

Sprint 07 – Introdução ao JavaScript

Prof. Me. Gabriel Caixeta Silva

gabriel.silva@prof.sc.senac.br

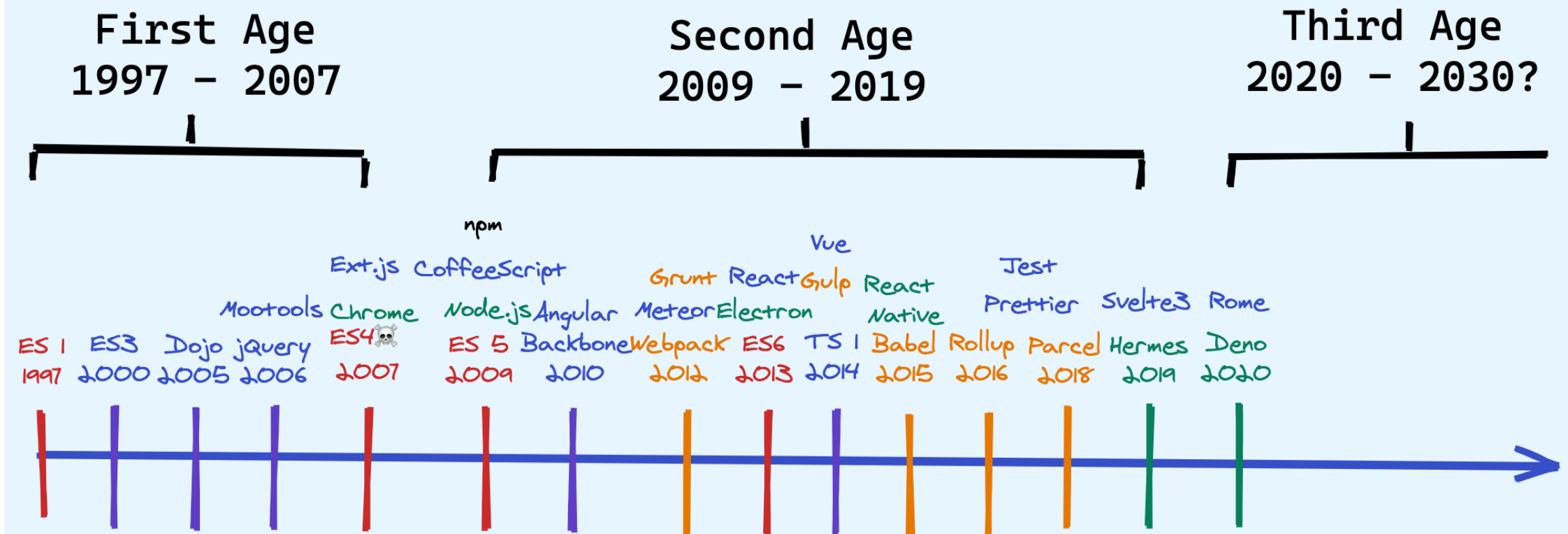
O que é / Para que serve o JavaScript?

JavaScript é uma linguagem de **programação interpretada** que apareceu pela primeira vez na década de 90, com o intuito de dar aos navegadores mais poder e interação a uma página web.

Nasceu assim o conceito de **páginas dinâmicas**.

Um pouco de história

JavaScript's Third Age



ECMAScript

ECMAScript é a **especificação** na qual o JavaScript se baseia.

Como especificação, significa que o ECMAScript é um **modelo** ao qual os motores JavaScript (implementações) devem aderir.

Um **motor JavaScript** é basicamente um programa que lê código JavaScript e roda ele.

Alguns motores javascript?

V8: Usado pelo navegador Google Chrome, Opera e Node.js. Suporta ES6 e recursos do ES7 e ES8.

SpiderMonkey: Usado pelo navegador Firefox e Adobe Acrobat. Suporta ES5.1, recursos do ES6, ES7 e ES8.

WebKit: Usado pelo navegador Apple Safari. Suporta ES6 e recursos do ES7 e ES8.

Chakra: Usado pelo navegador Microsoft Edge. Suporta ES5.1 e recursos do ES6, além de alguns recursos do ES7.

Por que isso é importante?

Em algumas situações alguns recursos do Javascript foram criados, modificados ou removidos em alguma versão do ECMAScript (ES5/ES5.1, ES6, ES7 ou ES8).

Alguns navegadores podem ou não suportar alguma versão do ECMAScript.

O [CanIuse](#) é um site bem interessante que permite ver a compatibilidade de uma funcionalidade do javascript em relação a diferentes browsers.

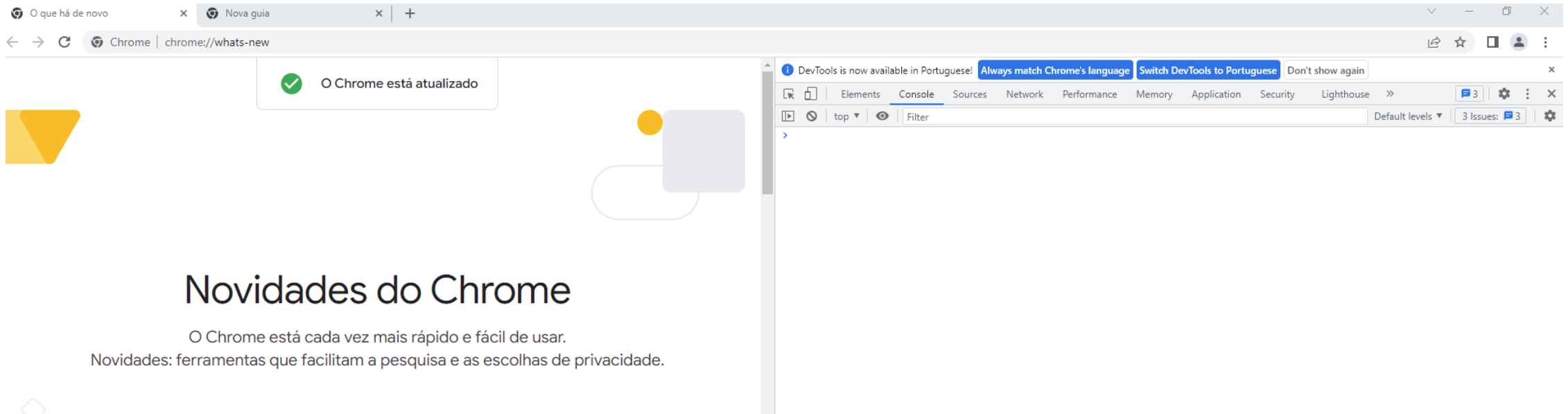
Dev tools for developers

Permite realizar uma infinidade de coisas:

- Inspecionar elementos HTML e CSS; 🔍
- Ver o desempenho de uma página web;
- Ver as requisições entre frontend e backend.
- Ver erros nas páginas;
- Testar comandos javascript;
- Depurar código para entender um bug 🐛

Dev tools for developers

Para abrir o **Dev tools** em qualquer navegador use a tecla **F12** .



Funções de saída

Funções de Console

```
console.log("Isso é um texto");  
  
console.log(123456);  
  
console.log([1, 2, 3]);  
  
console.log({ sprint: "Javascript" });
```

Funções de saída



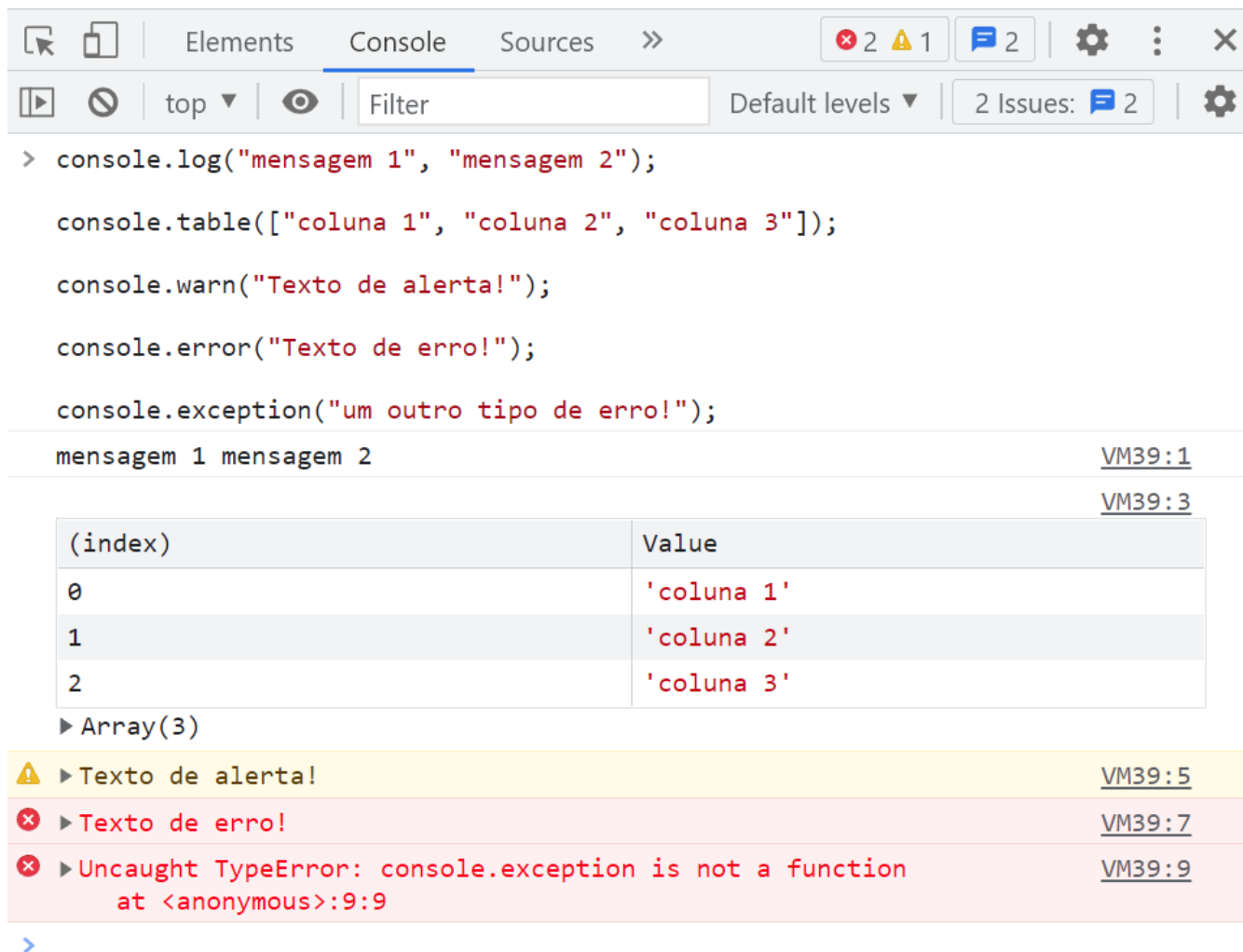
The image shows a web browser's developer console with the 'Console' tab selected. The console displays four log statements and their corresponding outputs. The first log statement is `console.log("Isso é um texto");`, which outputs the string `Isso é um texto`. The second log statement is `console.log(123456);`, which outputs the number `123456`. The third log statement is `console.log([1, 2, 3]);`, which outputs an array `(3) [1, 2, 3]`. The fourth log statement is `console.log({ sprint: "Javascript" });`, which outputs an object `{sprint: 'Javascript'}`. The console also shows `< undefined` and a blue prompt character `>` at the bottom.

```
> console.log("Isso é um texto");  
  
console.log(123456);  
  
console.log([1, 2, 3]);  
  
console.log({ sprint: "Javascript" });  
Isso é um texto  
123456  
▶ (3) [1, 2, 3]  
▶ {sprint: 'Javascript'}  
< undefined  
>
```

Tipos funções de saída

```
console.log("mensagem 1", "mensagem 2");  
console.table(["coluna 1", "coluna 2", "coluna 3"]);  
console.warn("Texto de alerta!");  
console.error("Texto de erro!");  
console.exception("um outro tipo de erro!");
```

Funções de saída



The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays the following JavaScript code and its output:

```
> console.log("mensagem 1", "mensagem 2");  
  
console.table(["coluna 1", "coluna 2", "coluna 3"]);  
  
console.warn("Texto de alerta!");  
  
console.error("Texto de erro!");  
  
console.exception("um outro tipo de erro!");
```

The output of the first line is "mensagem 1 mensagem 2" (VM39:1). The output of the second line is a table (VM39:3):

(index)	Value
0	'coluna 1'
1	'coluna 2'
2	'coluna 3'

Below the table, it says "► Array(3)".

The console also shows three error messages:

- Warning: "► Texto de alerta!" (VM39:5)
- Error: "► Texto de erro!" (VM39:7)
- Uncaught TypeError: "console.exception is not a function" (VM39:9)
at <anonymous>:9:9

A blue arrow at the bottom indicates that the console log can be expanded.

Comentários

Comentários em linha única

```
// isso é um comentário  
console.log("isso não é um comentário");  
// outra linha comentada
```

Comentários em bloco

```
/* isso é um comentário  
   com várias  
       linhas */  
console.log("isso não é um comentário");
```

Variáveis

```
variavel_1 = "valor"; // cuidado com o hoisting, use com parcimônia  
  
var variavel_2 = "outro valor"; // cuidado com o hoisting, use com parcimônia  
  
let variavel_3 = [123456]; // usa o escopo de bloco  
  
const constante = 123; // usa escopo de bloco e não pode ser alterada.
```

Leia o [artigo](#) da Alura para complementar a explicação.

Tipos de dados

Conceito de tipagem fraca

Trata quando o `tipo do dado não é bem definido`, a linguagem pode alterar o tipo por alguma ação, sem intervenção direta do programador, como é o caso da linguagem javascript.

Tipos de dados

Exemplo

```
const numberOne = "5";  
  
const numberTwo = 5;  
  
console.log(numberOne + numberTwo);  
  
// 55
```

Fonte: dev.to

Tipos de dados

Conversão de tipos

Font: [devheroes](#)