OBJETOS
(AULA 7)

CURSO BÁSICO DE PROGRAMAÇÃO COM JAVASCRIPT

MAYARA MARQUES

mmrosatab@gmail.com

SUMÁRIO

- Operações em objetos
  - Spread Operator
  - Spread Operator and change value
  - Destructuring
  - Destructuring rename
  - Destructuring with default value
  - Shorthand Properties
  - Object.keys
  - Object.values
  - Object.entries
  - Object.assign
- Mão na massa

OPERAÇÕES COM OBJETOS

O Javascript apresenta uma série de operações úteis para se trabalhar com objetos. No (ECMAScript 6) ES6 foram introduzidas vários novos recursos que facilitam e agilizam a programação, tornam o código mais limpo e expressivo.

ECMAScript é a especificação padronizada da linguagem de programação que serve como base para o **JavaScript**, **JScript** (da Microsoft) e **ActionScript** (do Flash). Ele define a sintaxe, recursos e funcionalidades essenciais da linguagem, garantindo que diferentes implementações sigam um mesmo conjunto de regras.

Saiba mais sobre ECMAScript 6 em: https://www.w3schools.com/js/js_es6.asp

SPREAD OPERATOR

O operador spread (...) é usado para **expandir elementos de arrays ou propriedades de objetos**.

```
const objetoOriginal = { a: 1, b: 2 }
const novoObjeto = { ...objetoOriginal, c: 3 }

console.log(novoObjeto) // Saída: { a: 1, b: 2, c: 3 }
```

SPREAD OPERATOR AND CHANGE VALUE

O Spread Operator também pode ser usado para **criar cópias de arrays ou objetos** e, em seguida, **modificar valores específicos**.

```
const original = { nome: 'Alice', idade: 30 }
const modificado = { ...original, idade: 31 }

console.log(modificado) // Saída: { nome: 'Alice', idade: 31 }
```

DESTRUCTURING

A atribuição via ***desestruturação*** permite extrair valores de arrays ou propriedades de objetos e atribuí-los a variáveis.

```
const pessoa = { nome: 'Bob', idade: 25 }
const { nome, idade } = pessoa

console.log(nome, idade) // Saída: Bob 25
```

DESTRUCTURING AND RENAME

Você pode renomear variáveis ao usar a desestruturação em objetos.

```
const pessoa = { nome: 'Carol', idade: 28 }
const { nome: nomeDaPessoa, idade: idadeDaPessoa } = pessoa

console.log(nomeDaPessoa, idadeDaPessoa)  // Saída: Carol 28
```

DESTRUCTURING WITH DEFAULT VALUE

Você pode fornecer valores padrão para as variáveis durante a desestruturação.

```
const pessoa = { nome: 'Daniel' }
const { nome, idade = 30 } = pessoa

console.log(nome, idade) // Saída: Daniel 30
```

SHORTHAND PROPERTIES

O shorthand de propriedades permite criar objetos de forma mais concisa, utilizando o mesmo nome para a variável e a propriedade.

```
const nome = 'Eva'
const idade = 22

const pessoa = { nome, idade }

console.log(pessoa) // Saída: { nome: 'Eva', idade: 22 }
```

OBJECT.KEYS

O método Object.keys retorna um array com todas as **chaves** do objeto.

```
const usuario = { nome: "Ana", idade: 25, cidade: "São Paulo"
}
console.log(Object.keys(usuario))
// Saída: ["nome", "idade", "cidade"]
```

OBJECT.VALUES

O método Object.values retorna um array com todos os **valores** do objeto.

```
const usuario = { nome: "Ana", idade: 25, cidade: "São Paulo"
}
console.log(Object.values(usuario))
// Saída: ["Ana", 25, "São Paulo"]
```

OBJECT.ENTRIES

O método Object.entries retorna um array de arrays, onde cada sub-array contém um par [chave, valor].

```
const usuario = { nome: "Ana", idade: 25, cidade: "São Paulo" }
console.log(Object.entries(usuario))
// Saída: [["nome", "Ana"], ["idade", 25], ["cidade", "São Paulo"]]
```

OBJECT.ASSIGN

O método Object.assign copia as propriedades de um ou mais objetos para um objeto de destino.

```
// sintaxe

Object.assign(objDestino, obj1, obj2, ...)
```

OBJECT.ASSIGN

Se passarmos um objeto vazio {} como o primeiro argumento, ele cria um novo objeto e copia as propriedades dos demais objetos para ele.

```
const obj1 = { a: 1, b: 2 };
const obj2 = { c: 3, d: 4 };
const resultado = Object.assign({}, obj1, obj2);
console.log(resultado);
// Saída: { a: 1, b: 2, c: 3, d: 4 }
```

OBJECT.ASSIGN

Se usarmos um objeto existente como o primeiro argumento, ele será modificado diretamente.

```
const destino = { x: 10 }
const origem = { a: 1, b: 2 }

Object.assign(destino, origem)

console.log(destino) // { x: 10, a: 1, b: 2 }
```

MÃO NA MASSA

MÃO NA MASSA

1. Crie uma função que receba dois objetos e retorne um novo objeto com todas as propriedades dos dois objetos usando o operador spread.
2. Crie um objeto representando uma pessoa. Em seguida, crie uma função que modifique a idade dessa pessoa usando o operador spread.
3. Crie um objeto representando um livro (com propriedades como título, autor, etc.). Em seguida, crie uma função que receba o objeto e imprima suas propriedades usando a desestruturação.
4. Crie uma função que receba um objeto com propriedades "largura" e "altura", e renomeie essas propriedades para "w" e "h" usando a desestruturação.
5. Crie uma função que receba um objeto representando uma pessoa, com propriedades "nome" e "idade". Se a propriedade "idade" não estiver presente, defina um valor padrão de 25.
6. Crie uma função que receba duas variáveis, "fruta" e "quantidade", e retorne um objeto representando uma cesta de frutas usando o shorthand de propriedades.
7. Crie uma função que receba um objeto representando um carro com propriedades como "modelo" e "ano". Modifique o objeto usando a desestruturação e o operador spread para alterar o modelo e adicionar uma nova propriedade "cor".
8. Cria uma função que receba um objeto como parâmetro e retorne quantas chaves ele possui.
9. Cria uma função que receba um objeto como parâmetro e imprime todos os seus valores associados as chaves desse objeto.

REFERÊNCIAS

- https://www.w3schools.com/js/js_objects.asp
- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Working_with_objects