

# Objetos

# O que vamos ver hoje?

Objetos:

- Estrutura
- Acessando e alterando valores
- Acessando valores diferentes:
  - objetos dentro de objetos
  - arrays dentro de objetos
  - array de objetos
- Adicionando propriedades
- Espalhamento ou spread

# Objetos

- **Objetos** são estruturas que nos permitem representar **dados mais complexos** de uma maneira mais **organizada**
- Com os objetos conseguimos criar **modelos do mundo real** de forma mais intuitiva/humanizada

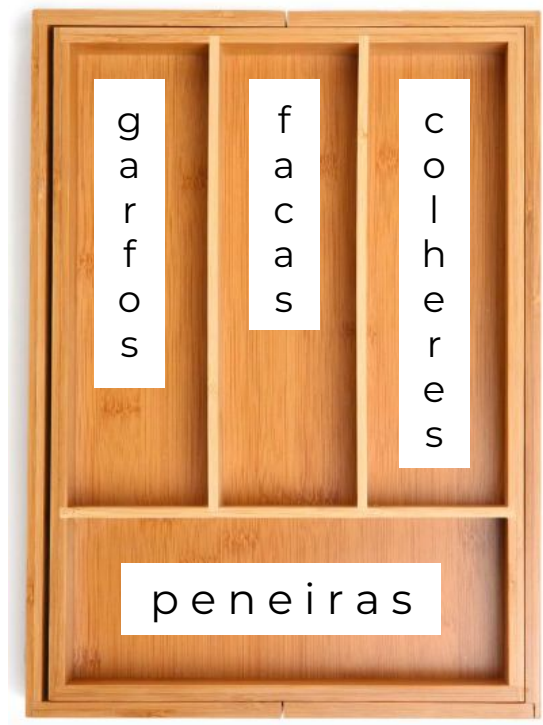
# Objetos

- Se fizéssemos uma comparação com a cozinha, as variáveis com valores dos tipos: **string**, **number** e **boolean** seriam gavetas pequenas e simples para guardar **um item**



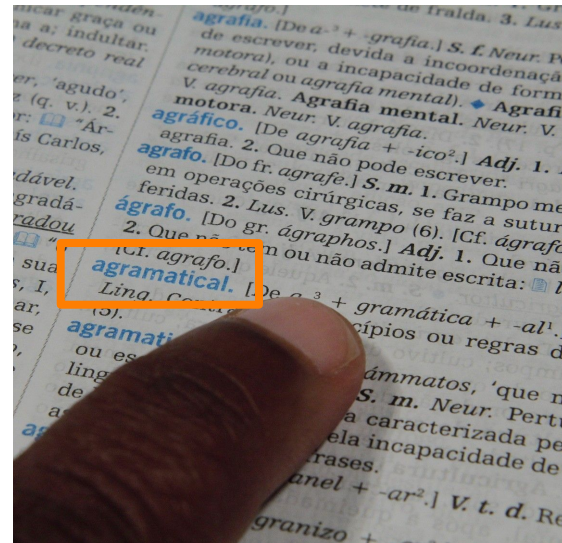
# Objetos

- Os **objetos** seriam uma gaveta maior e organizada, permitindo guardar **diversos itens** e cada separação possui um **identificador** para os diferentes itens



# Estruturas

- Objeto é uma **estrutura** análoga a um dicionário. Buscamos a **definição** da palavra por meio do seu nome (**identificador**)
- Assim como array está para listas, objeto está para um dicionário de definições



objeto é o dicionário e as palavras são as propriedades com seus respectivos valores

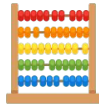
# Objetos

- As propriedades dos objetos podem assumir **quaisquer valores**
  - String, number, boolean, array, etc.
  - Funções (neste caso, quando estão dentro de um objeto, são chamadas de **método**)

# Estrutura de um objeto



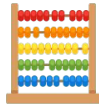
# Estrutura de um objeto



- Declaramos uma variável com **let** ou **const** e damos um **nome** ao objeto

```
const professor
```

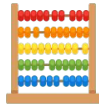
# Estrutura de um objeto



- Utilizamos **chaves** para representar a estrutura de um objeto

```
const professor = {}
```

# Estrutura de um objeto



- Dentro das chaves, podemos criar **propriedades** contendo **chave** e **valor**

```
const professor = {  
  nome: 'Leonardo',  
}
```

# Estrutura de um objeto

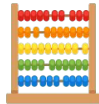


- Dentro das chaves, criamos uma **propriedade** contendo **chave** e **valor**

```
const professor = {  
  nome: 'Leonardo',  
}
```

← propriedade

# Estrutura de um objeto

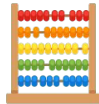


- Dentro das chaves, criamos uma **propriedade** contendo **chave** e **valor**

```
const professor = {  
  nome: 'Leonardo',  
}
```

↑  
**chave**

# Estrutura de um objeto



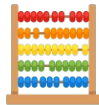
- Dentro das chaves, criamos uma **propriedade** contendo **chave** e **valor**

```
const professor = {  
  nome: 'Leonardo',  
}
```



**valor**

# Estrutura de um objeto

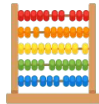


- Separamos propriedades com vírgula

```
const professor = {  
  nome: 'Leonardo',  
}
```

↑  
**separamos  
propriedades  
com vírgula**

# Estrutura de um objeto



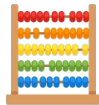
- Podemos ir inserindo novas propriedades no objeto

```
const professor = {  
  nome: 'Leonardo',  
  idade: 23  
}
```

← nova propriedade



# Estrutura de um objeto



- Abaixo, temos um objeto com duas propriedades: nome e idade

```
const professor = {  
  nome: 'Leonardo',  
  idade: 23,  
}
```

Vamos ver na prática!



# Estrutura de um objeto



- Os valores de uma chave também podem ser arrays e funções (nesse caso, métodos)


```
const professor = {  
  nome: 'Leonardo',  
  idade: 23,  
  tarefas: ['Lecionar', 'Responder dúvidas'],  
  contarPiada: function() {  
    console.log('aí tinha um papagaio')  
  }  
}
```

# Estrutura de um objeto

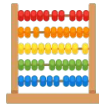


- Os valores de uma chave também podem ser **arrays** e funções (nesse caso, métodos)

```
const professor = {  
  nome: 'Leonardo',  
  idade: 23,  
  tarefas: ['Lecionar', 'Responder dúvidas'],  
  contarPiada: function() {  
    console.log('aí tinha um papagaio')  
  }  
}
```



# Estrutura de um objeto



- Os valores de uma chave também podem ser arrays e **funções** (nesse caso, **métodos**)

```
const professor = {  
  nome: 'Leonardo',  
  idade: 23,  
  tarefas: ['Lecionar', 'Responder dúvidas'],  
  contarPiada: function() {  
    console.log('aí tinha um papagaio')  
  }  
}
```

An orange line starts from the bullet point above and ends in an arrow pointing to the 'contarPiada' property in the code block.

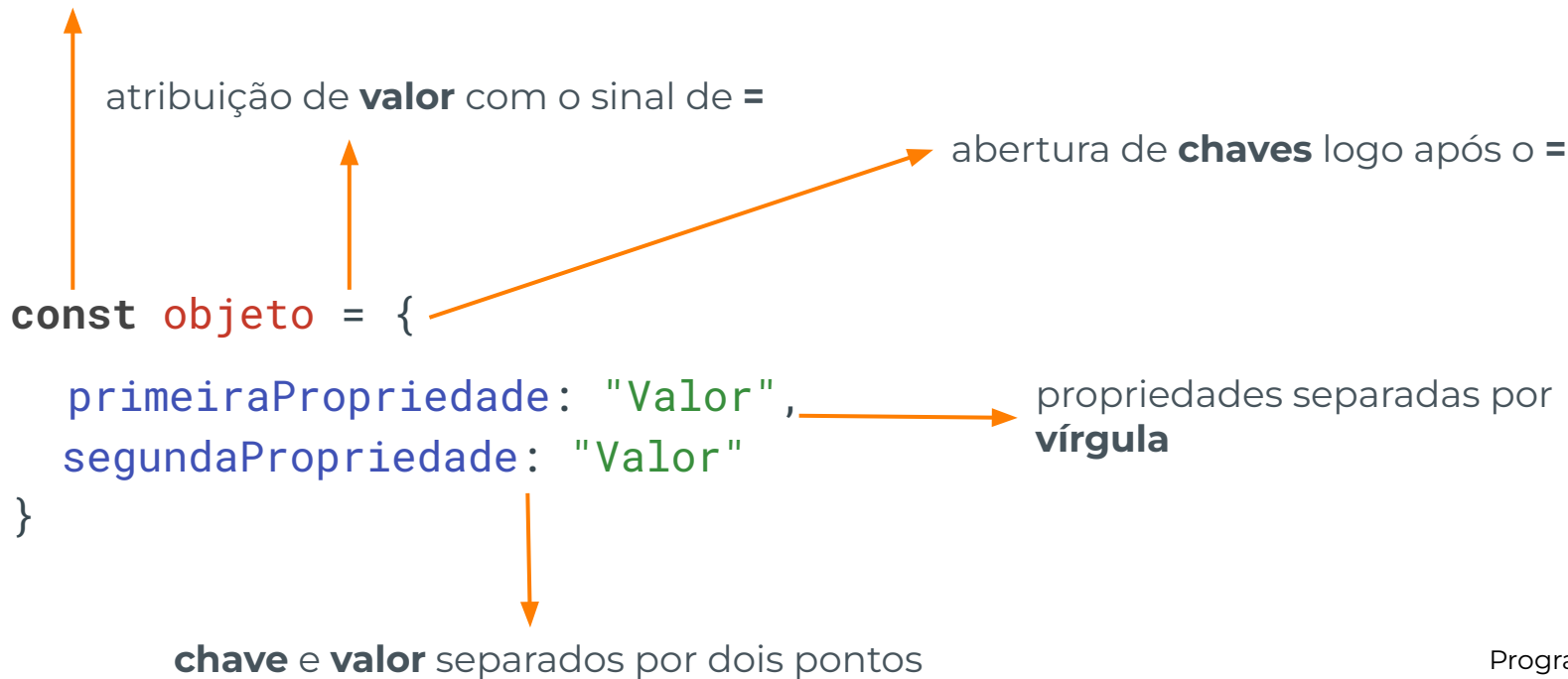
Vamos ver na prática!



# Estrutura padrão de um objeto



**declaração** com `let` ou `const`  
seguido do **nome** do objeto



# Acessando valores de um objeto

# Acessando e alterando propriedades

- Para **acessar** ou **alterar** as **propriedades** dos objetos, há duas sintaxes interessantes:
  - Notação do **ponto** ●  
(a mais "comum" entre as linguagens de programação)
  - Notação dos **colchetes** [ ]

# Notação de ponto .



# Notação de ponto

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

-  objeto
-  chave
-  valor

```
console.log(professor.idade)
```

# Notação de ponto ●

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

```
console.log(professor.idade)
```

nome do  
objeto



# Notação de ponto ●

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

```
console.log(professor.idade)
```

notação de  
ponto



# Notação de ponto ●

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

```
console.log(professor.idade)
```

nome da  
propriedade



Vamos ver na prática! 

# Notação de colchetes [ ]

# Notação de colchetes []

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

- objeto
- chave
- valor

```
console.log(professor["email"])
```

# Notação de colchetes [ ]

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

```
console.log(professor["email"])
```

notação de  
colchetes



# Notação de colchetes []

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

```
console.log(professor["email"])
```

string com o  
nome da chave



Vamos ver na prática!





# Alterando valores de um objeto

# Alterando valores

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

- objeto
- chave
- valor

```
professor.nome = 'Mika'
```

```
professor['email'] = 'profmika@gmail.com'
```

# Alterando valores

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

acessa a  
propriedade



```
professor.nome = 'Mika'
```

```
professor['email'] = 'profmika@gmail.com'
```

# Alterando valores

```
const professor = {  
  nome: "Leonardo",  
  idade: 23,  
  email: 'prof.lebc@gmail.com'  
}
```

atribui novo  
valor

```
professor.nome = 'Mika'
```

```
professor['email'] = 'profmika@gmail.com'
```

Vamos ver na prática!



# Exercício 1

- **Crie um objeto** que represente um filme. Ele deve ter dados da direção, o nome, o ano de lançamento, uma lista com o elenco e uma propriedade que diga se você já viu ou não.
- **Acesse e imprima** no console cada uma das propriedades: metade usando notação do ponto e a outra metade com notação de colchetes.

## Exercício 2

- **Crie** um objeto que represente uma pessoa. Essa pessoa precisa ter nome, idade, gênero musical preferido.
- **Acesse** e **imprima** no console as propriedades desse objeto, seguindo o modelo abaixo:

*"O nome da pessoa é \_\_\_\_, ela tem \_\_\_\_ anos e gosta muito de \_\_\_\_."*

# Fixação

- Objetos são estruturas que permitem a **representação** do mundo à nossa volta de uma maneira **mais intuitiva**
- Possuem **propriedades** com **chave e valor**
- Para acessar o conteúdo de dentro do objeto, existem as sintaxes do **ponto** e dos **colchetes**

# Acessando valores diferentes 🧐



# Acessando valores diferentes 🤪

- Não é incomum a existência de objetos dentro de objetos, objetos dentro de arrays, arrays de objetos...
- Pode parecer complicado, mas fica mais simples se pensarmos em **caminhos**



# Acessando objetos dentro de objetos {{ }}

# Acessando objetos dentro de objetos `{ { } }`

```
const donoDoPet = {  
  nome: "Leonardo Barbosa",  
  pet: {  
    nomeDoPet: "Bituca",  
    raca: "Vira-lata",  
    idade: 4  
  }  
}
```

- objeto
- chave
- valor

```
console.log(donoDoPet.pet.nomeDoPet)
```

# Acessando objetos dentro de objetos **{ { } }**

```
1 → const donoDoPet = {  
    nome: "Leonardo Barbosa",  
    pet: {  
        nomeDoPet: "Bituca",  
        raca: "Vira-lata",  
        idade: 4  
    }  
}
```

```
console.log(donoDoPet.pet.nomeDoPet)
```

# Acessando objetos dentro de objetos **{ { }}**

```
const donoDoPet = {  
  nome: "Leonardo Barbosa",  
  2 → pet: {  
    nomeDoPet: "Bituca",  
    raca: "Vira-lata",  
    idade: 4  
  }  
}
```

```
console.log(donoDoPet.pet.nomeDoPet)
```

# Acessando objetos dentro de objetos `{ { }}`

```
const donoDoPet = {  
  nome: "Leonardo Barbosa",  
  pet: {  
    nomeDoPet: "Bituca",  
    raca: "Vira-lata",  
    idade: 4  
  }  
}
```

3



```
    nomeDoPet: "Bituca",  
    raca: "Vira-lata",  
    idade: 4  
  }  
}
```

```
console.log(donoDoPet.pet.nomeDoPet) //Bituca
```

Vamos ver na prática!



# Acessando arrays dentro de objetos {[ ]}

# Acessando arrays dentro de objetos {[ ]}

```
const curso = {  
  nome: "Noturno Frontend",  
  linguagens: ["JS", "CSS", "HTML"]  
}
```

- objeto
- chave
- valor

```
console.log(curso.linguagens[0])
```



# Acessando arrays dentro de objetos {[ ]}

```
1 → const curso = {  
    nome: "Noturno Frontend",  
    linguagens: ["JS", "CSS", "HTML"]  
}
```

```
console.log(curso.linguagens[0])
```

# Acessando arrays dentro de objetos {[ ]}

```
const curso = {  
  nome: "Noturno Frontend",  
  2 → linguagens: ["JS", "CSS", "HTML"]  
}
```

```
console.log(curso.linguagens[0])
```

# Acessando arrays dentro de objetos {[ ]}

```
const curso = {  
  nome: "Noturno Frontend",  
  linguagens: ["JS", "CSS", "HTML"]  
}
```

3

acessa a primeira  
posição do array

```
console.log(curso.linguagens[0])
```

Vamos ver na prática!



# Array de objetos [{ }]

# Array de objetos [{ }]

- No seguinte exemplo, temos um array (lista) contendo três objetos

```
const professores = [  
  {nome: "Leonardo", modulo: 1},  
  {nome: "Mika", modulo: 2},  
  {nome: "Lucas", modulo: 3}  
]
```


```
console.log(professores[1].nome)
```

- objeto
- chave
- valor

# Array de objetos [{ }]

- No seguinte exemplo, temos um **array** (lista) contendo três objetos


```
const professores = [  
  {nome: "Leonardo", modulo: 1},  
  {nome: "Mika", modulo: 2},  
  {nome: "Lucas", modulo: 3}  
]
```



```
console.log(professores[1].nome)
```

# Array de objetos [{ }]

- No seguinte exemplo, temos um array (lista) contendo **três objetos**



```
const professores = [  
  {nome: "Leonardo", modulo: 1},  
  {nome: "Mika", modulo: 2},  
  {nome: "Lucas", modulo: 3}  
]
```

```
console.log(professores[1].nome)
```

# Array de objetos [{ }]

- Acessamos o objeto através da **posição** (index) que se encontra no array

```
const professores = [  
  {nome: "Leonardo", modulo: 1},  
  {nome: "Mika", modulo: 2},  
  {nome: "Lucas", modulo: 3}  
]
```

```
console.log(professores[1].nome) //Leonardo
```

posição do array  
que o objeto se  
encontra

Vamos ver na prática!





# Adicionando propiedades

# Adicionando propriedades

- Para **adicionar propriedades** aos objetos, podemos usar notação de ponto ou colchetes

```
const curso = {  
  nome: "Frontend",  
  linguagens: ["JS", "CSS", "HTML"]  
}
```

- Notação de ponto: `curso.numeroEstudantes = 50`
- Notação de colchetes: `curso['numeroEstudantes'] = 50`

Vamos ver na prática! 

## Exercício 3

- **Adicione** ao objeto do exercício 1 uma lista com os nomes dos personagens do filme.
- **Acesse** e **imprima** no console cada pessoa do elenco junto com seu respectivo personagem
- **Altere** a primeira pessoa do elenco por "Xuxa".
- **Imprima** no console todas as propriedades do objeto.

# Fixação

- Propriedades de objetos também podem ser arrays ou até mesmo outros objetos
- Para acessar esses valores seguimos o caminho, usando a notação de pontos (ou colchetes) e a posição dos elementos no array (ex: [0])

# **Espalhamento ou Spread**

# Espalhamento ou spread ...

- Existe uma sintaxe interessante, através da qual conseguimos realizar uma **cópia de um objeto (ou array) inteiro**
- Feita essa cópia, podemos manipular ela da maneira que quisermos (ex: mudar ou adicionar propriedades)
- Essa sintaxe é chamada de **espalhamento (ou spread)**

# Espalhamento ou spread ...

- Abaixo, copiamos o usuario e sobrescrevemos as propriedades nome e idade com novos valores

```
const usuario = {  
  nome: 'Prof',  
  idade: 25,  
  email: 'prof@senacrs.com.br',  
  cidade: 'São Paulo'  
}
```

```
const novoUsuario = {  
  ...usuario,  
  nome: 'João',  
  idade: 28  
}
```

# Espalhamento ou spread ...

- O spread é simbolizado por três pontos

```
const usuario = {  
  nome: 'Prof',  
  idade: 25,  
  email: 'prof@senacrs.com.br'  
  cidade: 'São Paulo'  
}
```

```
const novoUsuario = {  
  ...usuario,  
  nome: 'João',  
  idade: 28  
}
```

copiando  
propriedades do  
objeto usuario



# Espalhamento ou spread ...

- Propriedades com mesmo nome são sobrescritas

```
const usuario = {  
  nome: 'Prof',  
  idade: 25,  
  email: 'prof@senacrs.com.br',  
  cidade: 'São Paulo'  
}
```

```
const novoUsuario = {  
  ...usuario,  
  nome: 'João',  
  idade: 28  
}
```

↑  
propriedades com nomes  
iguais adicionadas por  
último são sobrescritas

# Espalhamento ou spread ...

```
const usuario = {  
  nome: 'Prof',  
  idade: 25,  
  email: 'prof@senacrs.com.br',  
  cidade: 'São Paulo'  
}
```

```
const novoUsuario = {  
  nome: 'João',  
  idade: 28,  
  email: 'prof@senacrs.com.br',  
  cidade: 'São Paulo'  
}
```

Vamos ver na prática!



# Espalhamento ou spread ...

- Copiando arrays

```
const listaDeNomes = ["Leonardo", "João", "Bruno"]
```

```
const copiaListaDeNomes = [...listaDeNomes]
```

```
console.log(copiaListaDeNomes) //["Leonardo", "João", "Bruno"]
```

# Espalhamento ou spread ...

- Copiando arrays

```
const listaDeNomes = ["Leonardo", "João", "Bruno"]
```

```
const copiaListaDeNomes = [...listaDeNomes]
```

```
console.log(copiaListaDeNomes) //["Leonardo", "João", "Bruno"]
```

# Espalhamento ou spread ...

- Sobrescrevemos valores através do seu **index**

```
const listaDeNomes = ["Leonardo", "João", "Bruno"]
```

```
const copiaListaDeNomes = [...listaDeNomes]
```

```
copiaListaDeNomes[0] = "Zé"
```

```
console.log(copiaListaDeNomes) //["Zé", "João", "Bruno"]
```

Vamos ver na prática!



# Exercício 4

- **Crie** uma **função** que receba um objeto de pessoa (Exercício 2) e crie um novo objeto mantendo as propriedades originais e acrescentando:
  - Uma propriedade com a **lista** de suas comidas preferidas;
  - Uma propriedade que seja um **objeto**, com nome e idade, para representar o melhor amigo da pessoa.
- Ainda na função, imprima no console as propriedades desse objeto seguindo o modelo abaixo:

*"O nome da pessoa é \_\_\_ e suas comidas preferidas são \_\_\_, \_\_\_ e \_\_\_\_\_. Seu melhor amigo se chama \_\_\_ e tem \_\_\_ anos"*

# Resumo

# Resumo



- **Objetos** são uma sintaxe que permite que a gente modele o mundo real de uma maneira mais fiel
- Os objetos possuem **propriedades**, que possuem **chave** e **valor**
- O valor das propriedades pode ser qualquer tipo, inclusive **funções**



# Resumo



- Se o valor da propriedade é uma função, chamamos ela de **método** do objeto
- Conseguimos fazer uma cópia do objeto, ou então acessar só algumas das propriedades dele utilizando as sintaxes de **spread**. O mesmo vale para arrays

# Dúvidas?



Programa  
**3000 TALENTOS TI**  
Obrigado(a)!