introducao-javascript

September 2, 2023

1 Introdução JavaScript

1.1 Comentários

[2]:
```
// este é um comentário de uma linha
```

[3]:
```
/* este é um comentário
de várias linhas */
```

1.2 Grámatica JavaScript

[50]:
```
// podemos escrever cada instrução em uma linha separada
const mensagem = 'Olá pessoal'
console.log(mensagem)
```

```
Olá pessoal
```

[7]:
```
// ou podemos escrever várias instruções em uma única linha
const mensagem = 'Olá pessoal'; console.log(mensagem);
```

```
Olá pessoal
```

[9]:
```
// é boa prática de programação terminar cada instrução com ;
// mesmo estando sozinho em uma linha
const mensagem = 'Olá pessoal';
console.log(mensagem);
```

```
Olá pessoal
```

[15]:
```
{
    // isto é um blodo de instruções
    const mensagem = 'Boa noite!';
    console.log(mensagem);
}
```

```
Boa noite!
```

1.3 Tipos de dados

[16]:
```
typeof 'mouse'
```

```
string
```

[17]:
```
typeof 10
```

```
number
```

[18]:
```
typeof 9.87
```

```
number
```

[19]:
```
typeof true
```

```
boolean
```

[21]:
```
typeof [1, 2, 3]
```

```
object
```

[24]:
```
typeof { cidade: 'São Luís', estado: 'MA' }
```

```
object
```

1.4 Variáveis

[26]:
```
const nome = 'Luiz Carlos'; // declara uma 'constante' e atribui um valor a ela
console.log(nome)
```

```
Luiz Carlos
```

[28]:
```
// não podemos atribuir um novo valor a uma constante
nome = 'outro nome'
```

```
2:1 - Cannot assign to 'nome' because it is a constant.
```

[31]:
```
// let cria uma variável cujo valor pode ser alterado
let idade = 50;
console.log(idade)
```

```
50
```

[30]:
```
idade = 51;
```

```
51
```

1.5 Escopo de variáveis

1.5.1 Escopo global

Variáveis definidas no escopo global podem ser acessadas em qualquer lugar

[ ]:
```
const cidade = 'São Luís'
```

[33]:
```
console.log(cidade)
```

```
São Luís
```

1.5.2 Escopo local

Variáveis que são definidas no escopo local só podem ser acessadas no contexto específico onde foram declaradas

[39]:
```
const estado = 'Maranhão'; // escopo global
{
    const sigla = 'MA'; // escopo local
    console.log(estado);
    console.log(sigla);
}
console.log(estado);
//console.log(sigla);  // remova o comentário e esta linha gerará um erro
```

```
Maranhão
MA
Maranhão
```

1.6 Nomeando variáveis

[42]:
```
// nomes de variáveis devem começar com uma letra ou $ ou _, podem conter␣
 ↪letras, números e $ e _.
let primeiroNome = 'Paulo';
let _nomeDoMeio = 'Costa';
let $ultimo_nome = 'Silva';
let ano2000 = 2000;
```

[46]:
```
exemplos de nomes inválidos
let #primeiroNome = 'Paulo';
let 2000anos = 2000;
let minha&variável = true;
```

```
4:10 - ',' expected.
4:11 - Cannot find name 'variável'.
4:20 - ';' expected.
```

[51]:
```
// notação camel case
let nomeCompleto = 'Luiz Muniz';  // é mais comum para javascript
// notação snake case
let nome_completo = 'João Souza';
```

1.7 Atribuição por valor e por referência

[59]:
```
// atrivuição por valor
let a = 10;
let b = a;  // b = 10
b = 11;
console.log(a);
console.log(b);
```

```
10
11
```

[80]:
```
// atribuição por referência
let estado = { sigla: 'MA' }
let outroEstado = estado;  // as duas variáveis estão apontando pra mesma sigla
console.log(estado.sigla)
console.log(outroEstado.sigla)
outroEstado.sigla = 'RJ'   // isto afeta a variável estado
console.log(outroEstado.sigla)
console.log(estado.sigla)
```

```
MA
MA
RJ
RJ
```

1.8 Propriedades e métodos de strings

[81]:
```
let nome = 'Luiz';
console.log(nome.length)
```

```
4
```

[82]:
```
console.log(nome.toUpperCase())
```

```
LUIZ
```

[83]:
```
console.log(nome.toLowerCase())
```

```
luiz
```

[88]:
```
console.log(nome.charAt(2))
console.log(nome.charAt(100))
console.log(nome.indexOf('z'))
console.log(nome.indexOf('p'))
```

```
i

3
-1
```

[103]:
```
let frase = 'Batatinha quando nasce esparrama pelo chão';
console.log('tamanho da frase: ' + frase.length)
console.log('índice do primeiro a: ' + frase.indexOf('a'))
console.log('índice do último a: ' + frase.lastIndexOf('a'))
console.log('começa com B: ' + frase.startsWith('B'))
console.log('repete: ' + nome.repeat(3))
```

```
tamanho da frase: 42
índice do primeiro a: 1
índice do último a: 31
começa com B: true
repete: LuizLuizLuiz
```

1.9 Templates literals

[109]:
```
let frase = 'Meu nome é ' + nome + '. Minha idade é ' + idade + '.';
console.log(frase);

let frase2 = `Meu nome é ${nome}. Minha idade é ${idade}.`;  // template literal
console.log(frase2);
```

```
Meu nome é Luiz. Minha idade é 50.
Meu nome é Luiz. Minha idade é 50.
```

1.10 Números

[126]:
```
let a = 10;
let b = 5.442343;

let aEhInteiro = Number.isInteger(a);
console.log(aEhInteiro)

let bEhInteiro = Number.isInteger(b);
console.log(bEhInteiro)
```

```
true
false
```

[134]:
```
// chamando métodos em números

// console.log(5.toFixed(2)) // isto gera um erro

console.log(5..toFixed(2))
console.log( (5).toFixed(3) )
console.log(3.14234324.toFixed(3))  // se for real pode  usar apenas um ponto␣
 ↪antes do nome do método
let numero = 10
console.log(numero.toFixed())
```

```
5.00
5.000
3.142
10
```

1.11 Operadores de incremento

[158]:
```
let x = 100;
console.log('x = ' + x)
x = x + 1;
console.log('x = ' + x)
x++;   //  x = x + 1
console.log('x = ' + x)
x += 1;
console.log('x = ' + x)

x = x + 2;
x += 2   // x = x + 2

let y = 100;
console.log('y = ' + y)
y = y - 1;
console.log('y = ' + y)
y--;   //  x = x + 1
console.log('y = ' + y)
y -= 1;
console.log('y = ' + y)

y = y - 2;
y -= 2   // x = x + 2

let z = 10
z *= 3   // z = z * 3
console.log('z = ' + z)
z /= 2   // z = z / 2
console.log('z = ' + z)
```

```
x = 100
x = 101
x = 102
x = 103
y = 100
y = 99
y = 98
y = 97
z = 30
z = 15
```

1.12 Conversão de tipos de dados

[160]:
```
'100' + 10
```

```
10010
```

[161]:
```
Number('100') + 10
```

```
110
```

[164]:
```
Number('adasd') + 10  // NaN é Not a number
```

```
NaN
```

[166]:
```
typeof String(5)
```

```
string
```

[168]:
```
typeof 10..toString()
```

```
string
```

[174]:
```
// converte entre bases numéricas
console.log('decimal = ' + 10..toString())
console.log('decimal = ' + 10..toString(10))
console.log('binário = ' + 10..toString(2))
console.log('octal = ' + 10..toString(8))
console.log('hexadecimal = ' + 10..toString(16))
```

```
decimal = 10
decimal = 10
binário = 1010
octal = 12
hexadecimal = a
```

[195]:
```
// converte string em número decimal
console.log('decimal = ' + parseInt('100'))
```

```
console.log('decimal = ' + parseInt('100', 10))
console.log('binário = ' + parseInt('1010', 2))
console.log('binário = ' + parseInt('11', 8))
console.log('binário = ' + parseInt('F', 16))
```

```
decimal = 100
decimal = 100
binário = 10
binário = 9
binário = 15
```

1.13 Operadores lógicos

[199]:
```
let a = 10;
let b = 20;

console.log(a == b)
console.log(a != b)
console.log(a > b)
console.log(a < b)
console.log(a >= b)
console.log(a <= b)
```

```
false
true
false
true
false
true
```

[213]:
```
let a = 10;
let b = 20;
let c = 30
// E lógico => &&
console.log(a > b && b > c)
console.log(a < b && b < c) // b está entre a e c ?
// OU lógico => ||
console.log(a > b || b > c)
console.log(a < b || b < c) // b está entre a e c ?
// negação
console.log('-------')

console.log( !(a > b)  )

console.log('-------')
let x = true
console.log('x = ' + x)
```

```
console.log('não x = ' + !x)
```

```
false
true
false
true
-------
true
-------
x = true
não x = false
```