

FULL STACK DEVELOPMENT
TYPESCRIPT

AULA 04

SUMÁRIO

O QUE VEM POR AÍ?	3
HANDS ON.....	4
SAIBA MAIS	5
O QUE VOCÊ VIU NESTA AULA?.....	9
REFERÊNCIAS	10

EMSE

O QUE VEM POR AÍ?

Mergulhe no universo dos Modules e Namespaces! Descubra como os Modules ajudam a organizar e encapsular seu código, mantendo-o isolado do escopo global. Também vamos entender o papel dos Namespaces em agrupar funcionalidades relacionadas, prevenindo conflitos de nomes. Com demonstrações práticas, esta aula será um guia para estruturar seu código de forma eficaz e inteligente. Prepare-se para uma aula enriquecedora e cheia de aprendizado!

EMANIP

HANDS ON

Nesta aula, os docentes introduzirão o conceito de Modules em TypeScript, explicando como os módulos contribuem para a organização e o encapsulamento do código. Por meio de um exemplo prático, demonstrarão como criar e exportar uma função em um arquivo TypeScript. Além disso, apresentarão a combinação eficiente de Modules e Namespaces em TypeScript. Esta parte da aula é essencial para compreender como eles possibilitam a construção de aplicações mais estruturadas e organizadas.

EMAN

SAIBA MAIS

Módulos e Namespaces

Utilizando módulos

Os módulos são estruturas que podem abrigar tanto códigos quanto declarações. Eles geralmente necessitam de um sistema para carregar módulos, como o CommonJs ou Require.js, ou então de um ambiente que já tenha suporte integrado para módulos ES. Eles oferecem vantagens como reutilização eficiente do código, maior isolamento e suporte aprimorado para ferramentas de empacotamento.

Implementando Namespaces

Os Namespaces são uma técnica específica do TypeScript para organizar o código. Eles funcionam como objetos JavaScript nomeados dentro do namespace global. Isso faz com que os Namespaces sejam bastante fáceis de implementar. Diferente dos módulos, eles podem se estender por vários arquivos e são uma alternativa eficaz para estruturar o código em aplicações web.

Agora, vamos simplificar ainda mais, focando em exemplos práticos e cotidianos para entender melhor os conceitos de Namespaces e módulos no TypeScript.

Namespaces

Imagine que você está organizando uma gaveta de ferramentas. Você decide separar as ferramentas por tipo: chaves de fenda, martelos, alicates, etc. Cada seção da gaveta é como um Namespace. Dentro de cada seção, as ferramentas (classes, funções) são facilmente identificáveis e organizadas.



Figura 1 – Ferramentas

Fonte: Freepik (s.d.)

A seguir, temos um exemplo simplificado de Namespace:

```
namespace Ferramentas {  
  export class Martelo {  
    // Detalhes do martelo  
  }  
  
  export class ChaveDeFenda {  
    // Detalhes da chave de fenda  
  }  
}  
  
let meuMartelo = new Ferramentas.Martelo();
```

Aqui, Martelo e ChaveDeFenda são classes dentro do Namespace Ferramentas. Para usar Martelo, você precisa especificar que está se referindo ao Martelo dentro de Ferramentas.

Módulos

Agora, pense em módulos como livros individuais em uma biblioteca. Cada livro (módulo) é autocontido e trata de um assunto específico. Se você precisa de informações sobre um tópico, você pega o livro correspondente. Em TypeScript, cada arquivo é um módulo e pode conter suas próprias classes, interfaces, funções, etc.



Figura 2 – Livros
Fonte: Freepik (s.d.)

Exemplo simplificado de Módulo:

```
export class Martelo {  
  // Detalhes do martelo  
}
```

Arquivo: UsarMartelo.ts

```
import { Martelo } from "../Martelo";  
  
let meuMartelo = new Martelo();
```

Aqui, Martelo é definido em seu próprio arquivo, Martelo.ts. Se você quiser usar a classe Martelo em outro arquivo, como UsarMartelo.ts, você deve importá-la.

Diferenças chave

Namespaces: são como seções em uma gaveta, organizando diferentes classes/funções sob um nome comum.

Módulos: são como livros individuais em uma biblioteca. Cada um é separado e contém informações específicas. Eles podem compartilhar informações usando importação/exportação.

Quando usar

Namespaces: quando você tem muitas classes ou funções relacionadas que deseja agrupar sob um nome comum e elas estão no mesmo contexto geral.

Módulos: quando você deseja manter seu código separado em arquivos distintos, facilitando a manutenção, o teste e o gerenciamento de dependências.

O QUE VOCÊ VIU NESTA AULA?

Nesta aula, focamos em dois conceitos de TypeScript: módulos e Namespaces. Ambos desempenham papéis importantes na organização e estruturação de códigos, especialmente em projetos de grande escala. Aprofundamos no entendimento de como os módulos servem como ferramentas essenciais para gerenciar e encapsular códigos de forma eficaz, permitindo uma separação clara e uma melhor manutenção. Além disso, discutimos a importância dos Namespaces no TypeScript, uma técnica que facilita a organização do código ao agrupar classes, interfaces e outras entidades sob um nome comum.

REFERÊNCIAS

Geeks for geeks. **What is Namespace in Typescript?** Disponível em: <<https://www.geeksforgeeks.org/what-is-namespace-in-typescript/>>. Acesso em: 29 jan. 2024.

TypeScript. **Documentação Modules.** Disponível em: <<https://www.typescriptlang.org/pt/docs/handbook/modules.html>>. Acesso em: 29 jan. 2024.

TypeScript. **Documentação Namespaces.** Disponível em: <<https://www.typescriptlang.org/docs/handbook/namespaces.html>>. Acesso em: 29 jan. 2024.

PALAVRAS-CHAVE

Palavras-chave: Typescript. Desenvolvimento Básico. Superset.

EMSE

POS TECH