* JavaScript
* O que é?

A linguagem JavaScript é uma linguagem de programação considerada dinâmica ( Por permitir que uma variável seja atribuido por um valor de um tipo e por permitir que o tipo dessa variável seja alterado por uma outra atribuição) e interpretada.   
A característica interpetrada se refere a uma linguagem de programação em que o código fonte dessa linguagem é executado por um programa de computador, chamado interpretador, que é executado pelo sistema operacional ou processador. Ele lê o código linha por linha em vez de ler ele todo de uma só vez.

O JavaScript executa diretamente no navegador. E ele surgiu como o navegador Netscape. E segue as especificações ECMAScript.   
  
O JavaScript pode ser adcionado de duas maneiras na página. Uma por meio de um script externo, que é um link no HTML direcionando para o documento JavaScript. O elemento Script não é um elemento vazio, então ele precisa de uma tag de abertura e uma tag de fechamento.   
Ex:

<Script scr=”./js/meu-script.js”> </Script>  
  
  
E a outra por meio do Script Inline, onde o conteúdo do Script fica dentro da tag <Script> não usando o atributo SRC.  
  
<Script>

Console.log (Este é um Script)

</Script>

Obs: As boas práticas indicam que é melhor usar um script externo. Pois o tamanho desse script pode ser muito grande para ficar dentro do html.

O JavaScript é executado na sequencia que ele é definido no HTML. Então o navegador processa o HTML, como resposta da requisição e sempre que há um elemento Script, ele vai executar esse elemento Script e só depois que ele executar o Script que ele vai continuar processando o documento.

* Sintaxe Básica de um Script  
    
  No JavaScript eu começo fazendo a declaração de uma variável atráves da palavra chave *Var* , seguida de um nome para essa variável, uso o operador = e atribuo um valor entre ‘’.   
  O uso do ponto e vírgula (;) no final da minha variável é opicional. Porém é importante seguir um padrão na escrita do código.   
    
  Var *mensagem =* ‘Olá’ ;

É possível atribuir a variável para um outro valor:

mensagem= ‘Olá Danilo”;   
  
Para imprimir essa váriavel é necessário usar uma função:   
  
 Alert (mensagem);

É assim que fica um documento JavaScript:   
  
 Var *mensagem =* ‘Olá’ ;  
mensagem= ‘Olá Danilo”;  
  
 Alert (mensagem);

* Tipos primitivos   
    
  São três tipos principais que são usados com frequência:   
    
  🡪 Var vNumber , que aceita tanto números inteiros como números decimais.   
    
  EX: Var vNumber= 5   
    
  🡪 Var String, é definido por qualquer sequência de carácteres. Elas podem ser escritas tanto entre aspas simples ‘’, como entre aspas duplas “”. Mas é importante manter uma consistência na escrita do código.   
    
    
  🡪 Var vBoolean, recebe apenas dois tipos de valores, o True ou false.   
    
  Obs: Dentro do Console, no navegador, é possível checar que tipo de variável eu estou trabalhando. Basta eu colocar (typeof e o nome da minha variável) e ele vai me trazer que tipo de variável eu estou trabalhando.
* Objetos – Em JavaScript, um objeto é uma entidade independente, com propriedades e tipos ue definem suas características. Elas são escritas entre chaves {}. Todos os dados que são colocados dentro do meu objeto podem ser alterados ou apagados por novas variáveis.   
  Quando eu solicito ao console uma variável que não está completa ou que foi deletada, ele me responde com a palavra “Undefined”.   
  Como em outros programas, é possível usar o valor Null em JavaScript. Ele representa um valor nulo ou vazio para aquela variável. Porém não confundir com o Undefined. Pois eles não são a mesma coisa.   
    
  Obs: Diferença entre Null e Undefined – Segundo a documentação ECMAScript, o valor undefined é utilizado quando uma variável não possui valor. Já o Null é utilizado quando se quer intencionalmente dizer que há uma ausência de qualquer valor de objeto para aquela variável.   
    
  🡪Array – Um Array no JavaScript é um objeto com um contrutor único, com uma sintaxe literal e com um conjunto adcional de propriedades e de metódos herdados de um protótipo de array. Eles são usados para armazenar vários valores em uma única variável, e são escritos entre colchetes [].   
    
  Obs: Uma diferença importante entre tipos primitivos e objetos no JavaScript é que os Objetos primitivos copiam o seu valor. Já os objetos tem o valor como uma referência.  
    
  Ex:   
  Var primitivo = 1;  
  Var primitivo2 = primitivo1 // copia o valor   
  primitivo2 =2   
    
  console.log(primitivo1); // Imprime 1 como valor   
  console.log(primitivo2); //Imprime 2 como valor   
    
    
  var objeto1 = {x: 1};  
  var objeto2 = objeto1 //copia a referência  
  objeto2.x = 2   
    
  console.log(objeto1.x); //Imprime 2 como valor   
  console.log(objeto2.x); //Imprime 2 como valor
* Operadores e expressões no JavaScript   
    
  🡪Operadores Lógicos – São usados para realizar operações lógicas. Elas podem ser do tipo Negação (!), tipo E lógico (&&) e OU lógico (||). Os operandos devem ser lógicos, verdadeiro ou falso. Também podem operar sobre expressões lógicas, ou seja, que retornem valores verdadeiros ou falso.   
    
  E lógico (&&) - O Operador E lógico recebe dois operandos e retorna verdadeiro se, e somente se ambos os operandos sejam verdadeiros . Retorna falso, caso contrário.   
    
  Ou lógico (||) – Para o operador Ou lógico (||) retornar verdadeiro, basta que um dos operandos seja verdadeiro. Ele também retorna verdadeiro caso os dois operando sejam verdadeiros. Retorna falso, se os dois forem falsos.   
    
  Negação (!) – O operador de negação (!) é um operando unário, isto é, opera sobre apenas um operando. Ele nega, inverte o valor lógico do operando.   
    
    
  🡪 Operadores Aritiméticos – São operadores usados nas contas aritiméticas da matemática:   
    
    
  X+Y - Adição   
  X-Y - Subtração   
  +X – Converte para número (Se já não for)  
  -X – Inversão de sinal  
  X/Y – Divisão   
  X\*Y – Multiplicação  
  X%Y – Resto da divisão de X por Y  
  X \*\* Y – Exponenciação  
    
  OBS: O operador Aritimético adição também pode ser usado para somar Strings. Chamado Concatenação. O resultado que ele produz pode ser visto como uma soma. Este operador irá avaliar os valores dos operandos, que podem ser mais de dois e retornar uma string que é a junção desses valores. Vale notar que os operandos não precisam necessariamente ser todos do tipo String.   
    
  EX: Var numitens = 10;   
    
  var mensagem = ‘O número de itens no pedido é’ + numitens + ‘.’  
    
  O Valor da variável ‘mensagem’ será: O número de itens no pedido é 10.  
    
    
  Os operadores podem ser escritos com atribuições:   
    
  X + = y - Adição   
  X - = y – Subtração  
  X / = y – Divisão   
  X \* = y – Multiplicação  
  X % = y – Resto da divisão de x por y   
  X \*\* = y – Exponenciação   
    
  Essa forma de escrever pode ser entendida como uma abreviação:   
    
  EX: X=5  
  X + = 2;   
  X = 7;   
    
  É o equivalente a X= X+2   
    
    
  🡪 Operadores de Incremento e Decremento -   
    
  X ++ - Retorna X e depois incrementa   
  X - - Retorna X e depois decrementa   
  ++X – Incrementa e depois retorna X  
  --X - Decrementa e depois retorna X  
    
  EX: Var X = 1;   
  X++   
  X = 1  
  X= 2   
    
  - Nesse primeiro exemplo, ele primeiro trouxe o valor original de X e depois aumentou, mudando o valor de X.   
    
  ++X   
  X= 3   
    
  -Nesse ele aumenta o valor de X antes de trazer o valor da variável.   
    
    
  🡪 Operadores de Comparação:   
    
  X < Y - Menor   
  X > Y – Maior   
  X <= Y – Menor ou igual   
  X >= y – Maior ou igual   
    
    
    
  🡪 Operadores de igualdade:   
    
  X==y – Igual   
  X != y - Diferente  
  X === y – Exatamente Igual   
  X ! == y - Não Extamente igual   
    
    
  🡪 Precedência de operadores – A precedência de operadores determina a ordem em que os operadores são processados. Operadores com maior precedência são processados primeiro