáveis em JavaScript:

1. var – escopo global e local, pode ter seu valor alterado, se não tiver um valor inicial será tratada como null;
2. let – escopo local de bloco, pode ter seu valor alterado, se não tiver um valor inicial será tratada como null;
3. const – escopo local de bloco, somente leitura, o valor inicial é obrigatório e não pode ser alterado.

**Escopo**

* O escopo em JavaScript define a limitação e visibilidade de um bloco de código.

**-** Escopo global: quando a variável é declarada fora de qualquer bloco, sua visibilidade fica disponível em todo o código;

**-** Escopo local: quando a variável é declarada dentro de um bloco, sua visibilidade pode ficar disponível ou não.

**Regras para Criação de Variáveis**

* Inicie o nome da variável com letra, underscore ou cifrão. Não inicie com número;
* Não utilize espaços. Use camelCase, PascalCase ou snake\_case;
* Não utilize palavras reservadas;
* Declare as variáveis no topo do bloco de código.

**Atribuição**

* O sinal de igualdade “=” em JavaScript, significa atribuição.

**Comparação**

* Para fazermos uma comparação de valores em JavaScript usamos “==”.

**Comparação Idêntica**

* Para fazermos uma comparação de valores e tipos em JavaScript usamos “===”.

**Operadores Aritméticos**

* Adição (+), subtração (-), multiplicação (\*), divisão real (/), resto da divisão (%), potenciação (\*\*).

**Operadores Relacionais**

* Maior que (>), menor que (<), maior ou igual a (>=), menor ou igual a (<=) e diferente de (!=).

**Operadores Lógicos**

* E (&&) --> Considera que todos os valores sejam true;
* Ou (||) --> Considera que qualquer um dos valores seja true;
* Não (!) --> Inverte o valor de true para false ou vice-versa.

**Vetores ou Arrays**

* Arrays são um tipo de lista, ou matriz de variáveis, onde cada variável possui um índice. Os valores podem ser de vários tipos;
* O array deve ser declarado entre colchetes “[]”, e podem guardar qualquer valor dentro de seus índices, inclusive outros arrays;
* O índice só é acessado por um número inteiro, onde 0 é o primeiro índice e cada índice é separado por uma vírgula;
* Ao ser declarado, o Array traz consigo uma sério de métodos para manipulá-lo (Exemplos: forEach(), push (), pop (), etc).

**Objetos**

* Dados que possuem propriedades e valores que definem suas características. Deve ser declarado entre chaves “{}”;
* As propriedades dos objetos podem ser atribuídas as variáveis, facilitando a manipulação do objeto. Chamamos isso de desestruturação;
* Outra forma de fazer a desestruturação é utilizando chaves ao declarar a variável.

**Estruturas Condicionais**

* São instruções para realizar determinadas tarefas a partir de uma condição, seja de decisão ou repetição.

**Estruturas de Decisão**

* Podemos usar as palavras reservadas “if”, “else” e “else if” para estabelecer condições;
* Podemos usar a palavra “if” para estabelecer uma condição, “else” no caso de a condição não ser atendida e “else if” caso haja mais de uma condição;
* Podemos usar o “if” dentro de um outro “if”, chamamos isso de aninhamento de if’s ou ninho de if’s;
* Podemos também fazer uma verificação em uma única linha usando o “if” ternário;
* O “switch/case” também funciona como uma estrutura condicional.

**Estruturas de Repetição**

* Laços de repetição são estruturas condicionais que repetem uma instrução até atingir determinada condição;
* For: funciona como uma repetição de instrução até que a condição seja falsa;
* For/in: funciona como uma repetição a partir de uma propriedade;
* For/of: funciona como uma repetição a partir de um valor. O For/of não funciona com objetos, pois as propriedades variam, diferentes do índice em um array que sempre serão números inteiros;
* While: executa uma instrução “enquanto” determinada condição for verdadeira, a verificação é feita antes da execução.

**Funções**

* São blocos de comandos e instruções para a execução de determinadas tarefas. Geralmente se utiliza a palavra reservada “function” seguida de parênteses “()” e chaves “{}”;
* Os parênteses indicam que é um objeto do tipo function, e as chaves indicam que é um bloco de instrução / comando;
* As funções podem receber parâmetros em sua declaração, que servem como variáveis, onde sua atribuição pode ser feita durante a chamada da função;
* Expressões de funções: são funções atribuídas a expressões. A nomeação das funções por expressão é opcional;
* Arrow function: são funções de expressão de sintaxe curta, são sempre anônimas, ou seja, não são nomeadas. Deve ser declarada com parênteses “()”, seguido de “=>” e depois chaves “{}”.

**Comandos / Funções**

* // ou /\* \*/ **-->** Comentar linha / comentar várias linhas de uma vez;
* var / let / const nome da variável = atribuição; **-->** Criação de variáveis;
* var nome da variável = ‘x’ == x; **-->** (true) Comparação em valor;
* var nome da variável = ‘x’ === x; **-->** (false) Comparação em valor e tipo – Comparação Idêntica;
* var nome da variável = x + y; **-->** Adição entre x e y;
* var nome da variável = x – y; **-->** Subtração entre x e y;
* var nome da variável = x \* y; **-->** Multiplicação entre x e y;
* var nome da variável = x / y; **-->** Divisão entre x e y;
* var nome da variável = x % y; **-->** Resto da divisão de x por y;
* var nome da variável = x \*\* y; **-->** Potenciação de x por y;
* var nome da variável = x > y; **-->** Se x for maior que y retorna true;
* var nome da variável = x < y; **-->** Se x for menor que y retorna true;
* var nome da variável = x >= y; **-->** Se x for maior ou igual a y retorna true;
* var nome da variável = x <= y; **-->** Se x for menor ou igual a y retorna true;
* var nome da variável = true && true; **-->** (e) Para a variável retornar true, as duas condições precisam ser true;
* var nome da variável = true || false; **-->** (ou) Para a variável retornar true, uma das condições precisa ser true;
* var nome da variável = !false; **-->** [diferente / oposto (!) de false = true] Retorna o valor oposto ao digitado a direita da exclamação;
* console.log (nome da variável); **-->** Valor da variável para imprimir no console;
* console.log (typeof (nome da variável)); **-->** Tipo da variável para imprimir no console;
* let nome do array = [valor1, valor2, etc]; **-->** Declaração de um array;
* nome do array.length **-->** Esse comando representa o tamanho / comprimento de um array;
* nome do array.forEach (function (parâmetro index ou item) {console.log (parâmetro index ou item} **-->** .forEach () para repetir os índices e / ou valores de um array;
* nome do array.pop (); **-->** .pop () para remover um item no final de um array;
* nome do array.push (Novo valor); **-->** .push () para adicionar um item no final de um array;
* nome do array.shift (); **-->** .shift () para remover um item no início do array;
* nome do array.unshift (); **-->** .unshift () para adicionar um item no início do array;
* console.log (nome do array.indexOf (valor)); **-->** indexOf () para retornar o índice de um valor;
* nome do array.splice (posição inicial para começar a remover, número de posições para remover ->); **-->** .splice () para remover item (s) de um vetor pelo seu índice;
* let nome do array2 = nome do array.slice (posição inicial para retornar, posição final para retornar -1 ->) **-->** .slice () para retornar parte(s) um array já existente;
* let nome do objeto = {nome da variável1: valor1, nome da variável2: valor2, etc} **-->** Criação de um objeto | Variável =~ Propriedade;
* console.log (nome do objeto.nome da variável); **-->** Seleção da(s) propriedade(s) a ser(em) mostrada(s) no console | Método 1;
* var nome da variável = nome do objeto.nome da variável; **-->** Desvincula o NOME da variável do seu respectivo objeto | Método 2;
* var {nome da variável1, nome da variável2, etc} = nome do objeto; **-->** Seleciona as variáveis a serem mostradas | Método 3;
* console.log (nome da variável1 , nome da variável2, etc); **-->** Mostra as variáveis selecionadas no console;
* condição(ões) ? instrução 1 : instrução 2 **-->** Verificação em uma única linha;
* if (condição(ões)) { comando(s) } **-->** Criação da estrutura condicional if;
* else if (condição(ões)) { comando(s) } **-->** Criação da estrutura condicional else if;
* else { comando(s) } **-->** Criação da estrutura condicional else;
* switch (nome da variável){

case condição1:

comando(s)

break;

case condição2:

comando(s) **-->** Criação da estrutura condicional switch /case;

break;

etc...

default: // opcional

comando(s)

}

* for (let cont = valor1; cont < / <= valor2; cont++){ comando(s) } **-->** Criação da estrutura de repetição for;
* for (nome da variável in nome do array / objeto){ console.log (nome da variável);} **-->** Mostra as posições / propriedades em que estão armazenados os valores do array e do objeto;
* for (nome da variável in nome do array / objeto){ console.log (nome do array / objeto [nome da variável]);} **-->** Mostra os valores armazenados dentro do array e do objeto;
* for (nome da variável of nome do array){ console.log (nome da variável);} **-->** Mostra os valores armazenados dentro do array;
* while (condição(ões)){

comando(s)

nome da variável++; **-->** Criação da estrutura de repetição while;

}

* do{

nome da variável++;

comando(s) **-->** Criação da estrutura de repetição do while;

}while (condição(ões));

* function NomedaFunção () { atribuições } **-->** Criação de uma função;
* function NomedaFunção (Parâmetro1, Parâmetro2, etc) { atribuições } **-->** Criação de uma função com parâmetros;
* NomedaFunção (); **-->** Chamar a função pelo nome;
* var NomedaVariável = function NomedaFunção () { atribuições } **-->** Criação de uma expressão de função;
* var NomedaVariável = function () { atribuições } **-->** Criação de uma expressão de função. Obs: a nomeação da função é opcional, pois o nome da variável é atribuído a função;
* var NomedaVariável = () => { atribuições } **-->** Criação de uma arrow function.

**Links Úteis:**

**Glossário:**

<https://developer.mozilla.org/pt-BR/docs/Glossary/JavaScript>

**O que é o JavaScript:**

<https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/What_is_JavaScript>

**Repositório do Curso “Sintaxe Básica em JavaScript”:**

github.com/DianaMartine/curso-dio-sintaxe-basica-javascript

**Tipagem:**

https://danvitoriano.medium.com/tipagem-din%C3%A2mica-no-javascript-e3551a445b38

**Sintaxe e Tipos de Variáveis:**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Grammar_and_types>

**Escopos:**

<https://imasters.com.br/desenvolvimento/escopos-em-javascript>

**Expressões e Operadores:**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Expressions_and_Operators#operador_atribuicao>

**Arrays:**

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global\_Objects/Array

**Array.protoype.indexOf():**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array/indexOf>

**Trabalhando com Objetos:**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Working_with_Objects>

**Desestruturação, parâmetros rest e espalhamento:**

<https://www.digitalocean.com/community/tutorials/understanding-destructuring-rest-parameters-and-spread-syntax-in-javascript-pt>

**Estruturas Condicionais:**

<https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Building_blocks/conditionals>

**Laços de Repetição:**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Loops_and_iteration>

**Funções:**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Functions>

**Number:**

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Number>

**Window.prompt():**

<https://developer.mozilla.org/pt-BR/docs/Web/API/Window/prompt>

**Window.alert:**

<https://developer.mozilla.org/pt-BR/docs/Web/API/Window/alert>

**Template strings:**

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Template\_literals