WP, predavanje 08, 2025./2026.

Direktive

1. Početi od `angular-scrapbook`

2. `@if` + `ngOnDestroy()`:

   a. U `app.component.ts` dodati:

   ```
     show_message: boolean = true;
     ngOnInit() {
       setTimeout(() => {
         this.show_message = false;
       }, 2000);
     }
   ```

   b. U `app.component.html` dodati:

   ```
   @if (show_message) {
     <app-message [message]="'Destroyed after 2 s'"></app-message>
   }
   ```

   c. Definirati u `message.component.ts`:

   ```
   ngOnDestroy() {
     console.log("This will be triggered on destroy");
   }
   ```

3. `NgStyle` i `NgClass`:

   a. `import { CommonModule } from "@angular/common";`

   b. Dodati `CommonModule` u `imports` komponente u kojoj koristimo te direktive

   c. U tijelu definirati npr.

   ```
     rectangleStyle(i: number) {
       const both_have: object = { 'width': '50px', 'height': '50px' };
       if (i === 0)
         return {...both_have, 'background': "tomato"};
       else if (i === 1)
         return {...both_have, 'background': "dodgerblue"}
       else return {...both_have};
     }
   ```

d. U predlošku definirati npr.:

```
<!--
  Ovo je brže napraviti korištenjem class nego s ngClass,
  ali primjera radi. Btw. ovaj objekt može živjeti u scope-u
  komponente i mijenjati se u ovisnosti o različitim
  uvjetima i događajima
-->
<div class="rectangles"
  [ngClass]="{'d-flex': true, 'justify-content-around': true}"
>
  <div class="rectangle" [ngStyle]="rectangleStyle(0)"></div>
  <div class="rectangle" [ngStyle]="rectangleStyle(1)"></div>
</div>
```

4. Custom direktiva
   a. Stvoriti ju s: `ng generate directive FadeIn`
   b. Definirati fade-in efekt:

```
import { Directive, ElementRef } from '@angular/core';

@Directive({
  selector: '[appFadeIn]'
})
export class FadeInDirective {

  constructor(protected el: ElementRef) {}

  n_steps = 30;
  delta_ms = 10;

  ngOnInit() {
    this.el.nativeElement.style.opacity = 0;
    let counter = 0;
    let interval = setInterval(() => {
      this.el.nativeElement.style.opacity = counter++ / this.n_steps;
      if (counter === this.n_steps)
        clearInterval(interval);
    }, this.delta_ms);
  }

}
```

c. Unijeti `FadeInDirective` u `imports` komponente u kojoj ju koristimo

d. Pričvstiti ju na neki element prikazan u predlošku komponente, npr.

```
<div appFadeIn class="rectangle" …></div>
```

e. Definirajmo još neki input za tu direktivu i event handler-e:

```
  setBackground(color: string) {
    this.el.nativeElement.style.backgroundColor = color;
  }

  lastColor: string = "";
  appFadeIn = input<string>();
  onMouseEnter() {
    this.lastColor = this.el.nativeElement.style.backgroundColor;
    this.setBackground(this.appFadeIn() || '#cfecfe');
  }

  onMouseLeave() {
    this.setBackground(this.lastColor || "#ffffff");
  }
```

f. Nadogradimo dekorator:

```
@Directive({
  selector: '[appFadeIn]',
  host: {
    '(mouseenter)': 'onMouseEnter()',
    '(mouseleave)': 'onMouseLeave()'
  }
})
```

g. Novi poziv:

```
<div [appFadeIn]="'gold'" …></div>
```

Template-driven forma

1. Stvoriti komponentu: `ng g c tdf`
2. Kopirati formu s Bootstrap dokumentacije `tdf.component.html` i nadograditi s Angular stvarima za formu:

```
<form class="w-50" #myForm="ngForm">
  <div class="mb-3">
    <label for="exampleInputEmail1" class="form-label">
      Email address
    </label>
    <input [(ngModel)]="initial_mail"
      name="email" type="email" class="form-control"
      id="exampleInputEmail1" aria-describedby="emailHelp"
    >
  </div>
  <div class="mb-3" [ngModelGroup]="'passwords'">
    <label for="exampleInputPassword1" class="form-label">
      Password
    </label>
    <input ngModel name="password" type="password"
      class="form-control" id="exampleInputPassword1"
    >

    <label for="exampleInputPassword2" class="form-label">
      Confirm Password
    </label>
    <input ngModel name="confirm_password" type="password"
      class="form-control" id="exampleInputPassword2"
    >
  </div>
  <div class="mb-3 form-check">
    <input ngModel name="check" type="checkbox"
      class="form-check-input" id="exampleCheck1"
    >
    <label class="form-check-label" for="exampleCheck1">
      Check me out
    </label>
  </div>
  <button (click)="submit(myForm)" type="submit" class="btn btn-primary">
    Submit
  </button>
</form>
```

3. `tdf.component.ts:`

```
import { CommonModule } from '@angular/common';
import { Component } from '@angular/core';

import { FormsModule, NgForm } from "@angular/forms";

@Component({
  selector: 'app-tdf',
  imports: [FormsModule, CommonModule],
  templateUrl: './tdf.component.html',
  styleUrl: './tdf.component.scss'
})
export class TDFComponent {

  initial_mail: string = "jmaltar@mathos.hr";

  fauxSendToServer(data: object) {
    console.log(data);
  }

  submit(form: NgForm) {
    const data: object = form.value;
    this.fauxSendToServer(data);
  }
}
```

4. Gotovo