3000 TALENTOS

interfaces & encapsulamento

DESENVOLVIMENTO BACKEND COM TYPESCRIPT

```typescript
interface OldModel {
  [key: string]: any;
}
interface NewModel {
 [key: string]: any;
}
function convertModel(model: OldModel): NewModel {
  const newModel: NewModel = {
    id: model.id,
    label: model.label || model.name,
    properties: {
      ...model.properties,
      componentType: model.componentType
    },
    // Magic Tree Parser
    children: createChildren(model.children || [])
  };
  if (model.foo === 'bar') {
    return {...newModel, foo: model.foo };
  }
  return newModel;
}
```

PROF. ESP. IURI NASCIMENTO SANTOS

interfaces

abstração

- MÉTODOS OBRIGATÓRIOS
- UMA CLASSE IMPLEMENTA VÁRIAS INTERFACES (0 OU N)

SINTAXE:

```
interface UmaPessoa{
    nome: string;
    idade: number;
    documento: number;
    falar(): string;
    cantar(): string;
    cumprimentar(nomeOutro: string): string;
}
```

interfaces

abstração

E O QUE MUDA?

```
export class Funcionario implements UmaPessoa{
    nome: string;
    idade: number;
    documento: number;
    constructor(nome: string, idade: number, documento: number){
        this.nome = nome;
        this.idade = idade;
        this.documento = documento;
    }

    falar(): string{
        return `eu sou ${this.nome}`
    }

    cantar(): string{
        return `lala estou cantando`
    }

    cumprimentar(nomeOutro: string): string{
        return `olá ${nomeOutro}, sou ${this.nome}`
    }
}
```

encapsulamento

- **PUBLIC**: ACESSO PERMITIDO DE QUALQUER LUGAR. É O PADRÃO.
- **PRIVATE**: ACESSO RESTRITO APENAS À PRÓPRIA CLASSE. OUTROS OBJETOS NÃO PODEM ACESSAR DIRETAMENTE OS ATRIBUTOS/MÉTODOS PRIVADOS.
- **PROTECTED**: ACESSO PERMITIDO À PRÓPRIA CLASSE E SUAS SUBCLASSES. É ÚTIL PARA DEFINIR COMPORTAMENTOS COMPARTILHADOS COM AS SUBCLASSES.

encapsulamento

como não usar:

```
interface UmaPessoa{
    private nome: string;
    protected idade: number;
    protected documento: number;
    falar(): string;
    cantar(): string;
    cumprimentar(nomeOutro: string): string;
}
```

não funciona!

encapsulamento

como usar:

```
export class Funcionario implements UmaPessoa{
    public nome: string;
    public idade: number;
    public documento: number;
    constructor(nome: string, idade: number, documento: number){
        this.nome = nome;
        this.idade = idade;
        this.documento = documento;
    }

    public falar(): string{
        return `eu sou ${this.nome}`
    }

    public cantar(): string{
        return `lala estou cantando`
    }

    public cumprimentar(nomeOutro: string): string{
        return `olá ${nomeOutro}, sou ${this.nome}`
    }
}
```

funciona!