Módulos (import e export)

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

AGENDA

- Módulos (import e export)

MÓDULOS: Porque estudar?

Arquivos grandes e com muita responsabilidade?

Alguma vez você já precisou escrever **tantas funções** em um arquivo que dificultou achar o que queria?

Ou já percebeu que tinham várias funções no mesmo arquivo que **não tinha nada haver umas com as outras**?

index.js

```
function soma(numero1, numero2) {
  return numero1 + numero2
}

function subtracao(numero1, numero2) {
  return numero1 - numero2
}

function multiplicacao(numero1, numero2) {
  return numero1 * numero2
}

function divisao(numero1, numero2) {
  return numero1 / numero2
}

function cumprimenta() {
  console.log('Bom dia!')
}

function alertaUsuario(nome) {
  alert(`Cuidado ${nome}!!!`)
}
```

MÓDULOS: Porque estudar?

Solução = Módulos

Os módulos nos permitem escrever **funções em diferentes arquivos** e comunicá-las, exportando e importando-as em arquivos diferentes.

Benefícios:

- Quebra o código em porções **menores**;
- Nos permite organizar o programa em pastas e arquivos que acomodam códigos semelhantes
- Faz com que o código fique mais **fácil** de compreender e de alterar
- Facilita encontrar e corrigir **bugs**
- **Desacoplamento**: no caso de alterações de lógica ou especificação não será necessário alterar todo o código, apenas alguns trechos.

calculos.js

```
function soma(numero1, numero2) {
  return numero1 + numero2
}

function subtracao(numero1, numero2) {
  return numero1 - numero2
}

function multiplicacao(numero1, numero2) {
  return numero1 * numero2
}

function divisao(numero1, numero2) {
  return numero1 / numero2
}
```

comunicacoes.js

```
function cumprimenta() {
  console.log('Bom dia!')
}

function alertaUsuario(nome) {
  alert(`Cuidado ${nome}!!!`)
}
```

ANTES DO ES6

O sistema de módulos não fazia parte da linguagem JS,
por isso eram utilizadas ferramentas alternativas,
sendo a principal: **CommonJS**

SINTAXE

funcoes.js

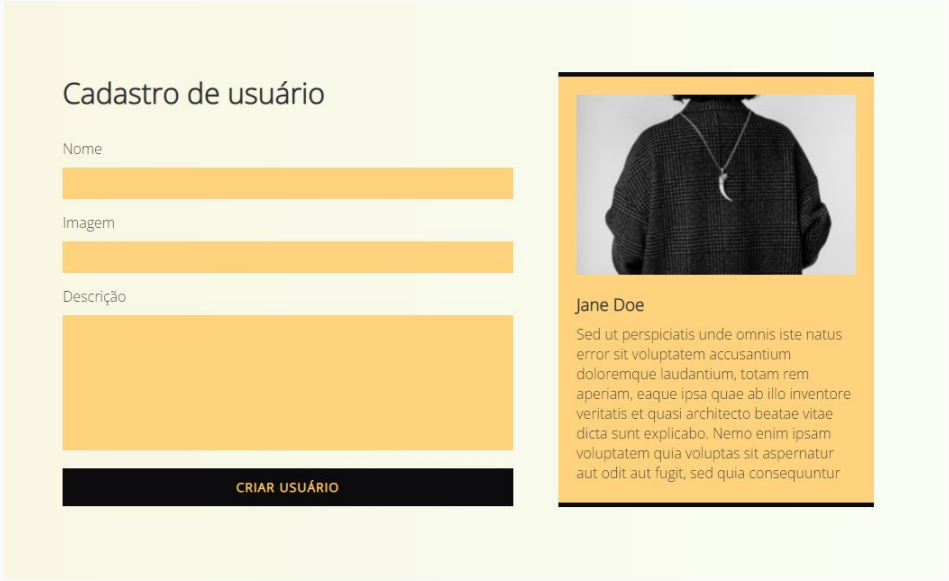
```
// Exemplos de EXPORTAÇÃO
function funcaoB() {
  console.log("Sou a função B");
}

funcaoC = function() {
  console.log("Sou a função C");
}

// === EXPORTANDO APENAS UMA FUNÇÃO
module.exports = funcaoB;

// === EXPORTANDO MAIS DE UMA FUNÇÃO
module.exports = {
  funcaoB,
  funcaoC
}
// ou
module.exports = funcaoB;
module.exports = funcaoC;
```

index.js

```
// Exemplos de IMPORTAÇÃO

// IMPORTANDO APENAS UMA FUNÇÃO
const funcaoB = require('./funcoes.js')

// IMPORTANDO MAIS DE UMA FUNÇÃO
const { funcaoB, funcaoC } = require('./funcoes.js')
// ou
const funcaoB = require('./funcoes.js')
const funcaoC = require('./funcoes.js')


// USANDO AS FUNÇOES IMPORTADA
function funcao() {
    console.log("Eu sou uma função");
    funcaoB();
    funcaoC();
}
funcao()
```

Exercício 1

Crie uma aplicação com três inputs que receba um nome,
link de imagem e descrição de um usuário.
Fazendo uso de módulos, exiba os valores recebidos em um card de usuário.



INTERVALO DE AULA

DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20
Retorno: 20:40

DEPOIS DO ES6

O sistema de módulos passou a ser **nativo** no JS,
(é suportado em quase todos os browsers e pelo Node)

TYPE MODULE

Para utilizar os módulos nativos será necessário:

Browser: incluir o atributo **“type”: “module”** dentro da tag **script**

index.html depoisES6 U

semana-05 > aula-04 > depoisES6 > index.html > html > head > script

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta http-equiv="X-UA-Compatible" content="IE=edge">
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <!-- ACRESCENTE PARA QUE O IMPORT E EXPORT NATIVO SEJA SUPORTADO-->
8    <script src="index.js" type="module"></script>
9    <script src="funcoes.js" type="module"></script>
```

Node: criar um arquivo **package.json** no projeto e incluir o atributo **“type”: “module”**

package.json depoisES6 1, U

semana-05 > aula-04 > depoisES6 > package.json > type

```
1  {
2    "name": "depoisES6",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
       ▷ Depurar
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC",
12   "type": "module"
13 }
14
```

SINTAXE

```
// EXEMPLOS DE EXPORTAÇÃO

// DEFAULT / PRINCIPAL
export default function funcaoB() {
  console.log("Sou a função B");
}

// COMUM
export function funcaoC() {
  console.log("Sou a função C");
}

export function funcaoD() {
  console.log("Sou a função D");
}

// ou
export { funcaoC, funcaoD };
```

```
// EXEMPLOS DE IMPORTAÇÃO

// default
import funcaoB from "./funcoes.js";

// comum
import { funcaoC, funcaoD } from "./funcoes.js";

// default + comum
import funcaoB, { funcaoC, funcaoD } from "./funcoes.js";


// USANDO AS FUNÇOES IMPORTADA
function funcaoA() {
    console.log("Executando função A");
    funcaoB();
    funcaoC();
    funcaoD();
}
funcaoA()
```

OBS: Podemos importar e exportar **variáveis**, **constantes**, **funções** e **classes**

OUTRAS FORMAS DE IMPORTAR

```
// IMPORTAR COM ALIAS/APELIDO
import { funcaoC as minhaFuncaoC } from './funcoes.js'
function funcao() {
    console.log("Eu sou uma função");
    minhaFuncaoC();
}
funcao()

// IMPORTAR TUDO *
import * as minhasFuncoes from './funcoes.js'

function funcao() {
    console.log("Eu sou uma função");
    minhasFuncoes.default();
    minhasFuncoes.funcaoC();
}
funcao()
```

```
// IMPORTAR O DEFAULT E TODO O RESTO COM O *
import funcaoB, * as minhasFuncoes from './funcoes.js'

function funcao() {
    console.log("Eu sou uma função");
    funcaoB();
    minhasFuncoes.funcaoC();
}
funcao()
```

Exercício 2

Em uma nova pasta, refatore o código do exercício 1 utilizando os módulos nativos do JS.

MATERIAL COMPLEMENTAR

- export - JavaScript | MDN
- import - JavaScript | MDN

AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique aqui ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.

DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!

SENAI

<LAB365>