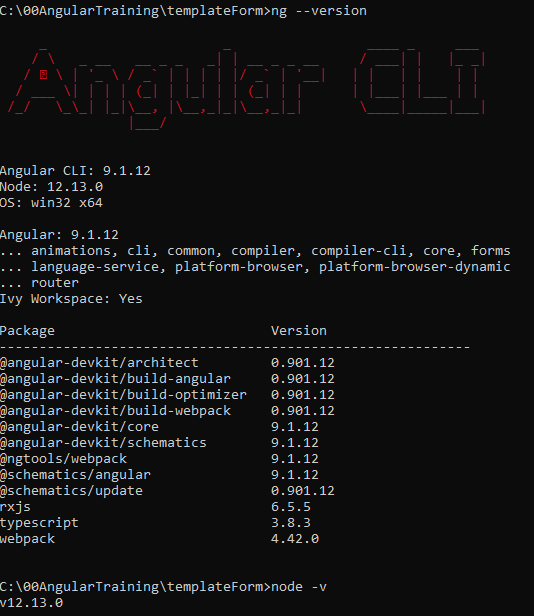
Angular Attribute Directive Exercise

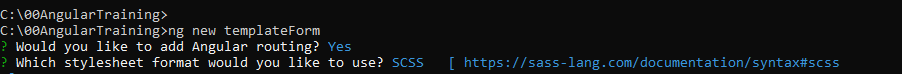
Prerequisite

1. Angular version 9 or above
2. Nodejs version 12.x or above

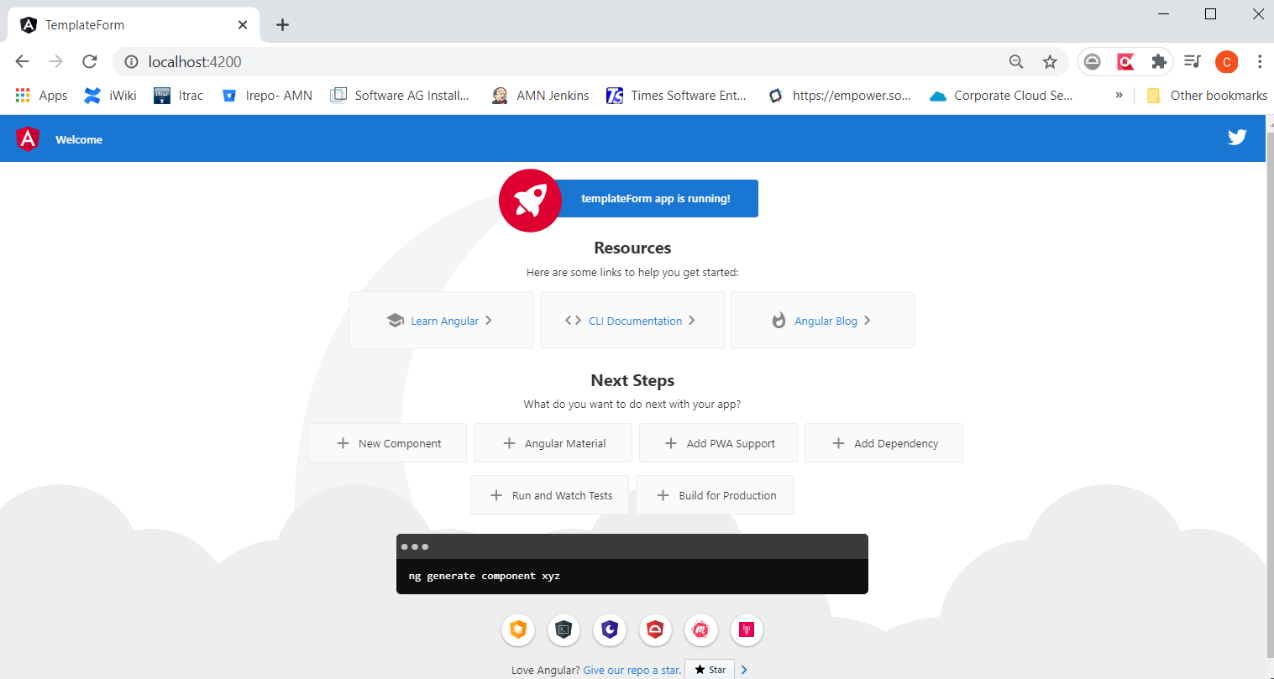


Preparation

1. Create New Angular Project
   1. ng new <project name>



* 1. ng serve



1. Add Bootstrap – add below into ./src/styles.css

@import url('https://unpkg.com/bootstrap@3.3.7/dist/css/bootstrap.min.css')

Exercise

1. Create a new highlight directive
2. Add below codes into highlight.directive.ts

  constructor(private el: ElementRef) { }

  @Input() defaultColor: string;

  @Input('appHighlight') highlightColor: string;

  @HostListener('mouseenter') onMouseEnter() {

    this.highlight(this.highlightColor || this.defaultColor || 'red');

  }

  @HostListener('mouseleave') onMouseLeave() {

    this.highlight(null);

}

  private highlight(color: string) {

    this.el.nativeElement.style.backgroundColor = color;

  }

1. Replace app.component.html with below

<h1>My First Attribute Directive</h1>

<h4>Pick a highlight color</h4>

<div>

  <input type="radio" name="colors" (click)="color='lightgreen'">Green

  <input type="radio" name="colors" (click)="color='yellow'">Yellow

  <input type="radio" name="colors" (click)="color='cyan'">Cyan

</div>

<p [appHighlight]="color">Highlight me!</p>

<p [appHighlight]="color" defaultColor="violet">

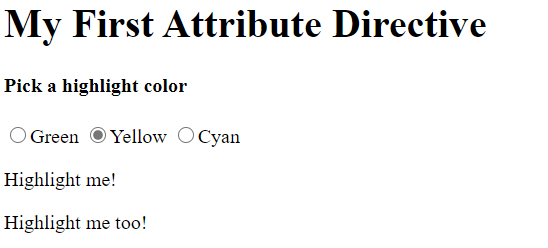
  Highlight me too!

</p>

1. Add below code into app.component.ts

  color: string;

1. Test the application, use the radio buttons to switch the colors and hover the text to see the effect



1. Update app.component.scss as below

.italicstyle {

    font-style: italic

}

1. Update highlight.directive.ts as below

  constructor(private el: ElementRef) { }

  @Input() defaultColor: string;

  @Input('appHighlight') highlightColor: string;

  @HostBinding('class.italicstyle') private ishovering: boolean;

  @HostListener('mouseenter') onMouseEnter() {

    this.highlight(this.highlightColor || this.defaultColor || 'red');

  }

  @HostListener('mouseleave') onMouseLeave() {

    this.highlight(null);

    this.ishovering = false;

  }

  private highlight(color: string) {

    this.el.nativeElement.style.backgroundColor = color;

  }

1. Test the application
2. Create a uppercase directive as below

  constructor(public el: ElementRef) { }

  @HostListener('input', ['$event']) onInput($event)

  {

    $event.target.value = $event.target.value.toUpperCase();

  }

1. Add below code into app.component.html

<input type="text" [formControl]="field" appUppercase>

<br>

1. Update app.component.ts as below

  field = new FormControl('');

1. Test the application, enter any character into the textbox, it will be automatically changed to uppercase



1. Create an namevalidator directive as below

@Directive({

  selector: '[appNamevalidator]',

  providers: [

      { provide: NG\_VALIDATORS, useExisting: forwardRef(() => NamevalidatorDirective), multi: true }

  ]

})

export class NamevalidatorDirective implements Validator {

  constructor() { }

  validate(c: AbstractControl): { [key: string]: any } {

    if (c.value === 'test') {

      return {test: true};

    }

    return null;

}

}

1. Add below code into app.component.ts

  name = new FormControl('');

1. Update app.component.html as below

<br>

<input type="text" [formControl]="name" appNamevalidator>

<button (click)="test()"> test</button>

<p \*ngIf="name.errors?.test"> error name</p>

1. Test the application, enter “test” into the new created field, you will able to see the error msg



Additional Exercise

1. Create a directive with below ability and apply it as “**HighlightDirective**” did
   1. Set font color to blue when hover
   2. Set font color to orange when not hover
   3. Set default color to black
   4. Set border to solid, 5px, red when hover
   5. Set border to none when not hover
2. Create a lowercase directive with ability to make input field always lowercase
3. Create a directive with validation ability – return an error when input value greater than 100.