Monitoria FuP DD prof PH

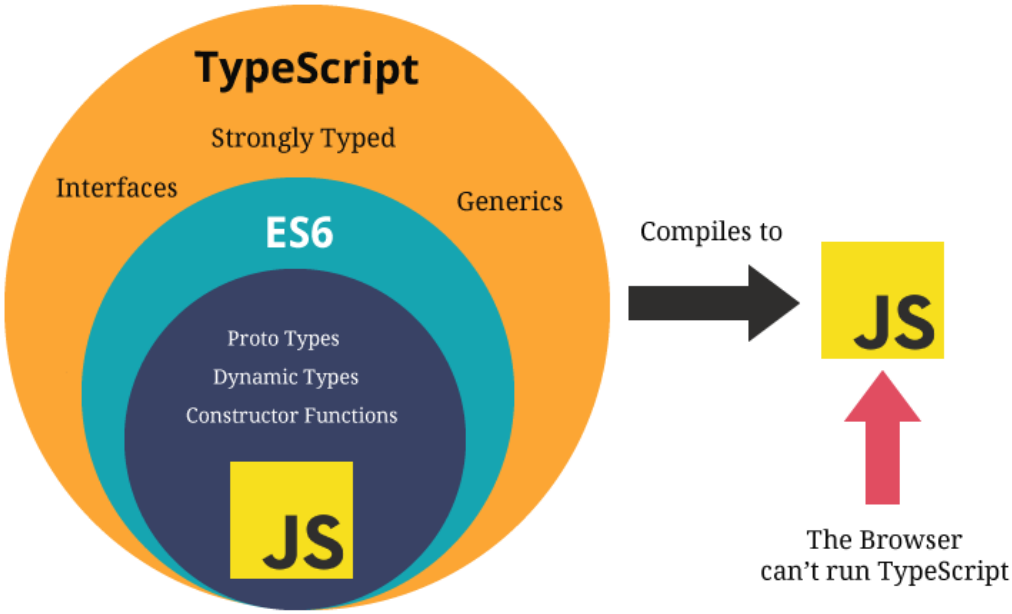
by: Henricky Lima

Requisitos

- Visual Studio Code
- NodeJS

Conteúdo

JavaScript



- **Linguagem de programação**

Uma linguagem de programação é um método padronizado, formado por um conjunto de regras sintáticas e semânticas, de **implementação** de um **código fonte** - que pode ser **compilado** e transformado em um **programa de computador**, ou usado como **script interpretado** - que informará **instruções** de processamento ao computador.

- **Utilizada:** FrontEnd Web (sites)
- **Também utilizada:**
  - BackEnd com Node.js (banco de dados, dinâmismo e interatividade)
  - Aplicações Desktop (Electron)
  - Aplicações Mobile (React Native)
- É uma das principais liguagens atualmente
- Em web temos 3 ramos principais, e por trás deles as seguintes linguagens:
  - **Conteúdo:** HTML
  - **Estilo ou Aparência:** CSS
  - **Interatividade:** JavaScript

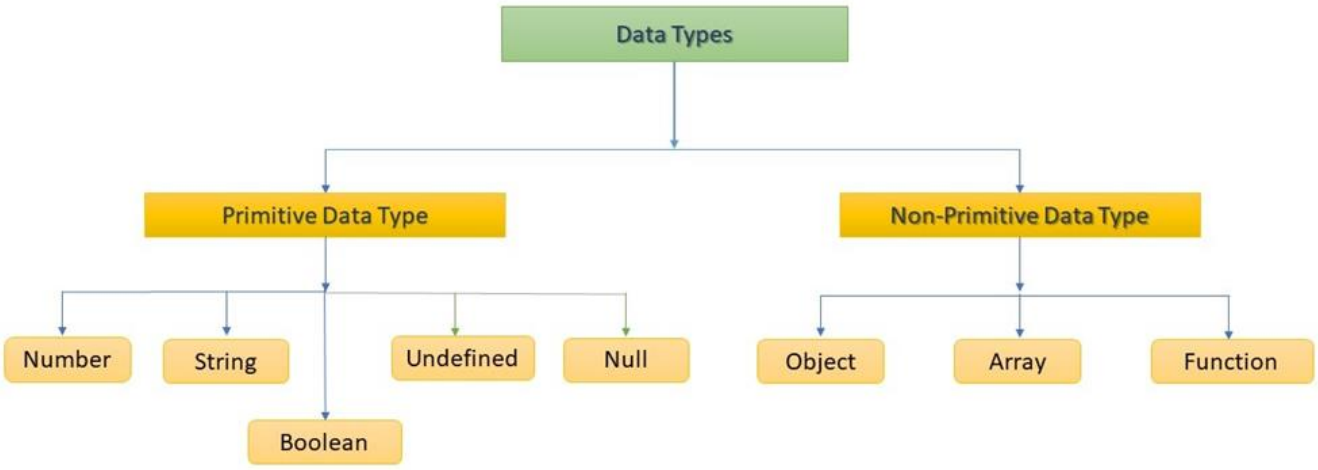
Algoritmo

Conjunto de passos **Finitos** e **Organizados** que, quando executados, resolvem um determinado **Problema**.

Pseudo-Código

```
VARIAVEIS//declaração de variaveis
n1, n2, resultado: Integer;
INICIO
    //atribuição de valores
    n1 := 0;
    n2 := 0;
    resultado := 0;
    // saída de dados (mostrar na tela)
    ESCREVA("Digite o primeiro número:");
    LER(n1); //entrada de dados
    ESCREVA("Digite o segundo número:");
    LER(n2);
    //atribuição
    resultado := n1+n2;
    ESCREVA("A soma dos dois números é:",resultado);
FIM
```

Tipos de Dados



Tipo de Dados	Exemplos
Number	1, -3, 8.56
String	"3,14", "Frases", "Palavras", 'c'
Boolean	true, false
Undefined	undefined(ausência de valor)
Null	null (explicito)
Symbol	-- não abordado
Object	[1,2,5], [6, "sete", true], {id:2, nome: "João"}
function	*

Conversão de Dados

Variáveis podem ser convertidas, ou seja, transformar de um tipo para outro. Normalmente se pensa em converter um tipo Number para String ou vice-versas.

Number:

Podemos converter o tipo **String** para **Number** utilizando as funções:

- **Number(X)**: Converte X para Number.

```
/*  Pode Retornar
 Numero Inteiro
 Numero Real
 NaN     - Not a Number
 Infinity - Infinito
 ---------------------------*/
 Number("5")         5
 Number("8.75")      8.75
 Number("Azul")      NaN
```

- **Number.parseInt(X)**: Converte X para um numero inteiro.

```
/* Retorna
 Number.Int
 NaN
 ------------*/
 Number.parseInt("42")      42
 Number.parseInt("7.15")    7
 Number.parseInt(1.75)      1.75
```

- **Number.parseFloat(X)**: Converte X para um número real.

```
/* Retorna
 Number.Float
 NaN
 Infinity
 ------------------------*/
Number.parseFloat("5")      5.0
Number.parseFloat("2.7")    2.7
Number.parseFloat("Lemore") NaN
Number.parseFloat(5/0)      Infinity
```

String

- **String(X)**: Converte X para texto.
- **X.toString()**: Converte X para string

```
X = 17.5
String(X)           "17.5"
X.toString(X)       "17.5"
```

Concatenação:

Concatenação é a operação de juntar duas strings, ou, juntar duas palavras.

```
/* Concatenação:
  String + String   = String
  String + Number   = String
  String + Boolean  = String
*/
//ex1
A = "Azul"
B = "Marinho"
A+B             "AzulMarinho"
//ex2
Nome    = "Luiz"
Idade   = 15
Nota    = 7.5
Txt     = Nome + " tem " + idade + " anos e tirou " +Nota+ " na média!"
//resultado
"Luiz tem 15 anos e tirou 7.5 na média!"
```

Formatando String - ${}:

Visando facilitar a vida de quem está aprendendo a programar, existe esta técnica chamada **Template Strings** que permite a concatenação de strings de uma forma mais robusta.

```
// TemplateString - entre crases utilize ${variável}
Txt = `${Nome} tem ${idade} anos e tirou ${Nota} na média! `
//concatenação antiga
Txt = Nome + " tem " + idade + " anos e tirou " +Nota+ " na média!"
```

Alguns Métodos da variável String

JavaScript trabalha com objetos, ou seja, elementos que carregam atributos e métodos padrões - Este conceito está relacionado com o conceito de **Orientação a Objetos**. Estas variáveis(atributos) e Funções(métodos), que um objeto trás podem ser acessadas utilizando ponto(**.**). Alguns exemplos desta funcionalidade:

```
S = "Ninho de Mafagafos"
//Obter o tamanho de uma String
S.length            18                      // atributo - Sem parenteses
//Tudo Maiusculo
S.toUpperCase()     "NINHO DE MAFAGAFOS"    //Método - com parenteses
//tudo Minusculo
S.toLowerCase()     "ninho de mafagafos"

//dividir uma string apartir de um separador
data = "12/05/2020"
data.split('/')     [ '12', '05', '2020' ]
```

Operadores

Operadores Aritmeticos

Operador	Atribuição	Soma	Subtração	Multiplicação	Divisão	Exponencial	Resto da Divisão	incremento	Decremento	Atribuição com operação
Simbolo	=	+	-	*	/	**	%	++	--	+= -= *= /= %=

```
//Operadores Artitmeticos
var res = x + y                     //x mais y
var res = x - y                     //x menos y
var res = x * y                     //x vezes y
var res = x / y                     //x dividido por y
var res = x % y                     //x resto da divisão por y
```

Operadores Relacionais

Operador	Igualdade forte	Igualdade fraca	Diferença	Maior	Menor	Maior ou igual	Menor ou igual
Simbolo	===	==	!=	>	<	>=	<= < td>

```
//Operadores de comparação/Relacionais
// x == y                     x igual y  (Erro Comum: x=y)
// x != y                     x não igual y
// x > y                      x Maior que y
// x < y                      x Menor que y
// x >= y                     x Maior ou igual y
// x <= y                     x Menor ou igual y
```

Operadores Lógicos			
Operador	**E**	**Ou**	**Não**
Simbolo	&&	||	!

```
//Operadores de combinação e negação -  Valores lógicos P e Q
// P && Q                      P e Q
// P || Q                      P uu Q
// ! P                         Não P
```

Lógica

- **Tabela verdade**

Truth Table of Logical Operators				
In C++ boolean *true* is 1 and *false* is 0				
a	**b**	**a && b**	**a || b**	**! a**
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Comandos JS

Comentário:

```
//isto é um comentário
var l = "Isto é uma comando"
/*
Isto é
     um comentário
                em bloco!
*/
```

Entrada e saída:

```
//(Output) Console
console.log("Hello world!") // printa na tala

//(Output) Document
document.write("Hello World!")
alert("Olá mundo!") //na forma de box
```

```
//(input) Document
var nome = prompt("Digite seu nome:") //na forma de box
```

Condicional:

- Condição simples

```
// If / Se
if( condição1 ){
        //Bloco de código
}else if( condição2){
        //bloco de código
}else{
        //bloco de código
}
//exemplo
if( x > y ){
    console.log("x é maior")
}else if( x < y ){
    console.log("y é maior")
}else{
    console.log("São iguais")
}
```

- Switch

```
//switch case
switch(variavel){
        case comparação1: //variavel == comparação1
                //bloco de codigo
                break;
        case comparação2:
                //bloco de código
                break;
        default: //caso contrário
                //bloco de código
}
```

* Operador Ternário

```
//Ternário
variable = condition ? op_if_yes : op_if_not
//exemplo
var a=5,b=7;
var res = (a>b):"maior": "menor";
```

Repetição

- Enquanto (while)

```
//while
while(condição){
    //Bloco de código
}
//exemplo
var n = 0, x = 0
while (n < 3) {
    n++ //n = n+1
    x += n
}
```

- for_statement

Um laço for é repetido até que a condição especificada seja falsa

```
// for_statement padrão
for([atribição] ; [condição] ; [incremento]){
        //bloco de código
}
//exemplo - andar 5 passos para sul
for(let i=0;i<5; i++){
    cosole.log("passo ao sul!")
}
```

- for_in

A declaração for...in executa iterações a partir de uma variável específica, percorrendo todas as propriedades de um objeto.

```
//for_in
for(index in object){
    //bloco de código
}
//exemplo
var arr = [3, 5, 7];
for (let i in arr) {
    console.log(i +" - "+ arr[i]) // 0 - 3, 1 - 5, 3 - 7
}
```

- for_of

A declaração for...of cria uma laço com objetos interativos executando uma iteração para o valor de cada objeto.

```
// for_of
for(valor_objeto of objeto){
    //bloco de código
}
//exemplo
var Nomes = ["caio", "yusuke", "Henricky", "Pedro", "Alfredo"]
for( let nome of Nomes ){
    console.log( nome + " tem " + nome.length + " letras")
}
```

Listas, Vetores e Objetos:

- **Vetores:**

Vetores são objetos que armazenam uma quantidade finita de valores do mesmo tipo. Eles tendem a ter um tamanho **n** e cada elemento é referenciado por um indice **i**.

```
//vetor
var v = [10, 15, 6, 8] // length: 4
//v[i]
v[0]            10
v[1]            15
v[3]            8
```

- **Lista:**

Lista são, basicamente, vetores que armazenam valores de tipo diferentes. Assim como vetor elas tem indice e tamanho.

```
//lista
var list  = ["baleia", 42, "toalha", true, false]
```

- **Objeto:**

```
var pessoa = {nome: "Luiz", idade: 23, cidade:"Quixadá", casado:true}
```

Em Construção...

Referência

- Laços e iterações