📘 OBJETOS EM JAVASCRIPT

🔹 O que é um objeto?

Em JavaScript, **um objeto é uma variável especial** que pode armazenar **vários valores ao mesmo tempo**.

Esses valores ficam organizados em **pares chave–valor**:

- **Chave** → nome da propriedade
- **Valor** → informação guardada

📌 Pense em um objeto como uma **ficha** que descreve algo do mundo real (uma pessoa, um carro, um animal, etc.).

🔹 Exemplo do objeto Carro

Um carro pode ter **propriedades** e **métodos**.

🚗 Propriedades (características):

- nome → Fiat
- modelo → 500
- peso → 850kg
- cor → branco

⚙️ Métodos (ações):

- ligar()
- dirigir()
- frear()
- parar()

➡️ **Carros diferentes** podem ter as **mesmas propriedades**, mas com **valores diferentes**.
➡️ **Os métodos são os mesmos**, mas usados em momentos diferentes.

🔹 Criando um objeto em JavaScript (Objeto Literal)

A forma mais comum e recomendada é o **objeto literal**, usando `{ }`.

```
const car = {
  type: "Fiat",
  model: "500",
  color: "white"
};
```

📌 **Importante:**

- Use sempre `const` para declarar objetos.
- Isso **não torna o objeto imutável**, apenas impede que ele seja reatribuído.
- As **propriedades ainda podem ser alteradas**.

🔹 Criando objetos de várias formas

✔️ Objeto literal (RECOMENDADO)

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

✔️ Objeto em várias linhas (mais legível)

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

✔️ Objeto vazio + adicionando propriedades

```
const person = {};

person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";
```

❌ Usando `new Object()` (não recomendado)

```
const person = new Object({
  firstName: "John",
  lastName: "Doe",
  age: 50,
  eyeColor: "blue"
});
```

📌 **Todos fazem a mesma coisa**, mas o **objeto literal é melhor** por ser:

- Mais simples
- Mais rápido
- Mais fácil de ler

🔹 Acessando propriedades de um objeto

Existem duas formas:

🔸 Notação de ponto

```
person.lastName;
```

🔸 Notação de colchetes

```
person["lastName"];
```

🔸 Usando variáveis

```
let x = "firstName";
person[x];
```

🔹 Métodos de Objetos

Métodos são **funções dentro de objetos**.

```
const person = {
  firstName: "John",
  lastName: "Doe",
  id: 5566,
  fullName: function () {
    return this.firstName + " " + this.lastName;
  }
};
```

📌 O que é `this`?

- `this` se refere ao **próprio objeto**
- `this.firstName` significa → a propriedade `firstName` do objeto `person`

🔹 Chamando métodos

```
person.fullName(); // Executa a função
```

⚠️ Se você fizer:

```
person.fullName;
```

➡️ Ele retorna a **função**, não o resultado.

🔹 Adicionando métodos a um objeto

```
person.name = function () {
  return this.firstName + " " + this.lastName;
};
```

Usando métodos JavaScript dentro do objeto

```
person.name = function () {
  return (this.firstName + " " + this.lastName).toUpperCase();
};
```

🔹 Funções Construtoras de Objetos

Usamos quando precisamos criar **muitos objetos do mesmo tipo**.

📌 O nome da função começa com **letra maiúscula**.

```
function Person(first, last, age, eye) {
  this.firstName = first;
  this.lastName = last;
  this.age = age;
  this.eyeColor = eye;
}
```

Criando objetos com o construtor

```
const myFather = new Person("John", "Doe", 50, "blue");
const myMother = new Person("Sally", "Rally", 48, "green");
```

📌 O `this` passa a representar **o novo objeto criado**.

🔹 Adicionando e removendo propriedades

➕ Adicionar

```
person.nationality = "English";
```

➖ Remover

```
delete person.age;
```

ou

```
delete person["age"];
```

📌 Após excluir, a propriedade **deixa de existir**.

🔹 Objetos Aninhados (Objeto dentro de objeto)

```
const myObj = {
  name: "John",
  age: 30,
  myCars: {
    car1: "Ford",
    car2: "BMW",
    car3: "Fiat"
  }
};
```

Acessando:

```
myObj.myCars.car2;
myObj["myCars"]["car2"];
```

🔹 Como exibir objetos JavaScript

Exibir um objeto direto mostra:

```
[object Object]
```

✔️ Exibir propriedades manualmente

```
let text = person.name + ", " + person.age + ", " + person.city;
```

✔️ Usando `for...in`

```
let text = "";
for (let x in person) {
  text += person[x] + " ";
}
```

✔️ Usando `Object.values()`

```
let text = Object.values(person).toString();
```

✔️ Usando `Object.entries()`

```
for (let [key, value] of Object.entries(person)) {
  console.log(key + ": " + value);
}
```

✔️ Usando `JSON.stringify()` (mais usado)

```
let text = JSON.stringify(person);
```

📌 Transforma o objeto em **texto no formato JSON**.

🔹 Resumo Final 📌

- Objetos armazenam **propriedades e métodos**
- Propriedades são **valores**
- Métodos são **funções**
- Objetos representam coisas reais
- Quase tudo em JavaScript é objeto
- Entender objetos = entender JavaScript