

# OBJETOS, CLASSES E ATRIBUTOS



DEVinHouse

Parcerias para desenvolver a sua carreira

**SENAI**

<LAB365>

# AGENDA

- Programação orientada a objetos (POO)
- Objeto
- Atributos
- Classes

**Programação orientada a objetos (POO, ou OOP)** é um paradigma (modelo) de programação baseado na construção de estruturas que agrupam **características** e/ou **comportamentos**. Esses agrupamentos são chamados de **objetos**.

Uma **classe** é um tipo de objeto.

As características de um objeto são chamadas de **propriedades, atributos** ou **campos**. Os comportamentos de um objeto são chamados de **métodos**.



# OBJETOS

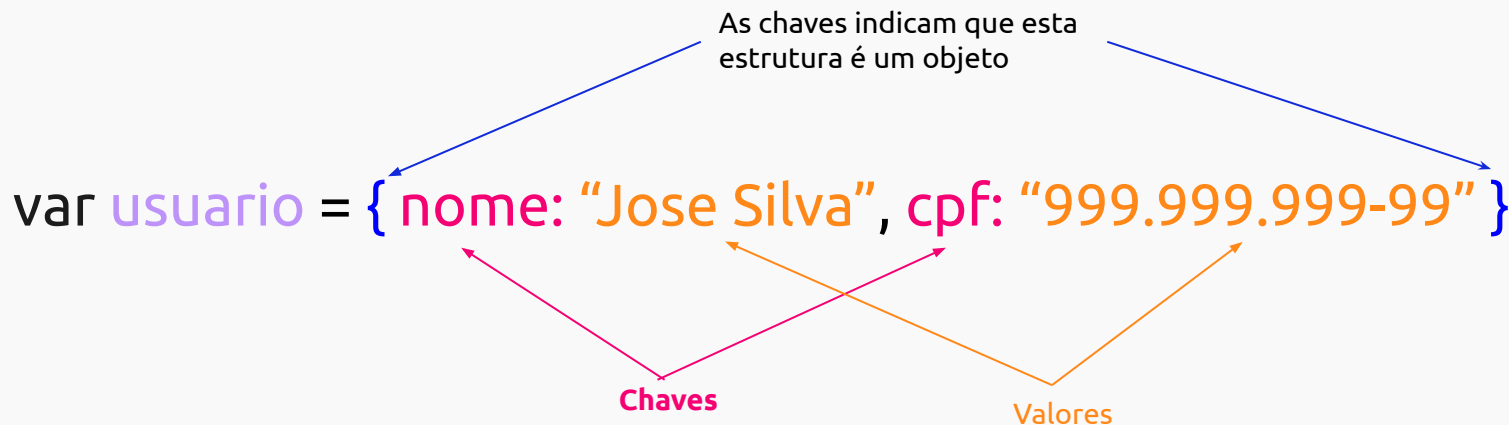
Objetos, diferentemente dos arrays, armazenam os valores em pares de “chave: valor”.

Usamos objetos para agrupar coisas com características similares ou que fazem parte de um mesmo contexto.

O valor pode ser qualquer tipo de dado (inclusive outros objetos).

# OBJETOS

Sintaxe de um objeto JavaScript:



Alguns métodos úteis de objetos:

- objeto.**keys**( ): retorna um **array** com os nomes das chaves (propriedades) do objeto
- array.**entries** ( ): retorna um array de arrays, sendo que cada array interno é formado pelos pares chave e valor.

[Veja muitos outros métodos neste link!](#)

# PROPRIEDADES DE UM OBJETO

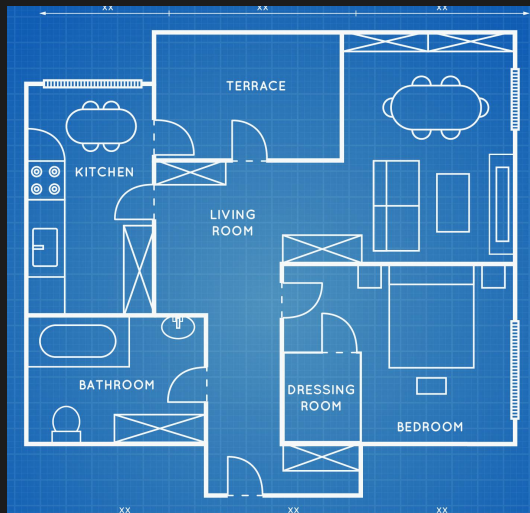
No corpo do objeto podemos acessar uma chave ou propriedade, como também é conhecida, através do ponto (.).

Se eu tenho um objeto filme com as propriedades nome, anoDeLancamento e diretor e desejo saber qual o nome do filme basta eu declarar o nome do objeto seguido de . e sua propriedade.

Exemplo:

```
let filme = {  
  nome: "Doutor Estranho no Multiverso  
da Loucura",  
  anoDeLancamento: 2022,  
  diretor: "Sam Raimi"  
}  
  
console.log(filme.nome);
```

Exemplo dinâmico:



**casa.cozinha**

# EXERCÍCIO

Imaginando que um gabinete de um computador tente passar suas características para um objeto com o mesmo nome.

Exemplo: cor, marca, modelo PlacaMae...





# INTERVALO DE AULA

## DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

**Início:** 20:40

**Retorno:** 21:00



# CLASSES & OO

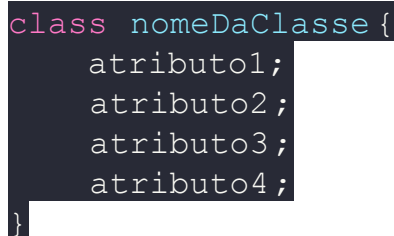
- Mas e se, por exemplo, você tiver uma loja de veículos? Vai ter que criar manualmente cada um dos objetos para o seu sistema?
- Para facilitar a criação de objetos com características e comportamentos similares, utilizamos as **classes**.
- A classe é uma espécie de “planta” de um objeto. Ela estabelece os atributos e métodos padrão a todos os objetos que foram criados a partir dela.



# CLASSES

- Assim como em diversas linguagens orientadas a objetos, o javascript possui a declaração de classes com as mesmas definições, conceito, atributos e métodos padrões.
- Uma classe é uma representação de algo que possui um conjunto de características ou regras ligadas a ela.

Declaração padrão:



```
class nomeDaClasse {  
  atributo1;  
  atributo2;  
  atributo3;  
  atributo4;  
}
```

# MÉTODOS DE PRIMITIVOS

Uma classe pode representar qualquer coisa que possua um grupo de atributos ou ações:

## Veículo



veículo = "carro";  
modelo = "fusca";  
cor = "vermelho";

buzinar () { ... }  
acelerar () { ... }

## Pensamento



tipo = "idéia";  
qualidade = "ruim";  
autor = "Michael";

curtir () { ... }  
esquecer () { ... }

## Veículo



veículo: "carro"  
modelo: "fusca"  
cor: "vermelho"

buzinar () { ... }  
acelerar () { ... }

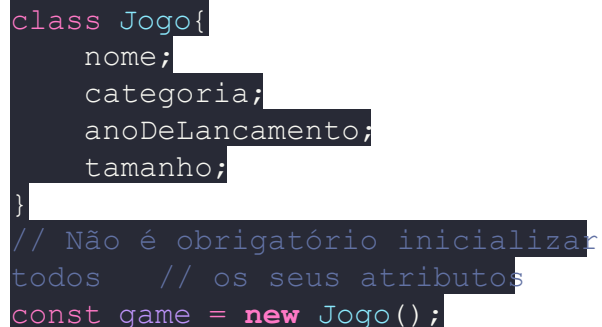


```
class Veiculo{  
    tipo = "carro";  
    modelo = "fusca";  
    cor = "vermelho";  
  
    buzinar(){}  
    acelerar(){}  
  
}
```

# CLASSES & OO

Para criar um objeto, utilizamos a palavra reservada **new**, seguida do nome da classe e de parênteses, de forma similar à execução de funções. E podemos atribuir esse objeto à uma variável:

```
const umCarro = new Veiculo ( )
```



```
class Jogo{  
    nome;  
    categoria;  
    anoDeLancamento;  
    tamanho;  
}  
  
// Não é obrigatório inicializar  
todos // os seus atributos  
const game = new Jogo();
```

# CLASSES & OO - ESTADO E COMPORTAMENTO

Chamamos de **estado** os valores que atribuímos aos **atributos** do objeto:

```
const umCarro = new Veiculo ( )
```

```
umCarro.veiculo = "carro"
```

```
umCarro.modelo = "fusca"
```

```
umCarro.cor = "vermelho"
```

## Veículo



veículo: "carro"  
modelo: "fusca"  
cor: "vermelho"

buzinar () { ... }  
acelerar () { ... }

# CLASSES & OO - ESTADO E COMPORTAMENTO

Chamamos de **comportamento** os **métodos** adicionados às classes:

```
const umCarro = new Veiculo ( )
```

```
umCarro.buzinar() // Bi.
```

```
umCarro.acelerar() // Vrum!
```

## Veículo



veículo: "carro"  
modelo: "fusca"  
cor: "vermelho"

buzinar () { ... }  
acelerar () { ... }



# CLASSES & OO - ESTADO E COMPORTAMENTO

Dentro de uma classe, existe um método especial chamado **constructor**( ).

Este método é executado imediatamente após chamar a classe com o new, e recebe os argumentos da criação da instância:

Exemplo:

```
class Pessoa{  
    nome;  
    idade;  
    cpf;  
  
    constructor(nome, idade, cpf){  
        this.nome = nome;  
        this.idade = idade;  
        this.cpf = cpf;  
    }  
}
```

*const professor = new Pessoa("Rayane", 22, "1234567")*

O objeto **professor** é uma instância da classe **Pessoa** sendo "Rayane", 22 e "1234567" como valores iniciais desta classe.

# CLASSES & OO - ESTADO E COMPORTAMENTO

Exemplo:

```
class Pessoa{  
    nome;  
    idade;  
    cpf;  
  
    constructor(nome, idade, cpf){  
        this.nome = nome;  
        this.idade = idade;  
        this.cpf = cpf;  
    }  
}
```

*const professor = new Pessoa("Rayane", 22, "1234567")*

OBS: O uso da palavra reservada **this**. é para apontar para o escopo geral da classe.

# EXERCÍCIO

Em equipe criem uma classe com constructor iniciando todos os atributos da classe e métodos referentes a um determinado objeto.

Exemplo:

```
class carro{  
    modelo;  
    ano;  
    cor;  
  
    constructor(modelo, ano, cor){  
        this.modelo = modelo;  
        this.ano = ano;  
        this.cor = cor;  
    }  
  
    dirigir(){}  
    parar(){}  
    frear(){}  
    lavar(){}  
}
```

## AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

[Clique aqui](#) ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.





# DEVinHouse

Parcerias para desenvolver a sua carreira

**OBRIGADO!**



<LAB365>