Angular

Ng g m [nome] –routing – para módulos com routing configurado.

Ng g c [caminho]/[name]

http://localhost:3000/flavio/photos

Vimos nesta aula:

* Instalação do Angular CLI
* Criação de um novo projeto com a ferramenta e como executá-lo
* Compreensão da estrutura criada
* *Data binding* através de *Angular Expression*
* *Data binding* de atributos
* Convenções adotadas até o momento

Utilizando o Angular, temos acesso à sua poderosa capacidade de Data binding. Mas existem algumas diferenças em relação ao seu uso!

Utilizamos {{ }} dentro de tags e [ ] para atributos

One way data binding – para atributo

**Em que lugar carregamos CSSs ou *scripts* globais?**

Se quisermos usar [jQuery](https://jquery.com/), [Bootstrap](https://getbootstrap.com/), [Foundation](https://foundation.zurb.com/apps/docs/#!/), e afins, conseguimos baixá-los pelo terminal.

Vamos, então, pausar o Angular CLI, usar o comando npm install bootstrap@4.1.1 dentro da pasta do projeto, isto é, iremos utilizar a versão 4.1.1 do Bootstrap. Não se preocupe caso não o conheça, pois aprenderemos seus aspectos fundamentais ao longo do curso.

Podemos declarar via import no arquivo styles.css de forma global,

@import 'bootstrap/dist/css/bootstrap.min.css';

Ou no arquivo angular.json declarar o caminho dentro da pasta node\_modules

"styles": [

"src/styles.css",

"./node\_modules/bootstrap/dist/css/bootstrap.min.css"

],

"scripts": []

**Inbound properties: passando dados para o componente**

Para usarmos temos que adicionar @Input(), que faz parte do angular/core,

import { Component, Input } from "@angular/core";

@Component({

selector: 'ap-photo',

templateUrl: 'photo.component.html'

})

export class PhotoComponent {

@Input() description='';

@Input() url='';

}

Assim, indicamos que description e url são **Inbound properties**, ou seja, aceitam receber um valor por meio de sua forma declarativa.

O @Input() permite que o componente receba valores externos quando usado na forma declarativa no template de outros componentes.

Componentes declarados no array declarations de um módulo são visíveis para os componentes também declarados no array.

**\*ngFor =**

**<app-photo \*ngFor="let photo of photos" [imageUrl]='photo.imageUrl' [description]='photo.description'>{{photo.description}}</app-photo>**

Vimos neste capítulo:

* Como o arquivo bootstrap.css pode ser adicionado ao processo de build do Angular CLI.
* Criação de um novo componente
* A importância de declarar o componente em um módulo
* Como passar dados para o componente através das inbound properties
* Criação de um módulo e boas práticas
* A diretiva \*ngFor

**HttpClient e injeção de dependência**

Para conseguirmos usar o HttpCliente após adicionarmos o import em nosso *app.module.ts ou no Module que estiver no contesto do componente, no caso photo.module.ts*, precisamos importar em nosso *component* o seguinte: import {HttpClient} from '@angular/common/http';

Tópicos vistos na aula:

* Consumir uma Web API através do serviço HttpClient
* Injeção de dependência e a importância de *providers*
* Isolamento da lógica de acesso à Web API em classe de serviço
* Tipagem do retorno da API através de interface e sua vantagem
* Componentes possuem ciclo de vida
* A fase *OnInit*