
Facens

Programação para Web JavaScript

Prof. Especilista Fábio Colombini

Facens

Opção de testar JavaScript

- https://scrimba.com/

Facens

Funções Anônimas

- Funções anônimas em JavaScript são aquelas que não possuem um nome identificador. Em vez disso, são definidas diretamente onde são necessárias, muitas vezes como argumentos para outras funções ou atribuídas a variáveis. Aqui estão alguns pontos importantes sobre funções anônimas:

```
let soma = function(a, b) {
    return a + b;
};

console.log(soma(2, 3)); // 5
```

Facens

Arrow Functions

- Com a introdução das arrow functions (funções de flecha) no ECMAScript 6, uma forma mais concisa de escrever funções anônimas foi introduzida.

```
let soma = (a, b) => a + b;

console.log(soma(2, 3)); // 5
```

Facens

Bigint

- Todos os números JavaScript são armazenados em formato de ponto flutuante de 64 bits.

- JavaScript BigInt é um novo tipo de dados (ES2020) que pode ser usado para armazenar valores inteiros que são grandes demais para serem representados por um número JavaScript normal.

- `let x = BigInt("12345678901234567890123456789");`

Facens

Array

- Array (Matriz) JavaScript são escritas entre colchetes.
- Os itens da matriz são separados por vírgulas.
- O código a seguir declara (cria) um array chamado cars, contendo três itens (nomes de carros):

- `const cars = ["Saab", "Volvo", "BMW"];`

- O operador typeof em JavaScript retorna “Object” para arrays
- Arrays usam números para acessar seus “elementos”.
  - Neste exemplo, cars[0]

Facens

Nested Array

```
var cars = [["Saab", 5], [ "Volvo",2], ["BMW",4]];

Cars[0]  // [“Saab”, 5]
cars[0][0] // Saab
cars[0][1] // 5
```

Facens

Array

- Você pode ter variáveis de tipos diferentes dentro do Array

```
let myArray = ["Fabio", 5, true, ["Ferrari", "Porsche", "Lamborghini"]];
```

- Como retornar os elementos desse Array?

Facens

Iterando com Array

```
fruits = ["Banana", "Orange", "Apple", "Mango"];

for (let i = 0; i < fruits.length; i++) {
    console.log(fruits[i]);
}

for (let item of fruits) {
    console.log(item);
}
```

Facens

Editando Array

- Adicionando element no Array
- final: fruits.push("Lemon");
- início: unshift("Lemon")

- Removendo Elemento
- pop(): remove last element
- shift(): remove first element

- Demo:
- `let myArray = ["Fabio", 5, true, ["Ferrari", "Porsche", "Lamborghini"]];`
  - Adicionar um item
  - Adicionar um item nos “carros” (array dentro de array)
  - Adicionar no Inicio
  - Remover Item no final
  - Remover item no final de array dentro de array

Facens

Como reconhecer um Array

- **Array**.isArray(fruits); // returns true
- fruits **instanceof** Array; // returns true

Facens

Principais Propriedade e Métodos

- length
- sort()
- toString()
- join() joins all array elements into a string.
- pop() removes the last element from an array
- push() adds a new element to an array (at the end)
- shift() removes the first array element and "shifts" all other elements to a lower index
- unshift() adds a new element to an array (at the beginning), and "unshifts" older elements
- delete fruits[0]; Changes the first element in fruits to undefined

Facens

Principais Propriedade e Métodos

- splice() can be used to add new items to an array
  - fruits.splice(2, 0, "Lemon", "Kiwi");
    - (2) defines the position where new elements should be added (spliced in)
    - (0) defines how many elements should be removed
    - ("Lemon" , "Kiwi") define the new elements to be added
  - can remove elements without leaving "holes" in the array
  - fruits.splice(0, 1); // Removes the first element of fruits
- concat() creates a new array by merging (concatenating) existing arrays
  - var myChildren = myGirls.concat(myBoys);
- slice() slices out a piece of an array into a new array
  - var fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];
  - var citrus = fruits.slice(1,2);

Facens

Objects

- Objetos JavaScript são escritos com chaves {}.
- As propriedades do objeto são escritas como pares nome:valor, separados por vírgulas.

```
// Object:
const person = {firstName:"John", lastName:"Doe"};
```

Facens

Acesando Propriedades do Object

```
var person = {
    "firstName":"John",
    "lastName":"Doe",
    "age":50,
    "eyeColor":"blue",
};
```

- objectName.propertyName
- objectName["propertyName"]
  - quando usa espaco no "nome"
  - “Last Name":"Doe"

Facens

Iterando as propriedades do Objeto

```
const pessoa = {
    "nome": "Fabio",
    "idade": 50,
    "isProfessor": true
}

for (let prop in pessoa) {
    console.log(prop +" "+pessoa[prop]);
}
```

Facens

Método no Objeto

```
var person = {
    "firstName":"John",
    "lastName":"Doe",
    "age":50,
    "eyeColor":"blue",

    fullName : function() {
        return this.firstName + " " + this.lastName;
    }
};
```

- this
  - Em uma definição de função, this refere-se ao “proprietário” da função.

Facens

Array dentro de um Objeto

```
const pessoa = {
    "nome": "Fabio",
    "idade": 50,
    "isProfessor": true,
    "Carros": ["Ferrari", "Porsche"],

    print: function() {
        return `Nome: ${this.nome} Idade: ${this.idade};`
    }
}
```

Facens

Conversão de Tipos de dados

- **Para número inteiro (integer):**
  - parseInt()

```
let stringNumero = "10";
let numero = parseInt(stringNumero);
```

- **Para número decimal (float):**
  - parseFloat()

```
let stringNumero = "10.5";
let numero = parseFloat(stringNumero);
```

Facens

Conversão de Tipos de dados

- Para string:
  - toString()

```
let numero = 10;
let stringNumero = numero.toString();
```

- JavaScript tentará converter strings em números em todas as operações numéricas:

```
// Funciona
let x = "100";
let y = "10";
let z = x / y;
```

Facens