Aula 7

Descrição

Recapitular o que vimos, ver exercício angular

Aplicativo do tempo

US2 - Melhorar aparecia com Angular Material

Conhecem angular material?

Framework de estilo que o angular fornece, onde temos diversos componentes prontos para uso.

https://material.angular.io

O que vamos ver:

- Como configurar o Angular Material
- Como atualizar o nosso UC com o Angular Material

Angular Material

Para entregar uma boa aplicação, precisamos estilizar ela. Como desenvolvedor full-stack, é difícil estilizar a aplicação, por isso vamos se fazer da biblioteca Angular Material, ela fornece já várias diretivas para deixar nossos aplicativos demais.

Site https://material.angular.io

Kit de desenvolvimento:
https://material.angular.io/cdk/categories

Angular Material

Vamos começar adicionando o angular material ao nosso aplicativo do tempo:
ng add @angular/material
Observem as mudanças após instalar, como no index.html, angular.json, packages.json e no angular modules.

Vamos criar o module do material
ng g m material --flat -m app

Angular Material

Vamos adicionar 3 componentes no nosso módulo

- MatButtonModule,
- MatToolbarModule,
- MatIconModule

Angular Material

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { MatButtonModule } from '@angular/material/button';
import { MatToolbarModule } from '@angular/material/toolbar';
import { MatIconModule } from '@angular/material/icon';
@NgModule({
  declarations: [],
  imports: [
    CommonModule,
    MatButtonModule,
    MatToolbarModule,
    MatIconModule
  ],
  exports: [
    MatButtonModule,
    MatToolbarModule,
    MatIconModule
  ]
})
export class MaterialModule { }
```

Angular Material

Juntamente com o angular material vamos usar o Angular FLex Layout

npm i @angular/flex-layout

Para mais detalhes acessar :

https://github.com/angular/flex-layout

Angular Material

Vamos começar alterando o nosso cabeçalho no app.component.html

- Trocar o h1 pelo componente <mat-toolbar>
  - https://material.angular.io/components/toolbar/overview
- Vamos criar um mat-card para mostrar nossa previsão
  - https://material.angular.io/components/card/overview

Angular Material

```
<mat-toolbar color="primary">
  <span>Aplicativo do tempo</span>
</mat-toolbar>
<div fxLayoutAlign="center">
  <div class="mat-caption vertical-margin">Sua cidade, sua previsão, aqui e agora!</div>
</div>
<div fxLayout="row">
  <div fxFlex></div>
  <mat-card fxFlex="300px">
    <mat-card-header>
      <mat-card-title>
        <div class="mat-headline">Tempo agora</div>
      </mat-card-title>
    </mat-card-header>
    <mat-card-content>
      <app-tempo-atual></app-tempo-atual>
    </mat-card-content>
  </mat-card>
  <div fxFlex></div>
</div>
```

Angular Material

Vamos alterar nosso componente do tempo

```
<div *ngIf="tempoAtual">
  <div fxLayout="row">
    <div fxFlex="66%" class="mat-title no-margin">{{tempoAtual.cidade}}, {{tempoAtual.pais}}</div>
    <div fxFlex class="right mat-subheading-2 no-margin">{{tempoAtual.date | date:'dd MM EEEE' }}</div>
  </div>
  <div fxLayout="row" fxLayoutAlign="center center">
    <div fxFlex="55%">
      <img style="zoom: 175%" alt="72" height="72" [src]="tempoAtual.image">
    </div>
    <div fxFlex class="right no-margin">
      <p class="mat-display-3">{{tempoAtual.temperatura | number:'1.0-0'}}℃
        <span class="mat-display-1 unit">℉</span>
      </p>
    </div>
  </div>
  <div fxLayoutAlign="center" class="mat-caption">
    {{tempoAtual.descricao}}
  </div>
</div>
```

Angular Material

Vamos alterar nosso componente do tempo

tempo-atual.component.CSS:

```
.right {
  text-align: right
}
.no-margin {
  margin-bottom: 0
}
.unit {
  vertical-align: super;
}
```

STYLE.CSS:

```
html, body { height: 100%; }
body { margin: 0; font-family: Roboto, "Helvetica Neue", sans-serif; }

.vertical-margin {
 margin-top: 16px;
 margin-bottom: 16px;
}
```

Angular - Exercício

1. Entregar aplicativo com angular material
2. O exercício da aula anterior era de criar um componente de poluição. Aplique o angular material neste componente.

Aplicativo do tempo

US - Criar busca

- Two-way data binding
- Forms
- Interação entre componentes
- Validar campo
- validação u

Aplicativo do tempo

Para termos nossa busca, precisamos de um input para buscar os dados. Geralmente se recomenda os input estarem em volta de uma form. O form permite validações e fazer devidas tratativas.

Existem dois tipos de forms:

- Template Drive: feito no HTML
- Reactive: Comportamento orientado pelo TypeScript

Importe os módulos de forms no app.module

```
import { FormsModule, ReactiveFormsModule } from '@angular/forms'
```

Aplicativo do tempo

Vamos criar o componente buscaCidade, nele iremos utilizar os componentes do material. MatFormFieldModule e MatInputModule. Importe eles no module do material gerado anteriormente.

Criei o novo componente com a cli
ng g c buscaCidade --module=app.module

Aplicativo do tempo

Criar um template

HTML:

```
<form>
  <mat-form-field>
    <mat-icon matPrefix>busca</mat-icon>
    <input matInput placeholder="Digita sua cidade ou cep"arialabel="Cidade ou CEP" [formControl]="busca">
  </mat-form-field>
</form>
```

TS:

```
busca = new FormControl()
import { FormControl } from '@angular/forms';
```

APP.COMPONENT.HTML

```
<div fxLayoutAlign="center">
  <app-busca-cidade></app-busca-cidade>
</div>
```

Aplicativo do tempo

Agora que temos um input, vamos adicionar capacidade no serviço do tempo para buscar de acordo com o que é digitado. Lembrando que a regra deve validar quando é uma cidade ou um cep .
Primeiro vamos alterar o nosso método do nosso serviço do tempo para aceitar os novos parâmetros e depois iremos refatorar a chamada da api para se comportar de acordo com o parâmetro que irá receber.

Aplicativo do tempo

```
getCurrentWeather(busca: string | number, pais?: string): Observable<ITempoAtual> {
    let uriParams = ''
    if (typeof busca === 'string') {
      uriParams = `q=${busca}`
    } else {
      uriParams = `zip=${busca}`
    }
    if (pais) {
      uriParams = `${uriParams},${pais}`
    }
    return this.getCurrentWeatherHelper(uriParams)
  }

  private getCurrentWeatherHelper(uriParams: string): Observable<ITempoAtual> {
    return this.httpClient.get<ICurrentWeatherData>(
      `${environment.baseUrl}api.openweathermap.org/data/2.5/weather${uriParams}&appid=${environment.appId}`)
      .pipe(map(data => this.transformToITempoAtual(data)))
  }
```

Aplicativo do tempo

Agora que temos o nosso método pronto vamos alterar a parte da busca no componete da cidade, Ts e Html.

```
constructor(private tempoService: TempoService) { }

  ngOnInit(): void {
    this.busca.valueChanges.subscribe((valorDaBusca) => {
      if (valorDaBusca) {
        const valorDoInput = valorDaBusca.split(',').map(letra => letra.trim())
        this.tempoService.getCurrentWeather(valorDoInput[0], valorDoInput.length > 1 ?
valorDoInput[1] : undefined).subscribe(data => (console.log(data)))
      }
    })
  }
```

```
 <mat-hint> Para buscar por país, informe como o exemplo 'Lages, BR'</mat-hint>
```

Aplicativo do tempo

Agora vamos fazer controle dos dados que são informados pelo usuário no forms. No componente de busca,

1. Altere o import que contém o FormControl e incluir o Validators.
2. Adicione um limite mínimo de valores no form de busca busca = new FormControl('', [Validators.minLength(2)])
1. Adicione um aviso no html se estiver invalido. <mat-error *ngIf="busca.invalid"> Digite mais que um character na busca</mat-error>

Aplicativo do tempo

Bem, se perceberem a cada alteração está fazendo uma busca na rede. Vamos alterar o nosso serviço para ele ter um delay antes de realizar uma busca.
No ngOnInit, antes de realizar a ação de subscribe, que é para realizar a ação de consumo da api, incluir a ação de .pipe(debounceTime(1000)) para que seja esperado um segundo antes de cada chamada.

Aplicativo do tempo

Neste momento ainda não há interação entre os componentes. Veremos isso na nossa próxima aula. Existem 4 formas de realizar essa ação e vamos entendê-las melhor.

Angular - Exercício

1. Altere a parte de erro de busca, para um método que valide os erros .
2. Agora que aprendemos a consumir a api, consuma a api de poluição do ar e mostre o resultado dos componentes como a concentração de carbono criando um novo componente do angular e mostrando junto com a previsão do tempo

api.openweathermap.org/data/2.5/air_pollution?lat=-27.8165664&lon=-50.325883&appid=bc93fb691907d0c9163732716afcb58d