

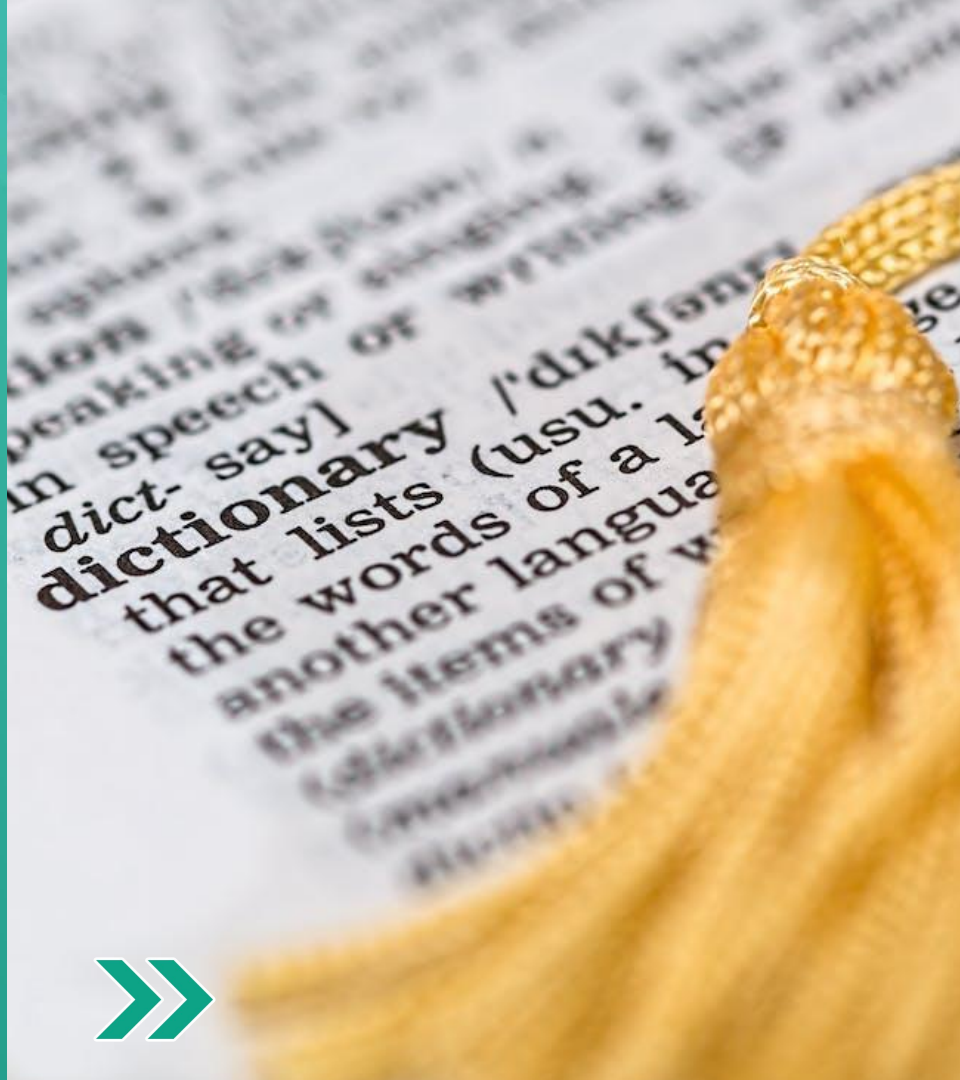


escola  
britânica de  
artes criativas  
& tecnologia

# Profissão: Engenheiro Front-End



# GLOSSÁRIO



# Orientação a Objetos com JavaScript

- Introdução
- Crie objetos
- Verifique os tipos e instâncias
- Encapsulamento



Dica: para encontrar rapidamente a palavra que procura aperte o comando CTRL+F e digite o termo que deseja achar.



# Introdução



# Introdução

## Glossário

### ● Encapsulamento

Envolve a combinação de dados e comportamentos relacionados em uma única unidade, chamada de objeto. Ele visa proteger os dados internos de uma classe ou objeto, ocultando-os do acesso direto e fornecendo métodos públicos para interagir com esses dados.

### ● Herança

Refere-se à capacidade de um objeto herdar propriedades e métodos de outro objeto. Isso permite que um objeto compartilhe características de outro objeto, facilitando a reutilização de código e a criação de hierarquias de objetos.

### ● Polimorfismo

Essa palavra tem origem no grego antigo e é composta pelos termos "poli", que significa "muitos", e "morfê", que significa "forma". Esse termo é amplamente utilizado na área da programação e foi adotado para descrever a capacidade de um objeto ou função se comportar de maneiras diferentes, dependendo do contexto em que é utilizado.



# Crie objetos



# Crie objetos

## Glossário

### • **Const**

Palavra reservada utilizada para criar constantes, são como variáveis porém não podem ter seu valor redefinido.

### • **Funções construtoras no Javascript**

São funções especiais que são usadas para criar e inicializar objetos. Elas são utilizadas em conjunto com a palavra-chave *"new"* para criar novas instâncias de objetos com base em um "modelo" fornecido pela função construtora.



# Verifique os tipos e instâncias





# Verifique os tipos e instâncias

## Glossário

### **Instanceof**

Operador usado para verificar se um objeto é uma instância de uma determinada classe ou função construtora. Ele retorna *true* se o objeto for uma instância da classe e *false* caso contrário.



# Encapsulamento



# Encapsulamento

## Glossário

### ● Encapsulamento privado:

São acessíveis somente dentro da própria classe ou objeto. Não podem ser acessados ou modificados diretamente por outras partes do código. São usados para esconder a implementação interna de uma classe e proteger os dados sensíveis. O acesso a esses elementos é geralmente feito por meio de métodos públicos (*getters* e *setters*) que controlam a manipulação dos dados. Podem ser usados para realizar validações, cálculos automáticos ou qualquer lógica adicional necessária antes de permitir o acesso ou a modificação dos dados. Em geral, têm nomes precedidos por um *underscore* (*\_*) para indicar que são privados, embora isso seja apenas uma convenção de nomenclatura.

### ● Encapsulamento público

Atributos e métodos públicos acessíveis a partir de qualquer parte do código, tanto dentro como fora da classe. Podem ser acessados e modificados livremente por outras partes do código. São frequentemente usados para fornecer interfaces de acesso aos dados e funcionalidades de um objeto. Podem ser chamados diretamente sem a necessidade de métodos intermediários. Em geral, têm nomes descritivos e intuitivos.



# Encapsulamento

## Glossário

### **Getters e setters**

São métodos especiais em uma classe que permitem o acesso e a atribuição de valores a atributos privados. Eles fornecem um controle mais preciso sobre o acesso aos dados de um objeto, permitindo a validação, o cálculo ou qualquer outra lógica adicional durante a leitura e gravação dos valores dos atributos. Os getters são definidos usando a palavra-chave `get`, seguida pelo nome do atributo. Os setters são definidos usando a palavra-chave `set`, seguida pelo nome do atributo. Ambos são definidos dentro de uma classe e podem ser chamados como se fossem atributos.



# Bons estudos!

