

# Variáveis Simples

**Disciplina:** Linguagem de Programação



## Conteúdos:

Variáveis simples.

## Habilidade(s):

- Declarar e atribuir valor a variáveis;
- Manipular dados na programação;
- Aplicação *mobile* de variáveis na programação js.

# Bloco 1

---

Vamos explorar as variáveis!

# Leitor feroz

Ao entrar em uma biblioteca, qual é o gênero literário que você escolhe?

Suspense

Drama

Aventura

Romance

Terror

Comédia

Ação

Ficção científica



# Um mundo vasto

Seja a história de um incrível bruxo ou o romance entre uma humana e um vampiro, no mundo atual, estamos rodeados de histórias que permeiam **as nossas memórias**.

É como uma incrível biblioteca repleta de informações, apenas aguardando para que possamos lê-la.

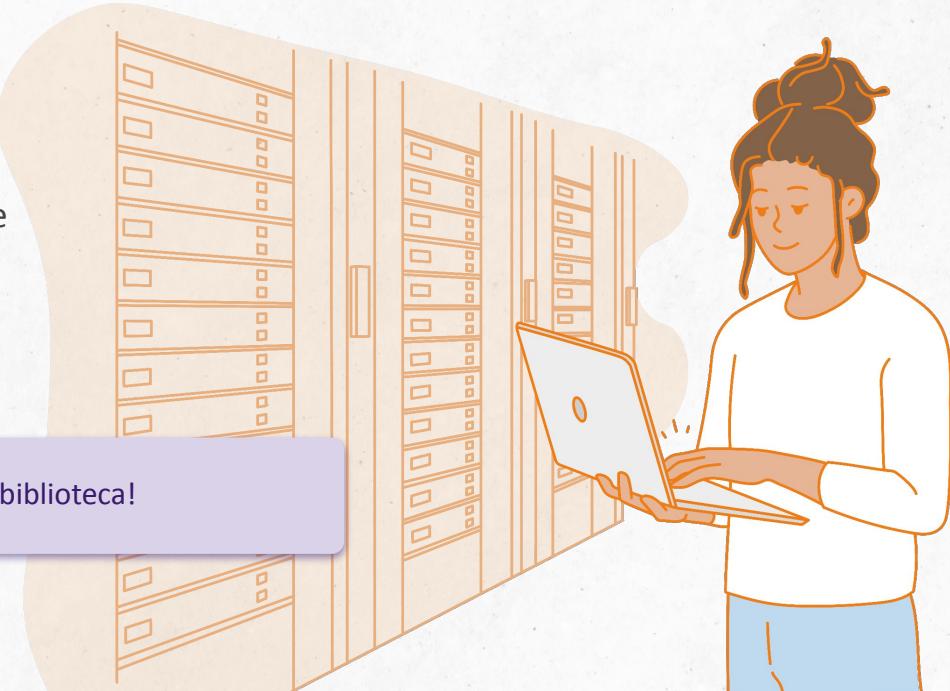


# O que isso tem a ver com as variáveis?

No mundo da programação, um tipo de elemento também é responsável por armazenar todos os dados de forma temporária ou permanente: as **variáveis**.

Assim como o seu próprio nome diz, esses tipos de dados podem **variar** a depender do tipo de programa.

É como uma vasta biblioteca!



# Onde as variáveis podem ser encontradas?

No nosso dia a dia, ainda que não notemos, lidamos com uma série de situações que envolvem as variáveis.



No histórico de pesquisa do navegador.



No armazenamento das curtidas que você deu na foto da amada.



Na concentração do histórico dos episódios que você assistiu.

# Funcionamento das variáveis

As variáveis podem conter inúmeros tipos de dados, como:

Números inteiros

Números de ponto flutuante

*Strings* (textos)

Objetos e *arrays*

# Atribuindo valor a uma variável

Para formar uma variável, é necessário atribuir um nome e um valor a ela.

Imagine que você está criando uma lista de tarefas para realizar no dia.

```
var tarefa = "Fazer compras de supermercado"  
var tarefa = "Lavar a louça"  
var tarefa = "Limpar o chão"
```

As variáveis são usadas para armazenar **valores** (tarefas) e facilitam o acesso posterior aos dados armazenados. Aqui é onde a magia acontece.

No exemplo, ‘tarefa’ se refere ao nome. As informações depois do = estão relacionadas aos valores.

# Palavras-chave das variáveis

As variáveis em JavaScript são criadas usando palavras-chave seguidas do nome que você atribuiu à variável.

## var

É possível modificá-la e mudá-la de posição a qualquer momento.

```
var idade = 10;  
idade = 12;
```

## let

Os dados variam, mas ela só pode ser modificada dentro de um bloco.

```
let quantidadeBrinquedos = 5;  
if (quantidadeBrinquedos > 7) {
```

## const

É uma variável com um dado que não pode ser mudado.

```
const PI = 3.14159;  
PI = 4;
```

# Bloco 2

---

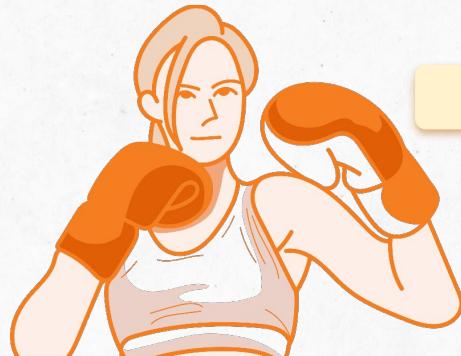
Explorando os aspectos da declaração de variáveis.

## Vamos realizar uma dinâmica?

Separem-se em duas grandes equipes.

A equipe A deverá citar **um exemplo de variável** e a equipe B deverá dizer corretamente de qual tipo a variável se trata (**let, const ou var**). Em seguida, deverão inverter: a equipe B deverá citar um exemplo de variável, enquanto a equipe A dirá a qual tipo ela pertence.

A rodada é finalizada assim que uma das equipes errar, iniciando uma nova. Ao todo, deverão ser cinco rodadas. Ganha a equipe que tiver o maior número de acertos.



Equipe A

VS

Equipe B



## Declaração de variáveis: uma boa prática

A declaração de uma variável em um programa é fundamental para evitar erros e inconsistências no código.

É considerada uma boa prática sempre declarar variáveis antes de usá-las, pois isso torna o **código mais legível** e ajuda a **evitar problemas**.



## Case sensitive

JavaScript é uma linguagem *case sensitive*, o que significa que ela faz diferença entre letras maiúsculas e minúsculas.



Por exemplo, você declara duas variáveis, 'nome' e 'Nome'.



O JavaScript entenderá que são variáveis diferentes.

# Formas de declarar variável com Javascript

No JavaScript, você pode declarar variáveis de quatro maneiras diferentes.

Automaticamente

**Não recomendado.**

Usando var

**Declara uma variável global ou local, opcionalmente, inicializando-a com um valor.**

Usando let

**Declara uma variável de escopo local, opcionalmente, inicializando-a com um valor.**

Usando const

**Declara uma variável de escopo global, que será usada apenas para leitura.**

# Escolha entre const, let e var

Algumas dicas são essenciais na hora de declarar variáveis:



Usar **const** quando a variável não deve ser alterada após a atribuição inicial;



Usar **let** quando não puder usar **const**;



Usar **var** apenas se precisar dar suporte a navegadores mais antigos.

# Regra CamelCase

É uma convenção de nomenclatura usada em programação, incluindo JavaScript.

O nome é escrito de forma que cada palavra subsequente comece com uma letra maiúscula, exceto a primeira palavra.

Existem duas variações comuns e a diferença está na letra maiúscula da primeira letra.

## CAMELCASE

nomeDoUsuario, idadeDaPessoa,  
totalDeProdutos



## PASCAL CASE

NomeDoUsuario, IdadeDaPessoa,  
TotalDeProdutos

# Bloco 3

---

Atribuindo valores às variáveis.

## Duas verdades e uma mentira

Analise as afirmações abaixo e identifique quais são as verdadeiras e qual é a falsa.

**const** se refere ao tipo de variáveis constantes, cujo valor não se altera.

Qualquer dado pode ser armazenado em uma variável.

A forma mais recomendada de declarar variáveis é a **automática**.

# Gabarito

E aí, acertou?

**const** se refere ao tipo de variáveis constantes, cujo valor não se altera.

Qualquer dado pode ser armazenado em uma variável.

A forma mais recomendada de declarar variáveis é a **automática**.

# Atribuir valores

Imagine que você tem um espaço de armazenamento com um nome único, como uma gaveta, onde você pode guardar algo.

Como você pode ter visto, chamamos esse espaço de armazenamento de "variável".

É como dar um nome a uma gaveta para que você possa colocar coisas lá dentro e encontrar com facilidade mais tarde.



Para colocar algo nessa gaveta (variável), usamos uma seta imaginária chamada de **operador de atribuição**, representada por um sinal de igual (=). Você coloca o que deseja guardar à direita do sinal de igual e o nome da gaveta à esquerda.

```
minhaVariavel = 5
```



# As variáveis contêm valores

Observe o exemplo a seguir.

```
var = x = 5
```

```
var y = 6
```

```
var = x = 5
```

```
var y = 6
```

```
var z = x+y
```

Qual é o valor de z?

# As variáveis contêm valores

E aí, acertou?

```
var = x = 5
```

```
var y = 6
```

```
var = x = 5
```

```
var y = 6
```

```
var z = x+y
```

**Isso mesmo, 11!**



Você pode exibir o valor no console, combiná-lo com outros valores ou utilizá-lo em cálculos.

## É válido lembrar

As variáveis podem ser reatribuídas com novos valores durante a execução do programa, permitindo que os dados sejam atualizados e reutilizados.



# Bloco 4

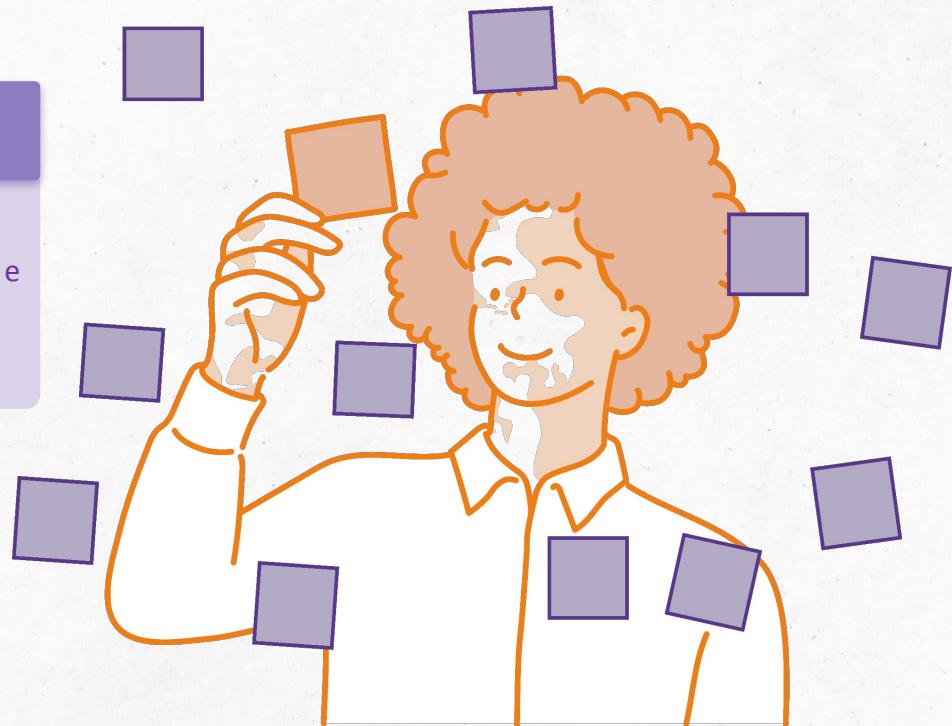
---

Explorando as regras de identificadores de variáveis e  
conhecendo os tipos de escopo.

# Mural de variáveis

Vamos elaborar um mural?

Atribua uma função a uma variável, escreva em um *post-it* e cole na cartolina.



# Como identificar variáveis?

Os identificadores são palavras utilizadas para identificar variáveis e possuem algumas regras que devem ser consideradas.

O primeiro caractere é uma letra (a-z ou A-Z), caracteres especiais \_(sublinhado) ou  $\$$  (cifrão).

Os próximos caracteres podem ser os mesmos vistos na primeira regra acrescidos de caracteres numéricos de 0 a 9.

Palavras reservadas da linguagem não podem ser usadas como identificadores.

Exemplo: **var** é uma palavra reservada para declaração de uma variável e um *array* é uma palavra reservada para uma variável composta.

# Escopo de variáveis

É como determinamos onde uma variável pode ser vista e usada em um determinado programa.

A seguir, confira os principais tipos de escopo.



# Escopo do bloco

Variáveis declaradas nesse escopo são visíveis somente dentro de um bloco.

Introduzido com o ES6 (ECMAScript 2015).

Usado com as palavras-chave **let** e **const**.

Variáveis declaradas dentro de um bloco de código só são visíveis dentro desse bloco.

São destruídas após o bloco ser executado.

```
if (true) {  
    let mensagem = "Isso é um exemplo"; // mensagem só é visível aqui dentro  
}  
console.log(mensagem); // Isso resultará em um erro, mensagem não é definida
```

# Escopo da função

Significa que a variável é visível apenas dentro da função onde foi declarada e não pode ser acessada de fora dessa função.

É o escopo onde variáveis declaradas com **var**, **let** ou **const** dentro de uma função são acessíveis.

Essas variáveis só podem ser vistas dentro da função onde foram declaradas.

Variáveis declaradas fora da função têm escopo global.

```
function exemploEscopo() {  
  let local = "Variável local"; // local só é visível dentro desta função  
  console.log(local);  
}  
exemploEscopo();  
console.log(local); // Isso resultará em um erro, local não é definida  
fora da função
```

# Escopo global

Variáveis declaradas nesse escopo podem ser acessadas e modificadas em qualquer parte do código, dentro de funções, blocos ou no escopo global.

Variáveis declaradas fora de qualquer bloco ou função são visíveis globalmente.

Podem ser acessadas e modificadas dentro de funções, blocos ou no escopo global.

```
let global = "Variável global"; // global é visível em todo o código
```

```
function exemploGlobal() {  
    console.log(global); // Você pode acessar global aqui dentro da função  
}  
exemploGlobal();
```

```
console.log(global); // E também aqui fora da função
```

# Bloco 5

---

Investigando os tipos de dados JavaScript.

# Tipos de dados JavaScript

Para poder operar em variáveis, é importante saber sobre os seus tipos.

Sem tipos de dados, um computador não pode resolver problemas com segurança.

Atualmente, a JavaScript possui oito tipos de dados que conheceremos a seguir.



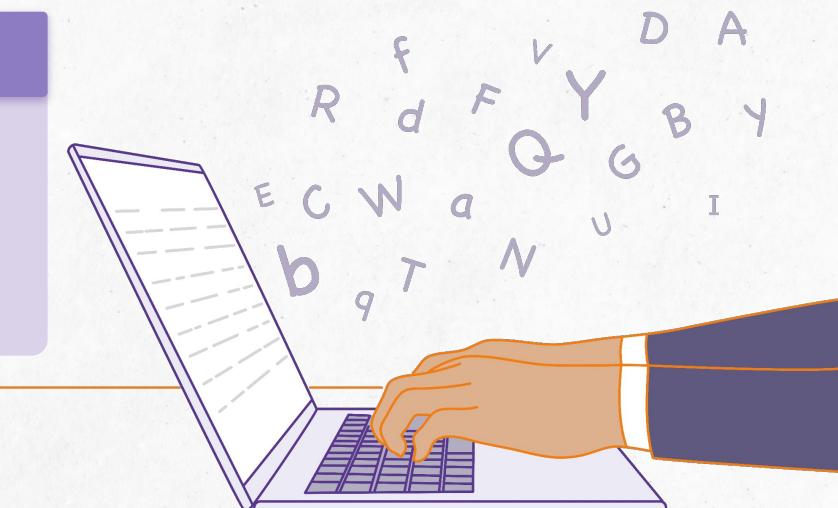
# String

As *strings* são usadas para armazenar texto.

Qualquer valor que seja armazenado entre aspas (simples ou duplas) é considerado uma *string* em JavaScript.

## Exemplo:

```
let nome = "João"  
  
let frase = "Isso é uma frase."x  
  
let numeroTexto = "42"
```



# Número

Armazenam valores numéricos.

A linguagem JavaScript não diferencia entre números inteiros e números de ponto flutuante.

## Exemplo

```
let numeroInteiro = 5  
let numeroDecimal = 3.14
```



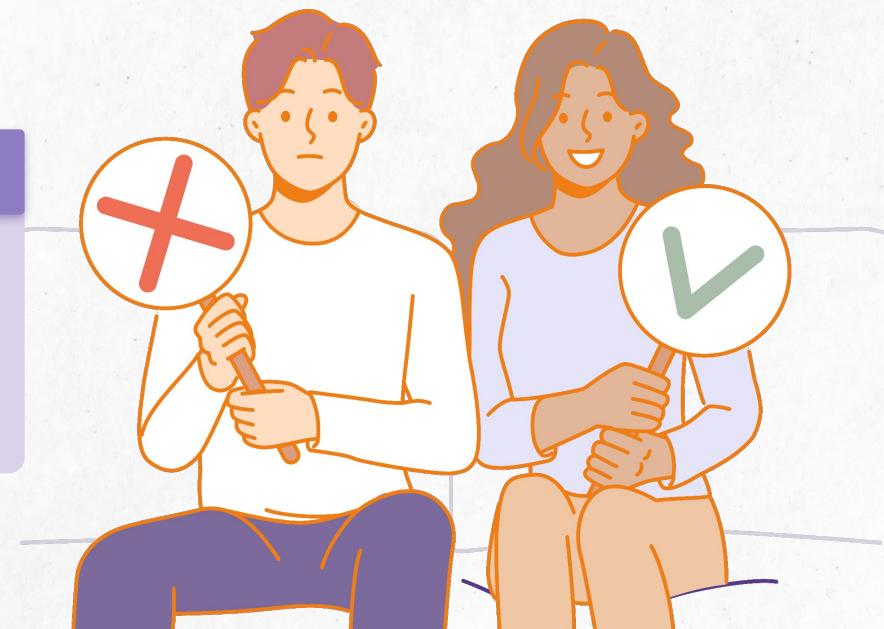
# Booleano

Representam valores lógicos, ou seja, verdadeiro (*true*) ou falso (*false*). São usados para expressar condições.

## Exemplo

```
let temChocolate = true
```

```
let estaChovendo = false
```

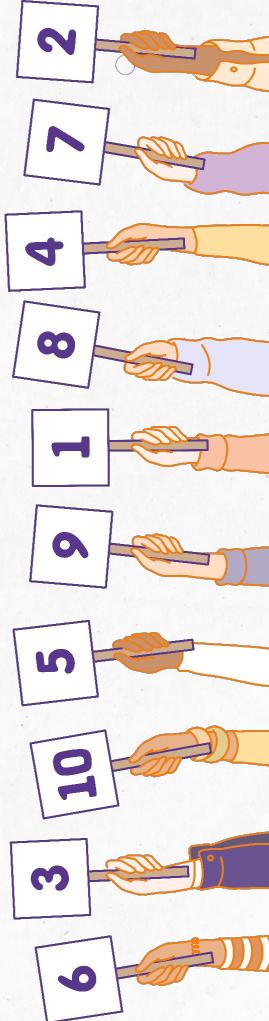


# Bigint

É usado para armazenar valores inteiros grandes que não podem ser representados com precisão usando números normais em JavaScript.

## Exemplo:

```
let numeroGrande =  
1234567890123456789012345678901234567890n
```



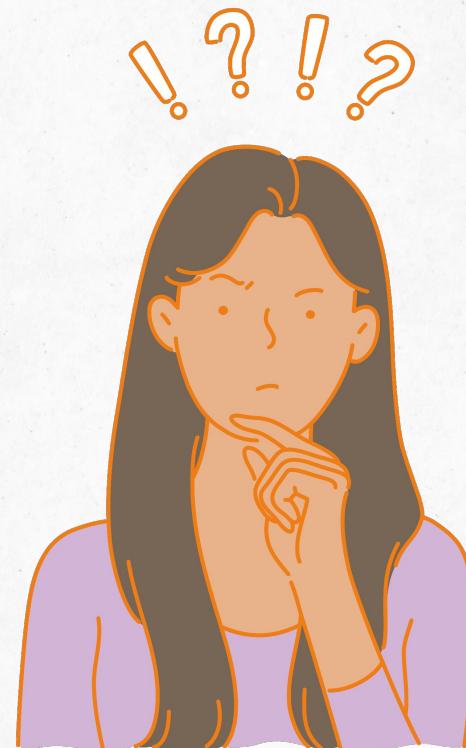
# Indefinido e nulo

São usados quando uma variável não tem um valor atribuído.

*Undefined* é usado por padrão quando uma variável é declarada, mas não recebe valor. *Null* é usado quando queremos representar a ausência de valor explicitamente.

## Exemplo:

```
let indefinida; // A variável indefinida é undefined por padrão  
  
let nula = null; // A variável nula é explicitamente definida como nula
```

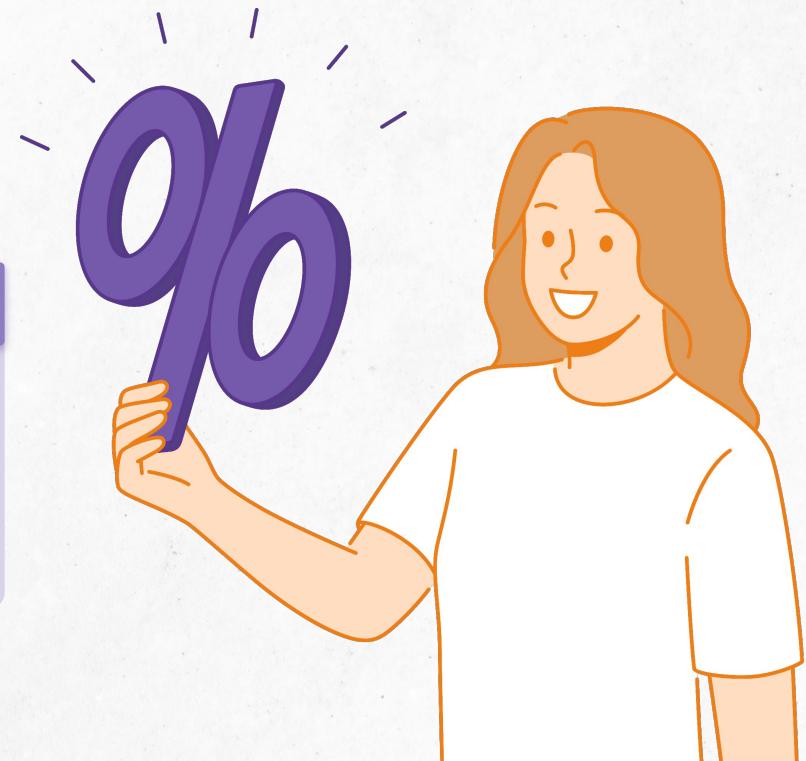


# Símbolo

São valores únicos usados principalmente como chaves de propriedades em objetos para evitar colisões de nomes de propriedades.

## Exemplo

```
const chaveUnica = Symbol('chaveUnica')
```

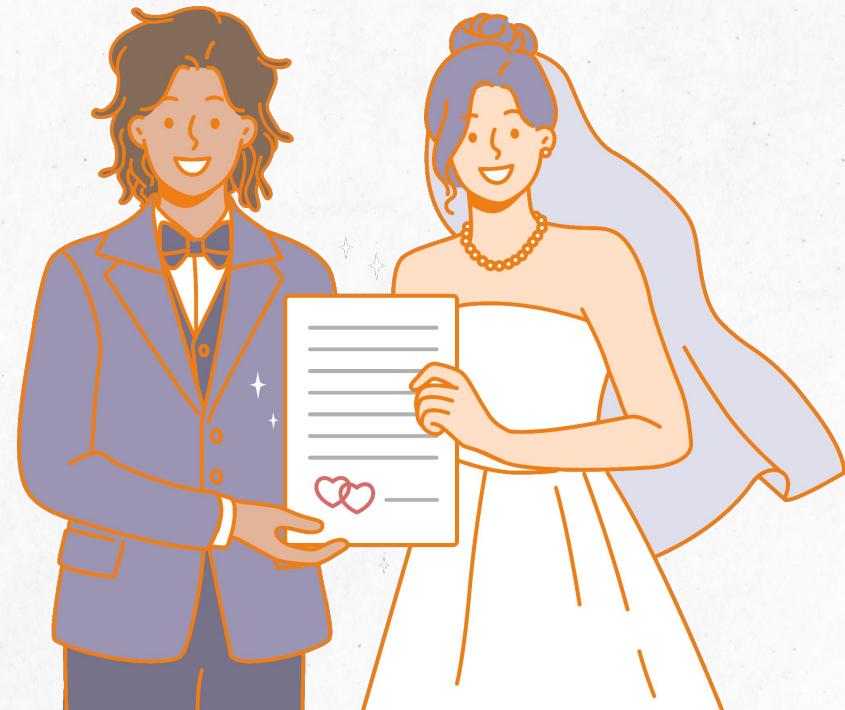


# Objeto

São estruturas de dados complexas que armazenam informações relacionadas em pares chave-valor.

## Exemplo

```
const pessoa = {  
    nome: "Maria",  
    idade: 30,  
    casada: true  
}
```



# Acesso a prioridades

Você pode usar a notação de ponto para acessar as propriedades de um objeto. Essas propriedades são valores associados a um objeto.

## Exemplo:

```
const pessoa = {  
    nome: "João",  
    idade: 30,  
};  
console.log(pessoa.nome)  
console.log(pessoa.idade)
```



# Chamada de métodos

Métodos são funções associadas a um objeto. Eles podem ser chamados usando a notação de ponto.

## Exemplo:

```
const calculadora = {  
    somar: function (a, b) {  
        return a + b;  
    },  
    subtrair: function (a, b) {  
        return a - b;  
    }  
};  
const resultadoSoma = calculadora.somar(5, 3)  
const resultadoSubtracao = calculadora.subtrair(10, 4)
```



# Bloco 6

---

Hora de praticar tudo o que vimos até aqui.

# Vamos realizar uma dinâmica?

Agora, vocês terão que aplicar tudo o que aprenderam até aqui. Dividam-se em grupos!

## Criação

Criem um cenário em que precisem usar variáveis em JavaScript.

## Distribuição

Entreguem para outro grupo.

## Discussão

Os grupos devem discutir qual tipo de variável seria mais apropriado para cada situação.

## Discussões e reflexões

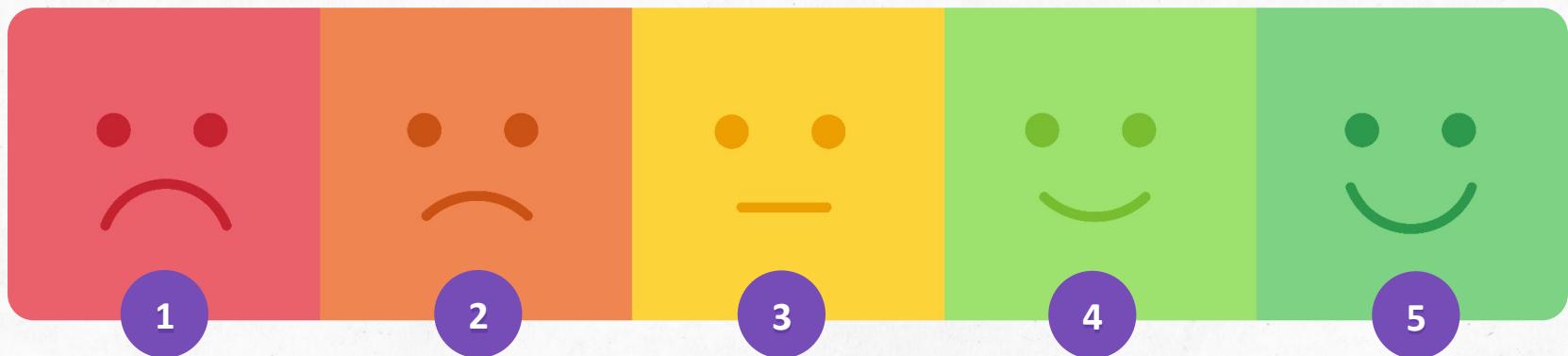


## Fechamento: avaliação de aprendizagem

Chegou o momento de avaliarmos o nosso nível de compreensão do conteúdo da aula.

Classifique o seu **nível de compreensão** em uma escala de 1 a 5, sendo 1 para “ainda estou perdido” e 5 para “entendi tudo e gostei do que aprendi!”.

Em seguida, justifique a sua resposta.



# Referências Bibliográficas

PROZ EDUCAÇÃO. *Desenvolvimento para Dispositivos Móveis I.* 2023.