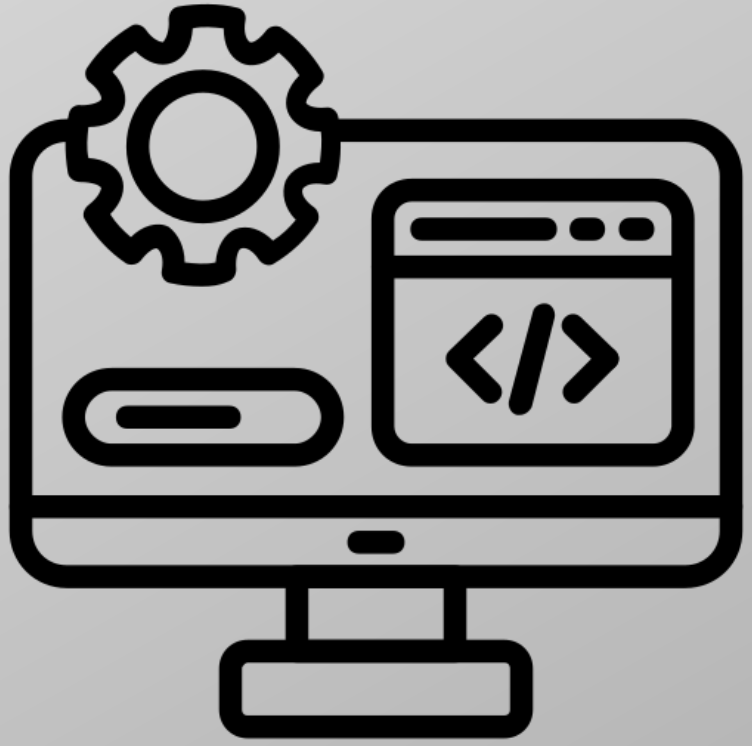


Desenvolvimento BACK-END I

Professor: Douglas Legramante

E-mail: douglas.legramante@ifro.edu.br

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

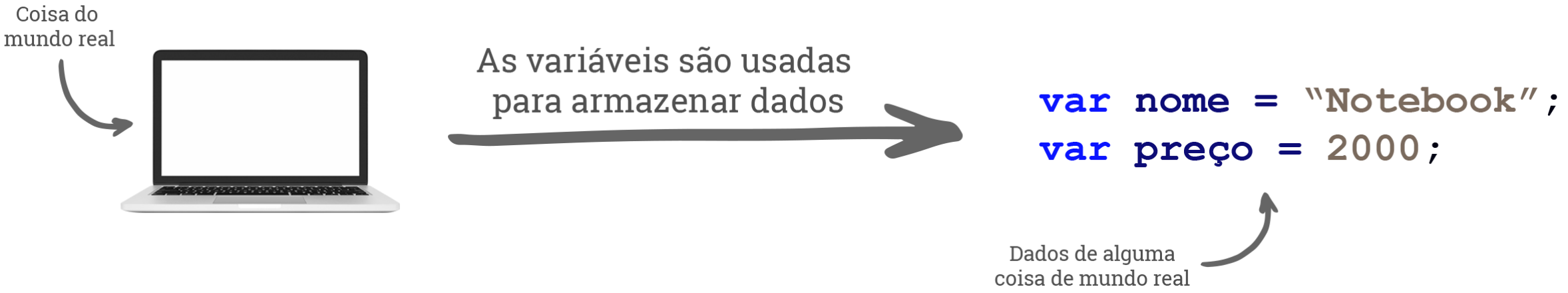
JavaScript

Aula 02 – Variáveis e tipos de dados

Declarando uma variável

INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

Variáveis são usadas para armazenar dados.



Para declarar uma variável usamos var, let ou const.

Atenção

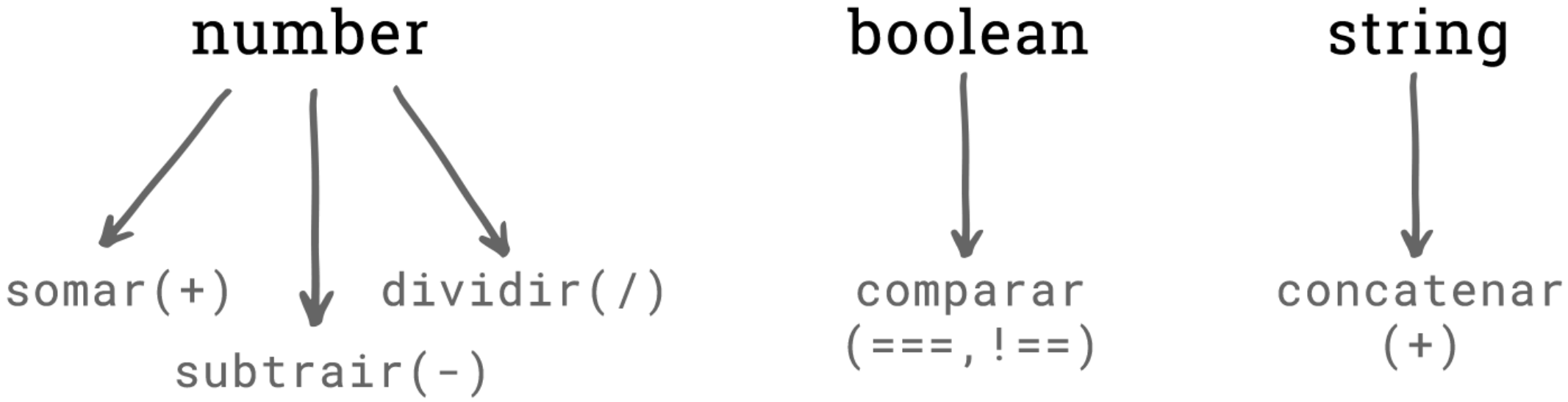
INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

- Ao utilizar **var** conseguimos redeclarar uma mesma variável.
- Com o padrão ECMAScript 6 (ES6) **let** e **const** são utilizados para declaração de variáveis.
- **let** é, agora, a forma preferida de declaração de variáveis.

Tipos de dados: string, number e boolean

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Certas operações só fazem sentido para um certo tipo de dado.



Tipos de dados: string, number e boolean

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Cada tipo de dado possui certas funções nativas, que nos ajudam a manipular os seus valores.

Como saber o tipo de uma variável?

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

O tipo de uma variável é determinado pelo seu valor:

- Todo número é do tipo number.
- Todo caractere dentro de aspas (simples ou duplas) é do tipo string.
- Valores true e false são do tipo booleano.

Como saber o tipo de uma variável?

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Para descobrir o tipo de uma variável sem ser pelo seu valor, basta utilizar '**typeof**' antes do nome da variável para imprimir no terminal.

```
1  var valor = 4;
2  console.log(typeof valor);
3  // Vai imprimir 'number'
4
5  var sobrenome = "Silva";
6  console.log(typeof sobrenome);
7  // Vai imprimir 'string'
8
9  var ligado = true;
10 console.log(typeof ligado);
11 // Vai imprimir 'boolean'
```

Tipos de dados: array

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

O array é uma coleção de dados e com esse recurso podemos colocar mais de um valor em apenas uma variável.

Podemos declarar um array da seguinte forma:

```
1 | var estados = ["Rio de Janeiro", "São Paulo", "Bahia"];
```

Tipos de dados: array

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Entendendo o código:

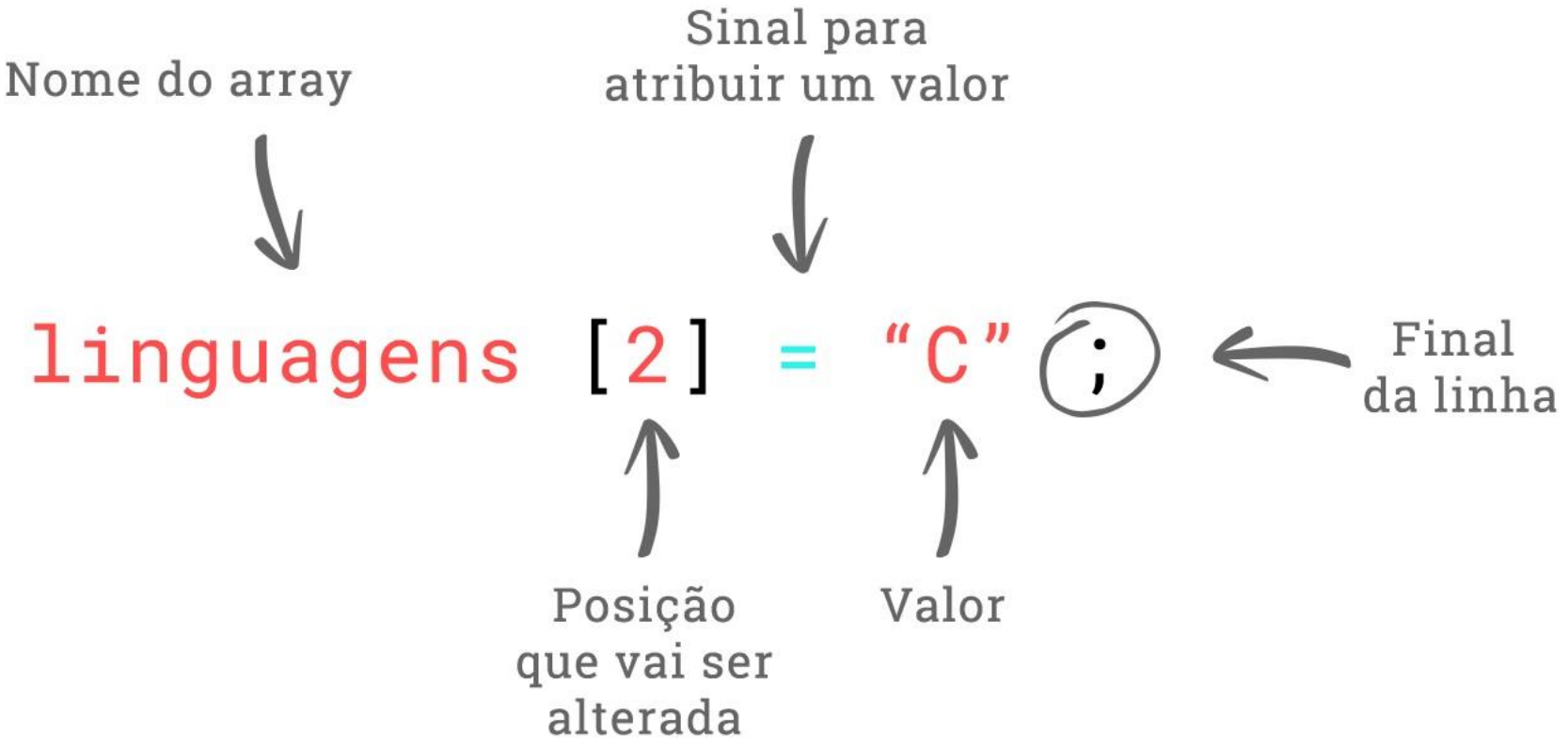
Acessando um valor do array

Em um array, cada variável está numa posição específica, que é representada por um índice numérico. Sendo assim, para acessar um valor específico usamos o índice da posição em que esse valor está.

Todo array começa com o índice 0, portanto para acessarmos o seu primeiro valor utilizamos **nome_do_array[0]**

Alterando um valor no array

```
1 | var linguagens = ["JavaScript", "PHP", "JAVA", "C#"];
2 | linguagens[2] = "C";
```

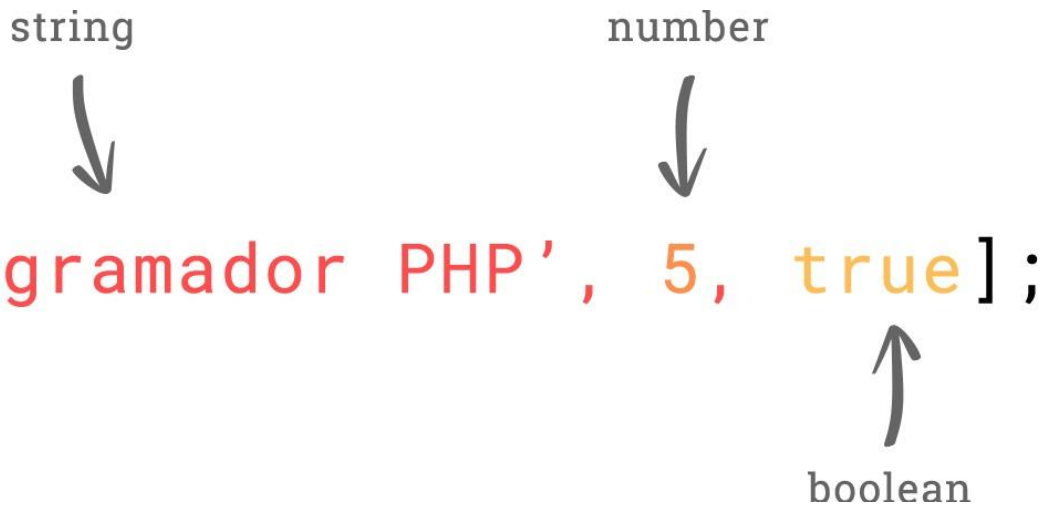


Armazenando tipos diferentes

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Um array pode ser usado para armazenar tipos de dados diferentes.

```
1 | var titulo = "Programador";
2 | var quantidadeDeVagas = 5;
3 | var vagaAtiva = true;
```



Contando a quantidade de elementos

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Existem casos em que precisamos saber quantos elementos tem dentro do array. Isso é feito utilizando .length depois do nome do array.

```
1  var linguagens = ["JavaScript", "PHP", "JAVA", "C#"];
2
3  console.log(linguagens.length);
4  // Vai imprimir 4
```

```
var telefone1 = "(22)3455-8819";
var telefone2 = "(11)98899-8787";
```

Quando usar arrays?

Digamos que um usuário possui dois telefones. Como armazenar esses dados? Uma primeira ideia seria usar duas variáveis.

Quando usar arrays?

Mas e se agora ele tiver três telefones? Precisaríamos mudar nosso código-fonte criando uma terceira variável, o que não é uma solução elegante.

O uso de array nesse caso é perfeito, veja:

```
1 | var telefones = [
2 |     '(11) 98899 - 8787',
3 |     '(22) 3455 - 8819',
4 |     '(91) 95620 - 0000'
5 | ];
```

Com o array conseguimos armazenar em uma variável mais de um valor em comum.

Tipos de dados: undefined

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

A propriedade indefinida indica que uma variável não recebeu um valor ou não foi declarada.

```
1 | var nome;
2 | console.log(nome); //vai imprimir undefined
```

Digamos que alguém tente contar quantas letras essa variável possui, presumindo que o seu tipo é string.

```
1 | console.log(nome.length);
```

Tipos de dados: undefined

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Uma variável undefined não é uma string e não possui a propriedade length, o que vai gerar um erro:

```
1 | TypeError: Cannot read property 'length' of undefined
```

Uma das formas de resolver esse erro é inicializando a variável.

```
1 | var nome = '';
```

Tipos de dados: undefined

INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

Um outro problema acontece quando você faz uma operação matemática com um valor undefined:

```
1 | var idade;
2 |
3 | console.log( idade + 1 );
```

O resultado será **NaN** (Not a Number), não é um número.

Esse erro também pode ser evitado se atribuímos um valor ao criar a variável.

```
1 | var idade = 0;
```

Tipos de dados: null

INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

É possível iniciar uma variável com **null**, o que significa que queremos adiar intencionalmente ou não atribuir um valor a ela.

```
1 | var nome  = null;
2 | console.log(nome);
3 | // vai imprimir null
```

Se imprimirmos esta variável teremos o valor null.

Tipos de dados: null

INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

Devemos ter cuidado ao lidar com uma variável cujo valor é null.

Se tentarmos acessar algum método ou atributo de uma variável cujo valor é null, um erro irá ocorrer:

```
1 | var nome = null;
2 | console.log(nome.length);
3 |
4 | TypeError: Cannot read property 'length' of null
```

Quando fazemos um cálculo e uma das variáveis utilizadas é null, o valor da mesma será convertido para 0.

Declarando uma constante

Algumas coisas mudam... outras não.

No código de uma aplicação é fácil encontrar valores que nunca devem mudar. Uma url, PI, um percentual de desconto, etc.

É uma boa prática declarar esses valores utilizando a palavra-chave **const**.

Uma vez definido esse valor não conseguimos mais alterar e se isso for feito também vai gerar um erro.

```
1 | const aula = "JavaScript";
2 | aula = "JS";
3 |
4 | TypeError: Assignment to constant variable.
```

Declarando uma constante

INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

Uma das vantagens do uso de **const** é o conceito de **imutabilidade**, que quer dizer "manter o mesmo valor" ou "não mudar“.

Por que é bom que as coisas não mudem?
Imagine que você declarou uma variável no início do seu código e foi usá-la muitas linhas abaixo. Como ter certeza que ela não foi alterada sem querer?

Utilizando **const** teremos a certeza de que isto não vai acontecer.

Exercícios

INSTITUTO FEDERAL
Rondônia
Campus
Vilhena

1. Escreva um programa que declare duas variáveis, “nome” e “idade”, e as imprima em um console em uma frase que diga “Olá, meu nome é [nome] e eu tenho [idade] anos”.
2. Declare uma string e utilize métodos para converter para maiúsculas e minúsculas.
3. Declare uma variável e verifique se o tipo dela é number.
4. Calcule o Índice de Massa Corporal (IMC) utilizando variáveis para altura e peso.

Para Nerds

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

MDN Web Docs – Java Script
https://developer.mozilla.org/pt-BR/docs/Web/JavaScript

JavaScript Tutorial
https://www.w3schools.com/js/

Cursos Gratuitos

INSTITUTO FEDERAL
Rondônia
Campus Vilhena

Curso JavaScript Curso em Video
https://www.cursoemvideo.com/curso/javascript/

Curso JavaScript YouTube
https://www.youtube.com/watch?v=McKNP3g6VBA