
Facens

Programação para Web
JavaScript

Prof.º Me Edson Martin Feitosa
Adapt. Prof. Especilista Fábio Colombini

Facens

Tópicos da aula

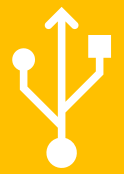
- Breve história do JS
- Por que aprender JS?
- Configurações do Ambiente
- Variáveis em JS
  - Como declarar
  - Tipos de variáveis
- Funções
- Operadores
  - Matemáticos
  - Lógicos
  - Atribuição
- Estruturas condicionais
- Estrutura de repetição (Controle)

Facens

O que é JavaScript?

JavaScript é uma linguagem de programação de alto nível, interpretada, multi-paradigma.

Criada por Brendan Eich em 1995, inicialmente para adicionar interatividade a páginas web.



Hoje, é amplamente utilizada para desenvolvimento front-end, back-end e mobile.

História do JavaScript

- Criado pela Netscape em 1995 como uma linguagem de script para páginas web.
- Rapidamente adotado por outros navegadores, incluindo Internet Explorer e Firefox.
- Padrão ECMAScript: padronização da linguagem pela ECMA International.

Por que aprender JavaScript?

- JavaScript é a linguagem de programação mais popular do mundo, de acordo com pesquisas como o Stack Overflow Developer Survey.
- É essencial para o desenvolvimento web moderno, permitindo criar páginas e aplicativos web interativos e dinâmicos.
- Amplamente utilizado em uma variedade de tecnologias, incluindo frameworks como React, Angular e Vue.js.

Facens

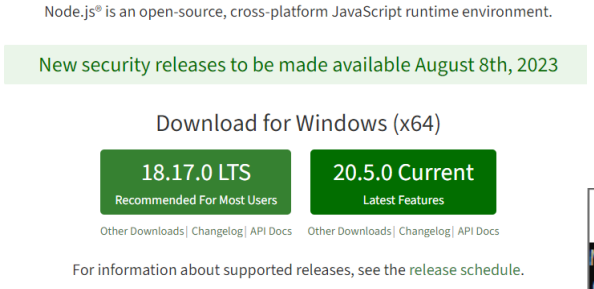
Configuração do Ambiente

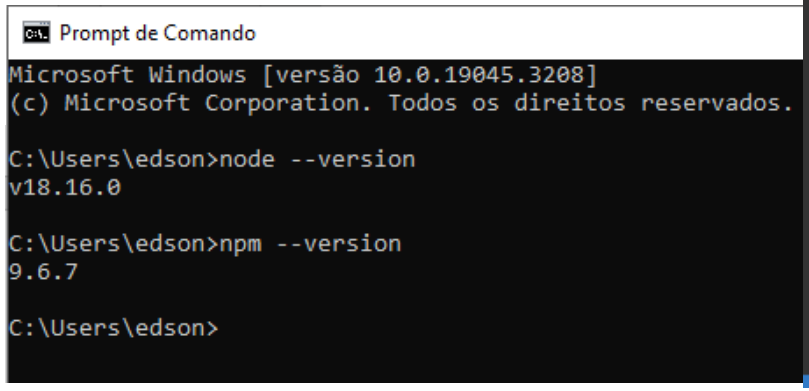
- Ambiente de Desenvolvimento: editores de código como Visual Studio Code, Sublime Text, etc.
- Navegadores: Chrome, Firefox, Safari, etc., que fornecem ferramentas de desenvolvimento integradas (DevTools – F12).
- Servidores de Desenvolvimento Locais: Node.js, Apache, etc.

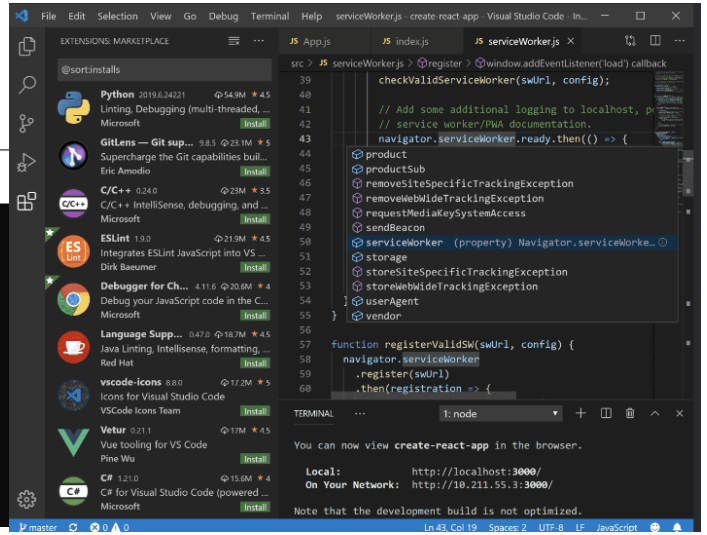
Facens

Configuração do Ambiente

- Visual Studio Code - https://code.visualstudio.com/
- Node - https://nodejs.org/en
- Facilitadores
  - NVM (Node Version Manager) - https://github.com/nvm-sh/nvm







Facens

Variáveis em Javascript: const, var e let

- Fracamente tipada
  - Const – Constante
  - Var e let – Variáveis
    - var eu posso declarar a mesma variável mais de uma vez e let não.
    - var tem escopo de function
      - Isso significa que elas são acessíveis em toda a função onde foram declaradas, mesmo que tenham sido declaradas dentro de blocos de código (como loops ou condicionais)
    - let tem escopo de bloco “{“.
      - elas são acessíveis apenas dentro do bloco em que foram declaradas, seja um bloco de função, um loop for, um bloco if, etc.
    - Hoisting: Variáveis declaradas com var são "elevadas" para o topo do seu escopo, o que significa que podem ser acessadas antes de serem declaradas no código.
    - Recomenda-se usar let em vez de var na maioria dos casos, pois let é mais seguro e previsível.

Facens

Variáveis em Javascript: const, var e let

- Fracamente tipada
  - O tipo da constante/variável é definida conforme a inicialização.

```
const a = 0
var x;
let y
z = 1
```

```
console.log(a)
console.log(x)
console.log(y)
console.log(z)
```

```
//tipos de variáveis
t = 'a'
console.log(typeof t)
t = 1
console.log(typeof t)
t = 1.12
console.log(typeof t)
```

  - Cuidado com undefined e infinity e NaN
    - Quando uma variável não tem valor no javascript não gera erro, apresenta o tipo **undefined**
    - Uma divisão por zero geraria erro em outras linguagem, mas o javascript retorna o tipo **infinity.** Ex.: console.log(3 / 0)
    - NaN – Not a number

Facens

```
function exemploVar() {
    if (true) {
        var mensagem = "Variável com escopo de função";
    }
    console.log(mensagem); // A variável 'mensagem' é acessível aqui
}

exemploVar(); // Saída: Variável com escopo de função
console.log(mensagem); // Erro: 'mensagem' is not defined
```

```
function exemploLet() {
    if (true) {
        let mensagem = "Variável com escopo de bloco";
    }
    console.log(mensagem); // Erro: 'mensagem' is not defined
}

exemploLet();
```

Facens

Tipos de variáveis

- Number (Número):
  - Usado para armazenar valores numéricos, sejam inteiros ou de ponto flutuante.
  - exemplo: let idade = 25;

- String (Texto):
  - Usado para armazenar sequências de caracteres.
  - exemplo: let nome = "Fábio";

- Boolean:
  - Usado para armazenar valores lógicos verdadeiro/falso.
  - exemplo: let estaChovendo = true;

Facens

Tipos de variáveis

- Array (Matriz): Usado para armazenar uma coleção ordenada de valores.
  - exemplo: let numeros = [1, 2, 3, 4, 5];

- Object (Objeto): Usado para armazenar uma coleção de pares chave/valor.
  - exemplo:

```
let pessoa = {
    nome: "Fábio",
    idade: 25,
    estaEstudando: true
};
```

- Undefined:
  - Usado para representar uma variável que foi declarada, mas ainda não foi atribuída a um valor.
  - exemplo: let endereco;

- Null:
  - Usado para representar a ausência intencional de valor.
  - exemplo: let dataDeNascimento = null;

Facens

Trabalhando com number

```
const peso1 = 1.0;
const peso2 = 2.1;

console.log(peso1);
console.log(Number.isInteger(peso1));

console.log(peso2);
console.log(Number.isInteger(peso2));

const avaliacao1 = 9.45;
const avaliacao2 = 6.456;
const total = avaliacao1 * peso1 + avaliacao2 * peso2;
const media = total / (peso1 + peso2);

console.log(media.toFixed(1)); //toFixed fixa a quantidade de casas após a vírgula
console.log(media.toString());
console.log(typeof media);

//number com n minúsculo é um tipo e com N maiúsculo é uma função
console.log(typeof Number);
```

Facens

Trabalhando com string

```
const nome = “Fábio“;

console.log(nome.charAt(3));
console.log(nome.charCodeAt(3));
console.log(nome.replace(“bi", "XX"));
console.log(nome.length);
console.log(nome.endsWith("n"));
console.log(nome.indexOf("d"));
console.log(nome.substring(0, 3));
console.log(nome.toLocaleUpperCase());
console.log("Ana,Maria,José".split(","));
console.log(nome.concat(‘ Rodrigo '.concat(Colombini’)));
if (nome.includes(“i”)) { }
```

Facens

TemplateString

```
//Concatenação de strings
let nome = “Fábio"
let concatenacao = "Olá "+ nome + "!"
console.log(concatenacao)

//template string
let template = `
    Olá
    ${nome}!`
console.log(template)

//expressões
console.log(`1 + 1 = ${1 + 1}`)

//string up(string texto){ return texto.toUpperCase();}
const up = texto => texto.toUpperCase();

console.log(`Ei! ${up('Cuidado')}!`)
```

Facens

Tipo booleano

```
let isAtivo = true;
console.log(isAtivo);

isAtivo = false;
console.log(isAtivo);
```

Facens

Tipo booleano

- Em JavaScript, **!!** é um **operador lógico duplo de negação**, comumente usado para converter um valor para seu equivalente booleano.
- é utilizado para garantir que um valor seja convertido para true ou false, independente do tipo de dado original.
- **!!** antes de uma expressão, o primeiro ! nega o valor e o segundo ! nega novamente, resultando em uma conversão direta para um valor booleano.

- Valores que geram **False**:
  - false, 0, null, undefined, NaN, ou uma string vazia "“
- Valores que geram **True**:
  - Se o valor original for qualquer outra coisa, ele será considerado “true"

Facens

Tipo booleano

```
console.log("os verdadeiros...");

console.log(!!isAtivo);
console.log(!!3)
console.log(!!-1)
console.log(!!' ')
console.log(!![])
console.log(!!{})
console.log(!!(isAtivo = true))
console.log(!!Infinity)
```

```
console.log('os falsos...’)

console.log(!!0)
console.log(!!'')
console.log(!!null)
console.log(!!NaN)
console.log(!!undefined)
console.log(!!(isAtivo = false))

console.log('para finalizar...')
console.log(!!('' || null || 0 || ' ‘))

//forma de usar expressão lógica para definir valor padrão
let nome = ''
console.log(nome || 'Desconhecido') //numa expressão lógica o javascript pega o 1º valor válido
```

Facens

Funções em JavaScript

- Declaração de Funções: function.
- Parâmetros e Argumentos.
- Retorno de Funções.

Facens

Funções em JavaScript

- Sem Parâmetros

```
function saudacao() {
    console.log("Olá! Bem-vindo à nossa aula de JavaScript!");
}

// Chamando a função
saudacao();
```

- Função com Parâmetros e sem Retorno:

```
function soma(a, b) {
    let resultado = a + b;
    console.log("A soma de", a, "e", b, "é:", resultado);
}

// Chamando a função
soma(3, 5); // Saída: A soma de 3 e 5 é: 8
```

Facens

Funções em JavaScript

- Função com Parâmetros e com Retorno:

```
function calcularAreaRetangulo(largura, altura) {
    let area = largura * altura;
    return area;
}

// Chamando a função e armazenando o resultado em uma variável
let areaRetangulo = calcularAreaRetangulo(4, 6);
console.log("A área do retângulo é:", areaRetangulo); // Saída: A área do retângulo
```

Facens

Funções em JavaScript

- Função Anônima:

```
let dobrar = function(numero) {
    return numero * 2;
}

// Chamando a função anônima
console.log(dobrar(5)); // Saída: 10
```

- Arrow Function:

```
let triplo = (numero) => {
    return numero * 3;
}

// Chamando a arrow function
console.log(triplo(4)); // Saída: 12
```

Facens

Operadores em JavaScript

- Operadores Aritméticos: +, -, *, /, % (resto ou módulo).
- Operadores de Atribuição: =, +=, -=, *=, /=.
- Operadores de Comparação: ==, ===, !=, !==, >, <, >=, <=.
  - **== (igualdade abstrata):**
    - Este operador compara dois valores após a conversão de tipos (coerção de tipo)
    - 1 == ‘1’ // true
    - Use == se você estiver interessado apenas no valor das variáveis e desejar que JavaScript faça a conversão de tipos.
  - **=== (igualdade estrita):**
    - Este operador compara os valores sem fazer a conversão de tipos. Ele verifica se os valores são idênticos tanto em valor quanto em tipo. Por exemplo,
    - 1 === ‘1’ // false
    - Use === se você quiser garantir que os valores e os tipos das variáveis sejam idênticos, sem a conversão de tipos.

Facens

Estruturas de Controle

- Estruturas Condicionais: if, else, else if, switch.
- Estruturas de Repetição: while, do...while, for.

Facens

Entrada do usuário

- let num = window.prompt("Digite um número: ");

Facens