JavaScript

Variáveis

Declaração de Variáveis

var, let e const

var => Declara uma variável, inicializando ou não seu valor

let => Declara uma variável em escopo de bloco, inicializando ou não seu valor

const => Declara uma constante em escopo de bloco

Declaração de Variáveis

Conceito de Variable Hoisting

```
console.log(x === undefined); // true
var x = 3;

(function () {
  console.log(x); // undefined
  var x = "local value";
})();
```

```
var x;
console.log(x === undefined); // true
x = 3;

(function () {
  var x;
  console.log(x); // undefined
  x = "local value";
})();
```

Declaração de Variáveis

Conceito de Variable Hoisting

```
console.log(x); // ReferenceError
const x = 3;

console.log(y); // ReferenceError
let y = 3;
```

Declaração de Variáveis

Const - previne redesignação, não mutação

```
const MY_OBJECT = { key: "value" };
MY_OBJECT.key = "otherValue";
```

```
const MY_ARRAY = ["HTML", "CSS"];
MY_ARRAY.push("JAVASCRIPT");
console.log(MY_ARRAY); // ['HTML', 'CSS', 'JAVASCRIPT'];
```

Tipos de Dados

Boolean => true e false

null => valor nulo

undefined => Uma variável cujo valor ainda não foi definido

Number => Número inteiro ou de ponto flutuante (vírgula). Ex.: 30 ou 33.4

BigInt => Um número inteiro com precisão arbitrária. Ex.: 83979834759843n

String => Uma sequência de caracteres que representa um texto. Ex.: “JavaScript"

Symbol => Um tipo de dado cujas instâncias são únicas e imutáveis

Object => Coleções de propriedades

Conversão de Dados

Tipagem Dinâmica

```
let answer = 42;
```

```
answer = "Thanks for all the fish!";
```

Conversão de Dados

Operador +

```
x = "The answer is " + 42; // "The answer is 42"
y = 42 + " is the answer"; // "42 is the answer"
z = "37" + 7; // "377"
```

```
"37" - 7; // 30
"37" * 7; // 259
```

Conversão de Dados

parseInt( )

```
parseInt(string)
parseInt(string, radix)
```

```
console.log(parseInt('123'));
// 123 (default base-10)
console.log(parseInt('123', 10));
// 123 (explicitly specify base-10)
console.log(parseInt('   123 '));
// 123 (whitespace is ignored)
console.log(parseInt('077'));
// 77 (leading zeros are ignored)
console.log(parseInt('1.9'));
// 1 (decimal part is truncated)
console.log(parseInt('ff', 16));
// 255 (lower-case hexadecimal)
console.log(parseInt('0xFF', 16));
// 255 (upper-case hexadecimal with "0x" prefix)
console.log(parseInt('xyz'));
// NaN (input can't be converted to an integer)
```

Conversão de Dados

parseFloat( )

```
parseFloat(string)
```

```
function circumference(r) {
  return parseFloat(r) * 2.0 * Math.PI;
}

console.log(circumference(4.567));
// Expected output: 28.695307297889173

console.log(circumference('4.567abcdefgh'));
// Expected output: 28.695307297889173

console.log(circumference('abcdefgh'));
// Expected output: NaN
```

Arrays

```
const coffees = ["French Roast", "Colombian", "Kona"];
```

```
const fruits = [];
fruits.push("banana", "apple", "peach");
console.log(fruits.length); // 3
```

```
fruits.length = 10;
console.log(fruits); // ['banana', 'apple', 'peach', empty x 2, 'mango', empty x 4]
console.log(Object.keys(fruits)); // ['0', '1', '2', '5']
console.log(fruits.length); // 10
console.log(fruits[8]); // undefined
```

```
fruits.length = 2;
console.log(Object.keys(fruits)); // ['0', '1']
console.log(fruits.length); // 2
```

Objects

```
const sales = "Toyota";

function carTypes(name) {
  return name === "Honda" ? name : `Sorry, we don't sell ${name}.`;
}

const car = { myCar: "Saturn", getCar: carTypes("Honda"), special: sales };

console.log(car.myCar); // Saturn
console.log(car.getCar); // Honda
console.log(car.special); // Toyota
```

```
const car = { manyCars: { a: "Saab", b: "Jeep" }, 7: "Mazda" };

console.log(car.manyCars.b); // Jeep
console.log(car[7]); // Mazda
```