# O que é JavaScript

Programação de alto nível.

Serve para criar scripts dinâmicos que realizam interações de apps ou páginas web.

# Entendendo variáveis e seus valores

## Tipagem

Funciona como uma regra de uso de dados, quanto mais forte a tipagem mais obrigatório é a declaração do tipo de dado.

Tipagem em JS é fraca e a declaração acontece de forma dinâmica.

Ex.: Ao criar uma variável, o JS já converte em string.

## Tipos primitivos

|  |  |
| --- | --- |
| **String** | Cadeia de caracteres (servirá como texto); |
| **Number** | Números, ficam azul no console; |
| **Null** | Nada/Vazio; |
| **Boolean** | Verdadeiro ou Falso; |
| **Undefined** | O computador não conseguiu encontrar um valor. |

Var – escopo global e local, pode ter o valor alteado. Se não tiver um valor inicial será tratado como null.

Let – escopo local de bloco, pode ter o valor alterado. Se não tiver um valor inicial será tratado como null.

Const – escopo local de bloco, comente leitura, o valor inicial é obrigatório e não pode ser alterado.

**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.

|  |  |
| --- | --- |
| **==** | Igual a; (compara valores, retornando um boolean)  Ex.: 3 número e “3” letra são iguais com o ==. |
| **===** | Valor igual e igual. (compara o valor e o tipo);  Ex.: 3 número e “3” letra. São diferentes com o ===. |

## Operadores relacionais

|  |  |  |  |
| --- | --- | --- | --- |
| > | < | >= | <= |

## Operadores lógicos

|  |  |  |
| --- | --- | --- |
| && - “e” | || - “ou” | ! – “não” |
|  |  |  |

# Vetores e Objetos

## Vetores ou Arrays

São um tipo de lista ou matriz de variáveis, onde cada variável possui um índice.

## Manipulando Arrays

|  |  |
| --- | --- |
| forEach () | Executa uma dada função em cada elemento de um array.  Ex.: array.forEach (function (item, index) {console.log (item, index) } ) |
| Push () | Adiciona itens no final do array; |
| Pop () | Remove item no final do array; |
| Shift () | Remove item no início do array; |
| Unshift () | Adiciona item no início do array; |
| indexOf () | Retorna o índice de um valor; |
| Splice () | Remove ou substitui um item pelo índice; |
| Slice () | Retorna uma parte de um array existente. |

## Objetos

Dados que possuem propriedades e valores que definem suas características. Devem ser declarados “{ }”.

## Manipulando objetos

Propriedade de obetos podem ser atribuídas à variáveis, facilitando a manipulação do objeto. Chamamos isso de desestruturação.

Var xicara = { cor: ‘azul’, tamanho: ‘p’, função: tomarCafe() }

Var cor = xicara.cor

Var tamanho = xicara.tamanho

Var função = tomarCafe()

Outra forma de fazer a desestruturação é utilizando chaves ao declarar a variável.

Var xicara = { cor: ‘azul’, tamanho: ‘p’, função: tomarCafe() }

Var {cor, tamanho, função} = xicara

# Estruturas condicionais

São instruções para realizar determinadas tarefas a partir de uma condição.

## If ternário

Quando faz uma verificação em uma única linha.

A<18 ? console.log (‘Menor de idade.” : console.log (‘Maior de idade.’)

## Switch/case

Funciona como uma estrutura condicional também.

|  |  |
| --- | --- |
|  | Ex.:  Switch (${expressão} {  Case placar>1:  Console.log (‘ texto1’)  Break  Case placar<1:  Console.log (‘ texto2’)  Break  } |

## Laços de repetição

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.  Ex.: For (i in lista) { } |
| For/of | Funciona como uma repetição a partir de um valor.  Ex.: for (i of lista) { } |
| While | Executa uma instrução ‘enquanto’ determinada condição for verdadeira. |
| Do/While | Executa uma instrução “até que” determinada condição seja falsa, a verificação é feita depois da execução.  Do {  A++  Console.log(a)  } while (a<10) |

# Funções e suas particularidades

## Funções

São blocos de comandos e instruções para a execução de determinada tarefa.

## Funções com parâmetros

As funções podem receber em sua declaração, parâmetros, que servem como variáveis, onde sua atribuição pode ser feita durante a chamada da função.

## Expressões de funções

São funções atribuídas à expressões. A nomeação das funções por expressão é opcional.

Var função = function ( ) {

Console.log (“Sou uma mensagem de expressão”) }

Função ( )

## Arrow function

São funções de expressão de sintaxe curta. Sempre serão anônimas e portanto não podem ser nomeadas. Deve ser declarada comparênteses “( )”, seguido de “=>” e depois chaves “{ }”.

Var função = ( ) => {

Console.log(“Sou uma arrow function”) }

Função ()