Guia Prático: ES6+, Promises, Async/Await, Módulos e Classes

Este guia apresenta os recursos modernos do JavaScript introduzidos no ES6 e posteriores, incluindo exemplos práticos e exercícios. Conteúdo: 1. ES6+ (Novidades) 2. Promises 3. Async/Await 4. Módulos 5. Classes

1. ES6+

Principais recursos: - let e const - Arrow Functions - Template Strings - Destructuring - Spread Operator

Exemplo:

```
let nome = "João"; const idade = 30; console.log(`Meu nome é ${nome} e tenho ${idade} anos.`);
```

2. Promises

As Promises representam operações assíncronas.

```
const promessa = new Promise((resolve, reject) => { let sucesso = true; if(sucesso) resolve("Tudo certo!"); else reject("Erro!"); }); promessa.then(resultado => console.log(resultado)) .catch(erro => console.log(erro));
```

3. Async/Await

Forma simplificada de lidar com Promises.

```
async function buscarDados() { try { let resposta = await fetch("https://jsonplaceholder.typicode.com/posts/1"); let dados = await resposta.json(); console.log(dados); } catch(erro) { console.log("Erro:", erro); } } buscarDados();
```

4. Módulos

Permitem dividir o código em arquivos menores e reutilizáveis.

```
// arquivo1.js export function soma(a, b) { return a + b; } // arquivo2.js import { soma } from './arquivo1.js'; console.log(soma(2, 3));
```

5. Classes

Introduzem a sintaxe de Programação Orientada a Objetos no JS.

```
class Pessoa { constructor(nome, idade) { this.nome = nome; this.idade = idade; } apresentar() { console.log(`Olá, meu nome é ${this.nome} e tenho ${this.idade} anos.`); } } const p1 = new Pessoa("Ana", 28); p1.apresentar();
```

Desafios:

1. Reescreva uma função tradicional como Arrow Function. 2. Crie uma Promise que simula buscar dados após 2 segundos. 3. Use async/await para consumir a API "https://jsonplaceholder.typicode.com/users". 4. Organize funções em dois módulos e importe-as no arquivo principal. 5. Crie uma classe Carro com métodos ligar() e desligar().