Pipes no Angular: Domine a Formatação e Transformação de Dados

Os pipes no Angular são ferramentas poderosas que permitem formatar e transformar dados exibidos no template HTML de seus componentes. Eles oferecem uma maneira flexível e declarativa de manipular dados antes de serem apresentados ao usuário, tornando seu código mais conciso e legível.

1. Conceitos Básicos:

- **O que são Pipes?** Os pipes são funções que transformam um valor em outro valor. Eles são aplicados a valores entre colchetes duplos `{{ }}` no template HTML.
- **Como Usar Pipes?** A sintaxe básica para usar um pipe é: `{{ valor | nomeDoPipe[:parâmetros] }}`.
- **Tipos de Pipes:** O Angular oferece diversos pipes integrados para formatação de datas, números, strings, moedas e muito mais. Você também pode criar seus próprios pipes personalizados.

2. Exemplos de Uso:

1. Formatando Datas:

- **Pipe Date:** `{{ data | date:'dd/MM/yyyy' }}` - Formata a data para o padrão dd/MM/yyyy.
- **Pipe DatePipe:** `{{ data | datePipe:'longDate' }}` - Formata a data com o nome completo do mês e ano.

2. Formatando Números:

- **Pipe Currency:** `{{ valor | currency:'BRL' }}` - Formata o valor como moeda brasileira (R$).

- **Pipe Number:** `{{ valor | number:'1.2-2' }}` - Formata o valor com duas casas decimais e separador de milhares.

3. Transformando Strings:

- **Pipe Uppercase:** `{{ texto | uppercase }}` - Converte o texto para maiúsculas.
- **Pipe Lowercase:** `{{ texto | lowercase }}` - Converte o texto para minúsculas.
- **Pipe Titlecase:** `{{ texto | titlecase }}` - Converte a primeira letra de cada palavra para maiúscula.

4. Pipes Aninhados:

É possível aninhar pipes para aplicar várias transformações em um mesmo valor. Por exemplo:

```
HTML
{{ valor | currency:'BRL' | number:'1.2-2' }}
```

5. Criando Pipes Personalizados:

Você pode criar seus próprios pipes para atender às suas necessidades específicas. Para isso, siga estes passos:

1. Crie uma classe que implemente a interface `PipeTransform`.
2. Implemente o método `transform` que recebe o valor a ser transformado e retorna o valor transformado.
3. Decore a classe com `@Pipe({ name: 'nomeDoPipe' })`.

6. Recursos Adicionais:

- **Documentação Oficial do Angular sobre Pipes:** [URL inválido removido]
- **Lista Completa de Pipes Integrados:** [URL inválido removido]

- **Tutorial sobre Criação de Pipes Personalizados:**

  https://www.freecodecamp.org/news/angular-for-beginners-course/

7. Exemplos Práticos:

Exemplo 1: Formatando uma Lista de Produtos:

HTML

```
<ul>
  <li *ngFor="let produto of produtos">
    {{ produto.nome }} - {{ produto.preco | currency:'BRL' }}
  </li>
</ul>
```

Exemplo 2: Exibindo uma Data com Hora Completa:

HTML

```
<p>Última atualização: {{ dataAtualizacao | date:'longDate' | time }}</p>
```

Exemplo 3: Criando um Pipe para Filtrar por Nome:

TypeScript

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'filtroNome'
})
export class FiltroNomePipe implements PipeTransform {
  transform(produtos: any[], nome: string): any[] {
    if (!nome.trim()) {
      return produtos;
    }

    return produtos.filter(produto =>
produto.nome.toLowerCase().includes(nome.toLowerCase()));
  }
}
```

Exemplo 4: Usando o Pipe Criado no Template:

HTML

```
<ul>
  <li *ngFor="let produto of produtos | filtroNome:'Camisa'">
    {{ produto.nome }} - {{ produto.preco | currency:'BRL' }}
  </li>
</ul>
```

Lembre-se: Os pipes são ferramentas valiosas para tornar seus templates Angular mais expressivos e fáceis de ler. Utilize-os com criatividade para formatar e transformar dados de forma eficiente e personalizada.