Resumo do Vídeo "Curso Angular #45: Aplicando Locale (internacionalização) nos Pipes"

Neste vídeo da série Curso Angular, Loiane Groner demonstra como aplicar internacionalização (locale) nos pipes do Angular 2 para formatar números e datas de acordo com a região do usuário.

Introdução

O vídeo começa com uma breve revisão do conceito de internacionalização no Angular 2 e como ele foi abordado em aulas anteriores. Em seguida, Loiane apresenta o objetivo da aula: mostrar como formatar números e datas de acordo com a localidade do usuário utilizando pipes.

Aplicando Locale nos Pipes

Para aplicar locale nos pipes, Loiane segue estes passos:

1. **Importar o token Locale:** O token `LOCALE_ID` é importado do pacote `@angular/core`. Este token é utilizado para fornecer ao Angular a localidade do usuário.
2. **Criar um serviço de configurações:** Um serviço é criado para armazenar as configurações de locale, como o idioma e o formato de data e hora.
3. **Fornecer o serviço de configurações no módulo raiz:** O serviço de configurações é fornecido no módulo raiz da aplicação para que ele possa ser acessado por todos os componentes.
4. **Utilizar o pipe `DatePipe` com locale:** O pipe `DatePipe` é utilizado para formatar datas de acordo com a localidade do usuário. Para isso, o token `LOCALE_ID` é injetado no pipe.

5. **Utilizar o pipe `CurrencyPipe` com locale:** O pipe `CurrencyPipe` é utilizado para formatar números como moedas de acordo com a localidade do usuário. Para isso, o token `LOCALE_ID` é injetado no pipe.

Exemplo Prático

Loiane demonstra como aplicar locale nos pipes em um exemplo prático. Ela cria um componente que exibe um número e uma data formatados de acordo com a localidade do usuário. O componente utiliza os pipes `DatePipe` e `CurrencyPipe` e injeta o token `LOCALE_ID` para obter a localidade do usuário.

Outras Maneiras de Aplicar Locale

Loiane também apresenta outras maneiras de aplicar locale nos pipes, como:

- Utilizando o método `setLocale` do serviço `PlatformLocation`.
- Utilizando a diretiva `locale` do Angular 2.

Conclusão

O vídeo conclui com um resumo dos tópicos abordados e um lembrete de que o código-fonte e o material de apoio da série estão disponíveis no site da autora. Loiane também incentiva os espectadores a deixarem seus comentários e feedback.

Recursos Adicionais

- Site da Loiane Groner: https://www.loiane.com/
- Repositório do curso no GitHub: https://github.com/loianegruner/curso-angular

Exemplos Práticos de Pipes no Angular: Dominando Formatação e Transformação de Dados

Os pipes no Angular são ferramentas poderosas que permitem formatar e transformar dados antes de serem exibidos na tela. Eles oferecem uma maneira flexível e declarativa de manipular dados, tornando seu código mais conciso e legível.

1. Formatando Datas:

Cenário: Exibir a data atual no formato dd/MM/yyyy.

Solução:

```
HTML
<p>Data atual: {{ dataAtual | date:'dd/MM/yyyy' }}</p>
```

Explicação:

- O pipe `date` formata a data `dataAtual` de acordo com o padrão especificado.
- No caso, o padrão `dd/MM/yyyy` representa dia, mês e ano com separadores de barra.

Exemplo Completo:

```
HTML
<p>Data atual: {{ dataAtual | date:'dd/MM/yyyy' }}</p>

<script>
  import { Component } from '@angular/core';

  @Component({
    selector: 'app-root',
    template: `
      <p>Data atual: {{ dataAtual | date:'dd/MM/yyyy' }}</p>
    `
  })
  export class AppComponent {
    dataAtual = new Date();
```

```
  }
</script>
```

2. Formatando Números:

Cenário: Exibir um valor monetário como real brasileiro (R$).

Solução:

```
HTML
<p>Valor: {{ valor | currency:'BRL' }}</p>
```

Explicação:

- O pipe `currency` formata o valor numérico `valor` como moeda.
- No caso, o código `'BRL'` indica que o formato desejado é o real brasileiro.

Exemplo Completo:

```
HTML
<p>Valor: {{ valor | currency:'BRL' }}</p>

<script>
  import { Component } from '@angular/core';

  @Component({
    selector: 'app-root',
    template: `
      <p>Valor: {{ valor | currency:'BRL' }}</p>
    `
  })
  export class AppComponent {
    valor = 1234.56;
  }
</script>
```

3. Transformando Strings:

Cenário: Converter um texto para maiúsculas.

Solução:

```
HTML
<p>Texto em maiúsculas: {{ texto | uppercase }}</p>
```

Explicação:

- O pipe `uppercase` converte o texto `texto` para maiúsculas.

Exemplo Completo:

```
HTML
<p>Texto em maiúsculas: {{ texto | uppercase }}</p>

<script>
  import { Component } from '@angular/core';

  @Component({
    selector: 'app-root',
    template: `
      <p>Texto original: {{ texto }}</p>
      <p>Texto em maiúsculas: {{ texto | uppercase }}</p>
    `
  })
  export class AppComponent {
    texto = 'texto original';
  }
</script>
```

4. Criando Pipes Personalizados:

Cenário: Filtrar uma lista de produtos por nome.

Solução:

1. Crie a classe do pipe:

```
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()));
  }
}
```

2. Utilize o pipe no template:

HTML

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

Explicação:

- A classe `FiltroNomePipe` implementa a interface `PipeTransform` do Angular.
- O método `transform` recebe a lista de produtos e o nome a ser filtrado, retornando a lista filtrada.
- No template, o pipe `filtroNome` é utilizado para filtrar a lista de produtos `produtos` pelo nome `Camisa`.

5. Pipes Aninhados:

Cenário: Formatar um valor monetário com duas casas decimais e separador de milhares.