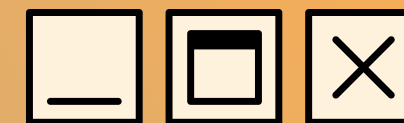


**hora da revisão!**



# Conceitos Importantes

**PROF. IURI NASCIMENTO SANTOS**

**TDS 24-1**

# Classe

```
class Carro{  
  rodas: number  
  motor: number  
  cor: string  
  modelo: string  
  marca: string  
  km: number  
  
  constructor(rodas: number, motor: number,  
    this.rodas = rodas  
    this.motor = motor  
    this.cor = cor  
    this.modelo = modelo  
    this.marca = marca  
    this.km = km  
}
```

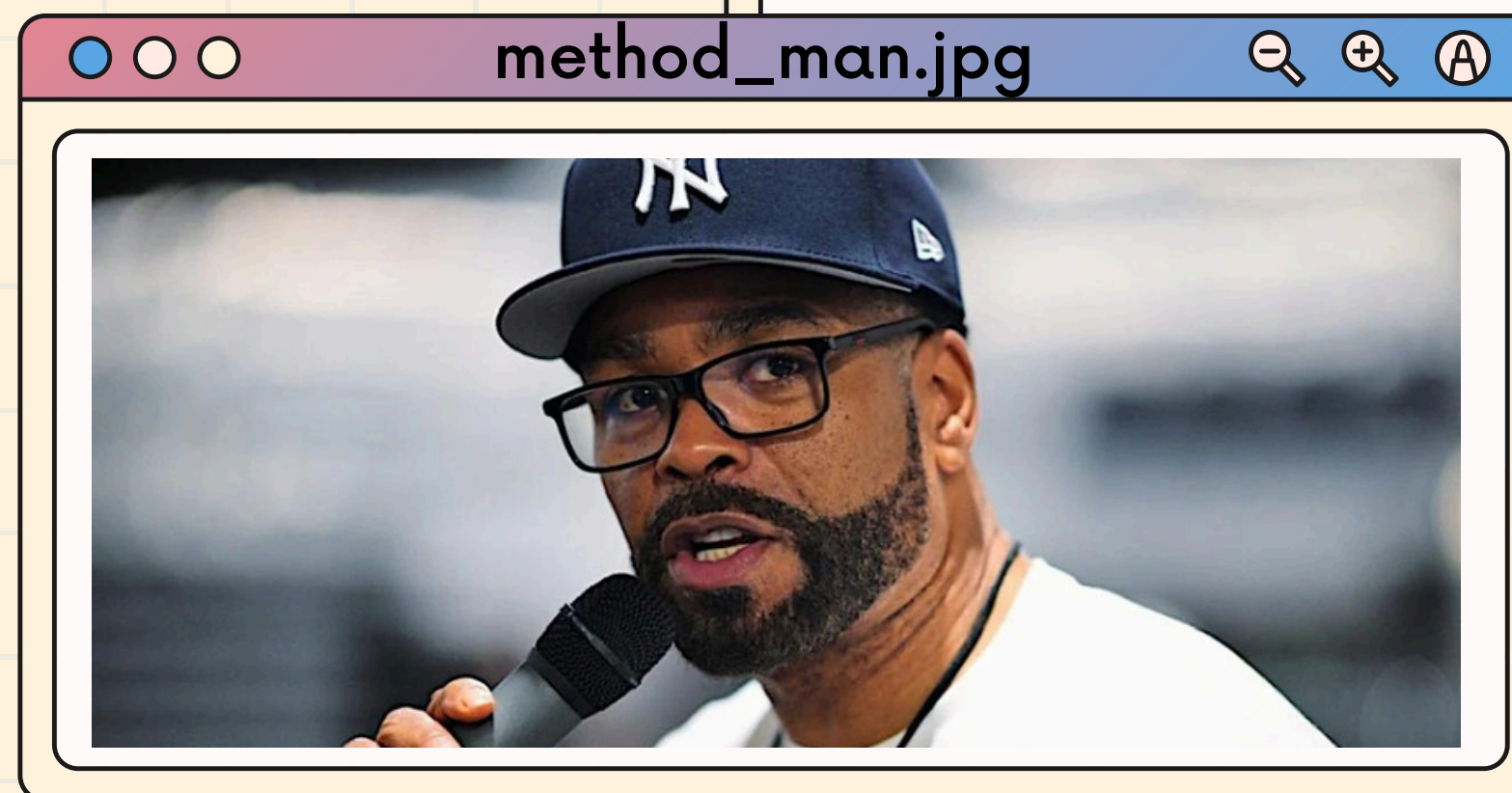
Modelo a ser seguido.  
Ela encapsula atributos  
e métodos que todos os  
objetos possuirão.



# Método

```
acelerar(){  
  console.log(`0 ${this.modelo} chegou a ${this.km} km/h`)  
}  
  
dirigir(){  
  console.log(`Estou dirigindo meu ${this.marca} ${this.modelo}`);  
}
```

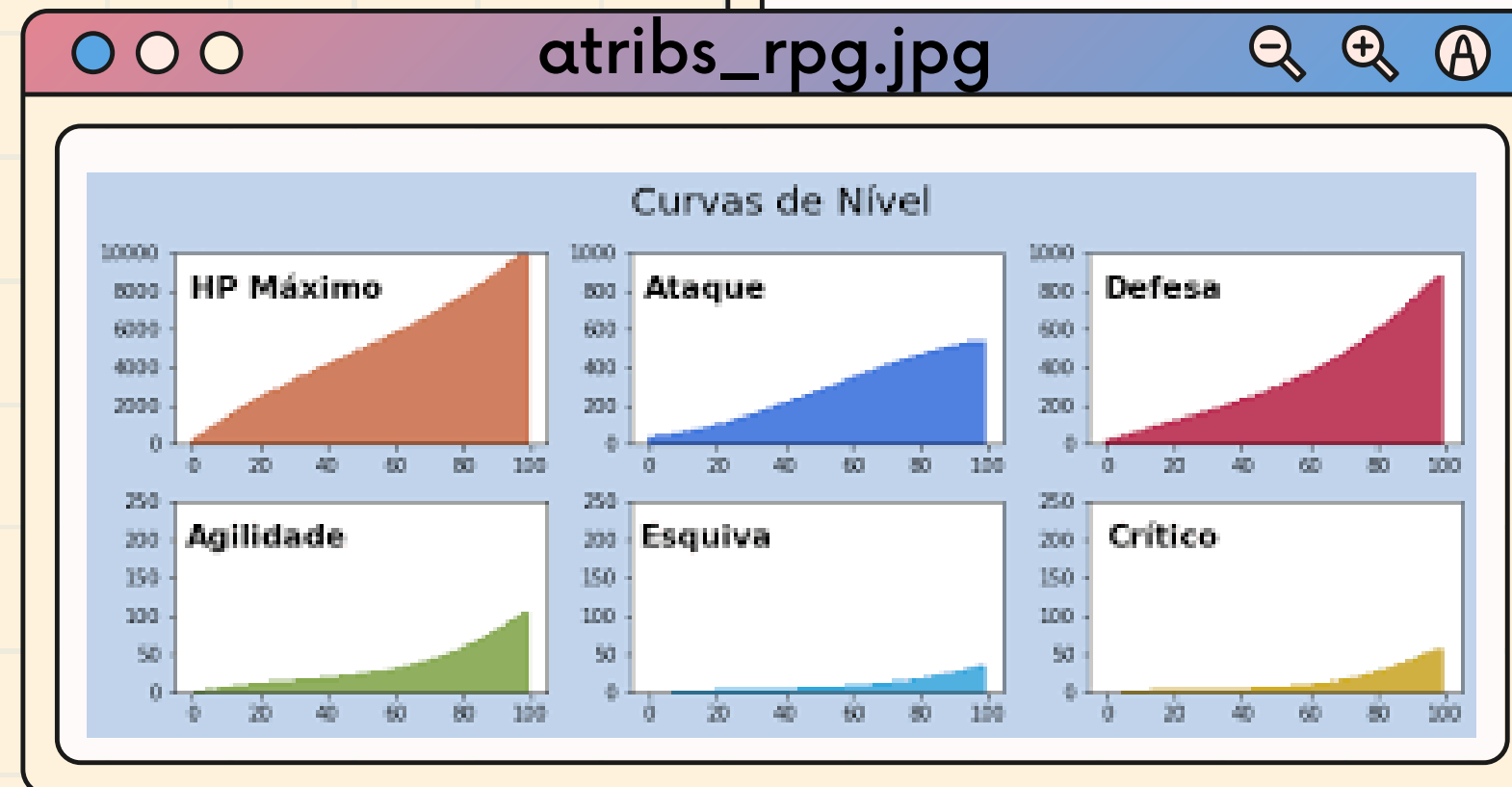
Ações que os objetos  
podem realizar.  
Quando invocado, ele  
realiza uma  
determinada tarefa.



# Atributos

```
rodas: number
motor: number
cor: string
modelo: string
marca: string
km: number
```

Características que pertencem a um objeto. Eles representam as informações ou dados que um objeto possui.

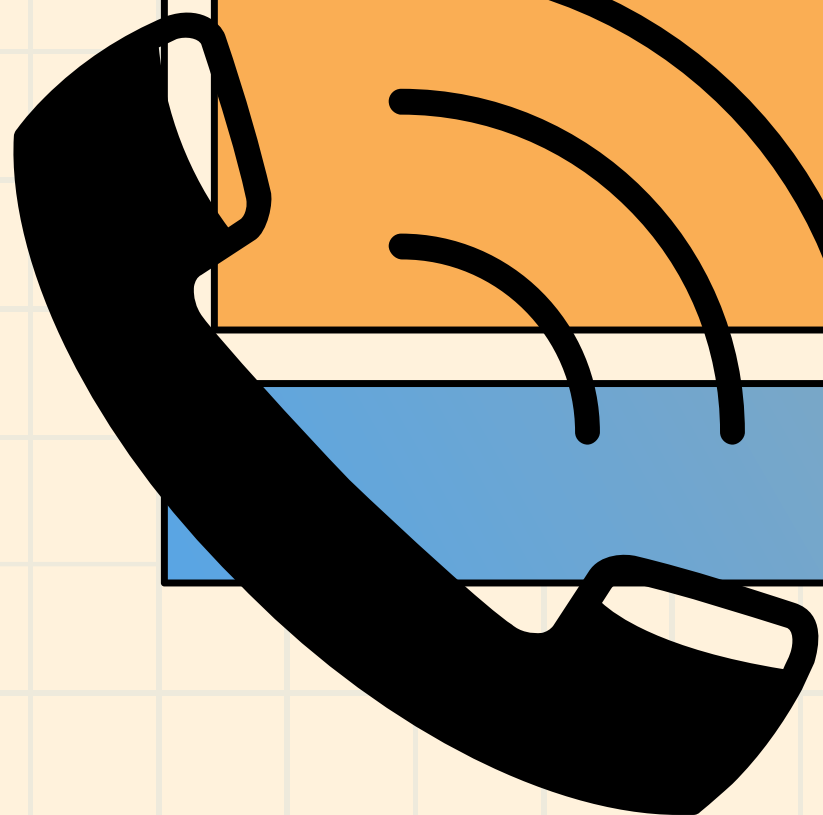


# **Atividade Inicial:**

**Escreva uma classe Data cuja instância (objeto) represente uma data. Sendo assim, terá os atributos dia, mês e ano.  
Criar o objeto e printá-lo.**


# Comunicação entre Classes

alô?



# Associações

Mas como funciona  
tudo isso?




```
class Restaurante{  
  cz: Cozinheiro  
  nome: string  
  endereco: string  
  num_end: number  
  
  constructor(cz: Cozinheiro, nome: string, endereco: string, num_end: number){  
    this.cz = cz  
    this.nome = nome  
    this.endereco = endereco  
    this.num_end = num_end  
  }  
}
```

```
class Cozinheiro{  
  nome: string  
  idade: number  
  ano_xp: number  
}
```



# Agregações


Uma forma de utilizar os atributos da classe associada.



```
apresentacaoChef(){  
  console.log(`Olá, sou o ${this.cz.nome}`);  
}
```

```
43 let chefEdivan = new Cozinheiro("Edivan", 38, 18)  
44 let rest = new Restaurante(chefEdivan, "LeBlanc", "Rua Rito Gomes", 420)  
45  
46 rest.apresentacaoChef()  
47
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

 powershell

```
PS C:\Users\Iuri Santos\Documents\241t> node .\Restaurante.js  
Olá, sou o Edivan  
PS C:\Users\Iuri Santos\Documents\241t> █
```



A window frame with a title bar at the top and a content area below it. The title bar has a blue gradient on the left and a tan gradient on the right, containing three window control icons (minimize, maximize, close) on the right side. The content area has an orange background and contains the text "Getters e Setters" in a large, bold, black font, followed by "input e output das classes." in a smaller, bold, black font.

# Getters e Setters

**input e output das classes.**

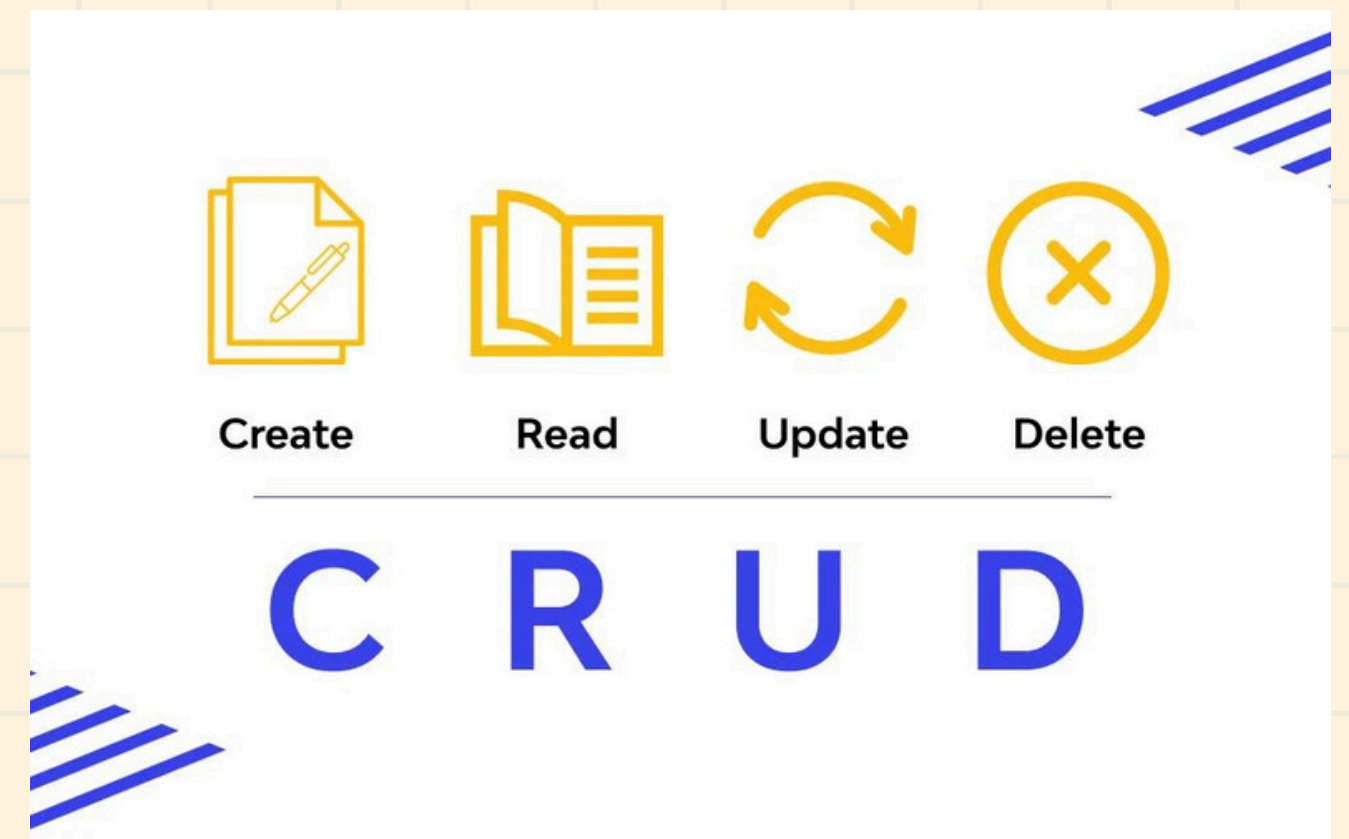
# CRUD

**Create** - criar o objeto e inserir valores iniciais

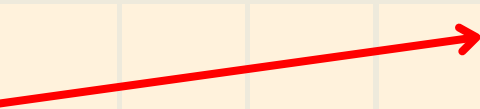
**Read** - ler os valores inseridos

**Update** - atualizar valores inseridos

**Delete** - deletar o objeto



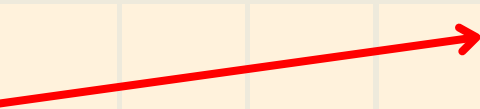
# Método get:



```
getCozinheiro(){  
  console.log(`Olá, meu nome é ${this.nome}, tenho ${this.  
  idade} anos e trabalho há ${this.ano_xp} anos.`);  
}
```

Mostra todos os atributos em apenas  
um método.

# Método set:



```
setCozinheiro(){
  let nomeUp = leitor.question("Qual o nome do cozinheiro? ")
  let idadeUp = leitor.question("Qual a idade do cozinheiro? ")
  let ano_xpUp = leitor.question("Qual o tempo de exp. do cozinheiro? ")
  this.nome = nomeUp
  this.idade = idadeUp
  this.ano_xp = ano_xpUp
}
```

Inserir e atualizar os valores dos atributos do objeto

# **Prática! Atividade:**

**Escreva uma classe Professor (com os mesmos atributos do cozinheiro) e uma classe Escola (com os mesmos atributos do restaurante, modificando o necessário). Ambas as classes necessitam dos getters e setters e o objeto da classe Escola necessita ter um atributo objeto da classe Professor.**

**Após isso criar um menu switch case com as opções: Criar Professor, Criar Escola, modificar Professor, modificar Escola, visualizar Professor, visualizar Escola e sair.**

# **Trabalho 1:**

**Fazer a partir do dia 27/06:**

**<https://github.com/iurisaints/typescriptClass/blob/main/241/TrabalhoUm.md>**

# **OBRIGADO.**

**EMAIL: IURISAINTS@GMAIL.COM**

**GITHUB: IURISAINTS**

**INSTAGRAM: IURI.COM.BR**

