**JavaScript**

O que vamos ver no curso?

* Tipos de dados
* Variáveis
* Funções
* Condicionais
* Estruturas de repetição

Data types

* Primitive / Primitive value
* Structural
* Structural Primitiva

**PRIMITIVOS**

* String
* Number
* Boolean
* Undefined
* Symbol
* BigInt

**ESTRUTURAIS**

* Object
  + Array
  + Map
  + Set
  + Date
  + ...
* Function

PRIMITIVO ESTRUTURAL / STRUCTURAL ROOT PRIMITIVE

* null

**SCOPE – ESCOPO**

O scope determina a visibilidade de alguma variável no JS.

//var é global e local, poderá funcionar fora de um escopo de bloco

//hoisting - elevação

 console.log('> existe X antes do bloco?', x)

{

    var x = 0

}

console.log('> existe X depois do bloco?', x)

//const e let são locais e só funcionam no escopo onde foram criadas

let y = 1;

{

    let y = 0

    console.log('> existe Y!', y)

}

console.log('> existe Y depois do bloco?', y)

//const e let são locais e só funcionam no escopo onde foram criadas

const y = 1;

{

    const y = 0

    console.log('> existe Y!', y)

}

console.log('> existe Y depois do bloco?', y)

//No caso da const não dá para declarar um valor isolado, igual o que fizemos no let, é necessário uma declaração em cada scope para que o JS entenda

/\*

## Para criar nomes das variáveis

\*JS é case-sensitive (sensível a maísculas/minúsculas)

\*JS aceita a cadeia de caracteres Unicod

- Posso:

    \* Iniciar com esses caracteres especiais: $ \_

    \* Iniciar com letras

    \* Colocar acentos

    \* Letras maiúsculas e minúsculas fazem diferença

- Não posso:

    \* Iniciar com números

    \* Colocar espaços vazios no nome

- Ideal:

    \* Criar nomes que fazem sentido

    \* Nomes que expliquem o que a variável é ou faz

    \* camelCase

    \* snake\_case

    \* Escrever em Inglês

\*/

**FUNCTIONS – FUNÇÕES**