ANGULAR

فصل هفتم

ساخت دستورهای صفتی (ATTRIBUTE DIRECTIVE) دلخواه در انگولار

ارسال اطلاعات به ویژگی‌ها (PROPERTY) و پاسخ به رویدادها (EVENT) در دستورهای صفتی

ایجاد دستورهای ساختاری (STRUCTURAL DIRECTIVE) در انگولار

Telegram id : @ArmanAbi

ساخت دستورهای صفتی (Attribute Directive)

دستورهای صفتی یا Attribute Directive: به دستورهایی گفته می‌شود که به عنوان یک صفت یا attributeدر تگ‌های HTMLمورد استفاده قرار می‌گیرند. مثلا دستور ngClassیک کلاس به تگ اضافه می‌کند و دخل و تصرفی در ساختار کلی صفحه ایجاد نخواهد کرد.

دستورهای ساختاری یا Structural Directive: به دستورهایی گفته می‌شود که به عنوان یک دستور ساختاری وارد عمل شده و تگ‌های HTMLجدیدی به صفحه و قالب ما اضافه می‌کند. مانند دستور *ngIf که یک المان جدید را تولید و یا حذف می‌کند. به عبارت دیگر این نوع دستور در ساختار DOMتغییرات ایجاد خواهد کرد.

یک مثال دیگر که در آن به صورت ترکیبی از دستورهای *ngIfو *ngForو ngClassو همچنین ngStyleاستفاده شده است، با یکدیگر بررسی خواهیم کرد.

ساخت دستورهای صفتی (Attribute Directive)

فرض کنید می‌خواهیم در یک صفحه اعداد زوج و فرد را به صورت مجرا نمایش دهیم. یعنی با کلیک روی یک دکمه مجموعه‌ی اعداد فرد نمایش داده شده که دارای استایل و کلاس خاص هستند. بنابراین در فایل app.component.html کدهای زیر را اعمال خواهیم کرد:

```
<div class="container" dir="rtl" style="margin-top: 30px;">
    <div class="row">
        <div class="col-xs-12">
            <button class="btn btn-primary" (click)="onlyFalse = !onlyFalse">نمایش اعداد فرد</
button>
        </div>
        <br><br>
        <ul class="list-group">
            <div *ngIf="onlyFalse">
                <li class="list-group-item"
                    *ngFor="let odd of oddNumber"
                    [ngClass]="{odd: odd % 2 !== 0}"
                    [ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Yellow':'transparent'}"
                >
                    {{odd}}
                </li>
            </div>
            <div *ngIf="!onlyFalse">
                <li class="list-group-item"
                    *ngFor="let even of evenNumber"
                    [ngClass]="{even: even % 2 === 0}"
                    [ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Black':'transparent'}"
                >
                    {{even}}
                </li>
            </div>
        </ul>
    </div>
</div>
```

ساخت دستورهای صفتی (Attribute Directive)

سپس در فایل app.component.ts یک سری ویژگی تعریف می‌کنیم:

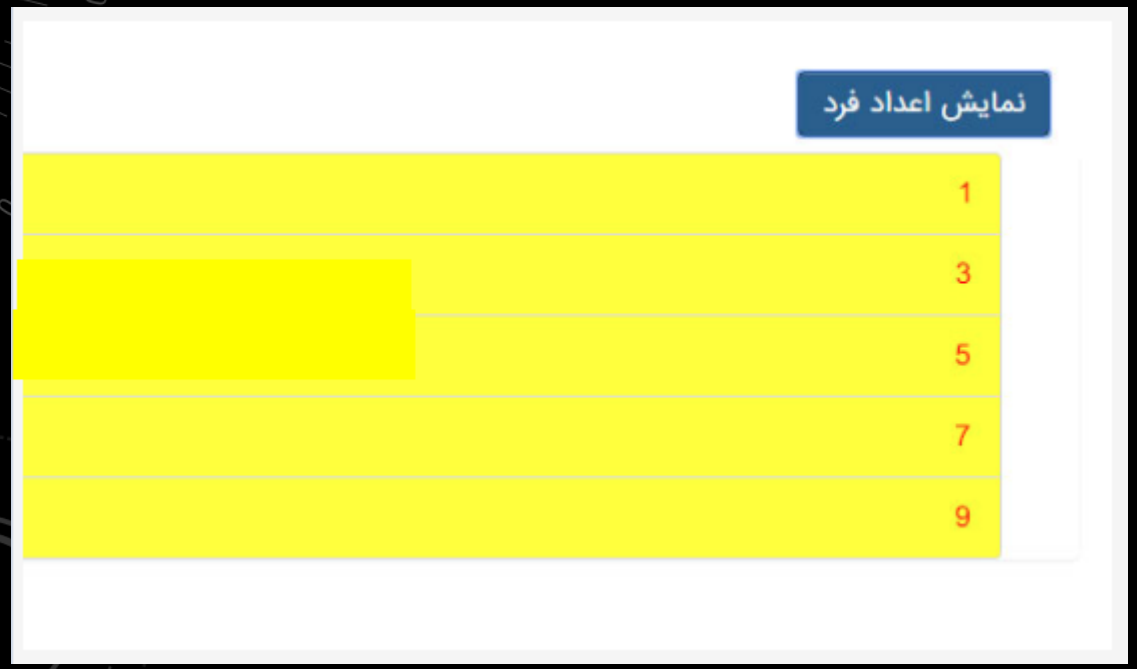
```
import {Component} from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']
})
export class AppComponent {
    numbers: number[] = [1,2,3,4,5,6,7,8,9,10];
    oddNumber: number[] = [1,3,5,7,9];
    evenNumber: number[] = [2,4,6,8,10];
    onlyFalse: boolean = false;

    onLoadOdd(){

    }
}
```

در نهایت فایل app.component.css را نیز به صورت زیر ارائه خواهیم داد تا در قالب HTML تغییرات متناسب با دستور ngClass اعمال شود:

```
.odd{
    color: red;
}
.even{
    color: yellow;
}
```

ساخت دستورهای صفتی (Attribute Directive)

در صورتیکه از این مجموعه‌ی دستورات خروجی بگیرید باید با تصویر زیر مواجه شده باشید:



ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

پاسخ به یک رویداد (Event) در انگولار

حال در این بخش می‌خواهیم هنگامیکه کاربر روی یک تگ خاص قرار گرفت به عنوان مثال رنگ پس زمینه‌ی آن تغییر کند و یا وقتی ماوس خود را از روی آن تگ برداشت رنگ آن مجددا به حالت اولیه باز گردد.

این موضوع دقیقا مشابه رخ دادن یک رویداد یا event است. یعنی شما در ابتدا یک event تعریف می‌کنید که متناسب با آن دستور خاص عمل خواهد کرد.

معرفی مفسر HostListener@ در انگولار

برای تعریف یک event درون یک دستور دلخواه، باید از مفسر یا دکوراتور HostListener استفاده کرده و نام event را به صورت آرگومان به آن ارسال کنی. در نهایت نام رویدادی که در DOM رخ می‌دهد را به ادامه‌ی دستور خود بیافزاییم.

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

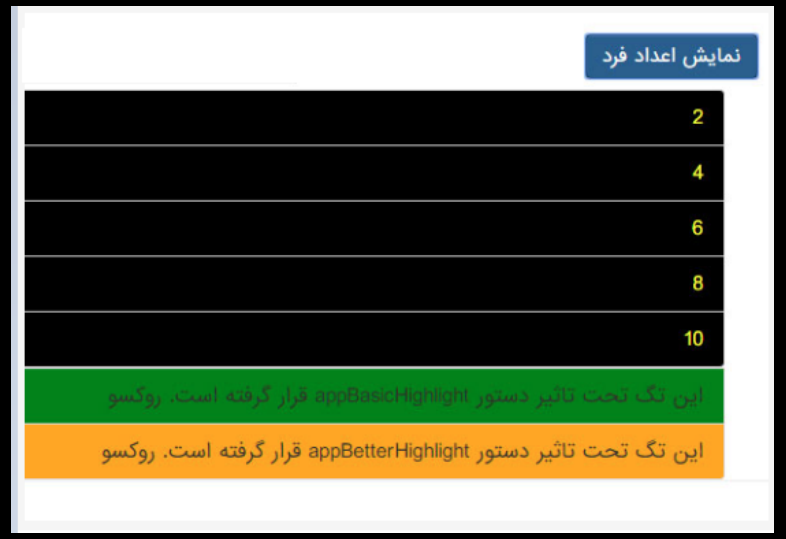
برای روشن‌تر شدن درک این موضوع مثال قبل را در نظر بگیرید. بنابراین فایل appBetterHighlight.directive.ts را باز کرده و دستورهای زیر را به آن اضافه می‌کنیم:

```
import {Directive, ElementRef, HostListener, OnInit, Renderer2} from '@angular/core';

@Directive({
  selector: '[appBetterHighlight]'
})
export class BetterHighlightDirective implements OnInit {

  constructor(private elRef: ElementRef, private renderer: Renderer2) {
  }

  ngOnInit() {

  }

  @HostListener('mouseenter') mouseover(eventData: Event) {
    this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'orange')
  }

  @HostListener('mouseleave') mouseleave(eventData: Event) {
    this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'transparent')
  }
}
```

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

کافیست این کد را اجرا کنید. همانطور که ملاحظه می‌کنید با قرار دادن ماوس روی تگ موردنظر با خروجی زیر مواجه خواهید شد

و در حالت عادی اگر ماوس خود را از روی تگ بردارد تصویر زیر برای شما نمایش داده می‌شود:



ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

معرفی مفسر @HostBinding در انگولار

قبل از توضیح این مفسر باید خدمت شما عزیزان عرض کنم که استفاده از دستور renderer به عنوان اصولی‌ترین راه شناخته می‌شود اما گاهی می‌خواهیم یک ویژگی را با تعداد خط کمتری تغییر دهیم. در این صورت می‌توان از مفسر HostBinding استفاده کرد که یک ویژگی را روی یک صفت CSSپیاده‌سازی می‌کند.

برای درک بهتر مثال قبل را در نظر بگیرید. برای آنکه رنگ پس زمینه در تگی که از دستور appBetterHighlight پیروی می‌کند، تغییر یابد می‌توان از این مفسر بهره به صورت زیر بهره برد:

```
import {Directive, ElementRef, HostBinding, HostListener, OnInit, Renderer2} from '@angular/core'

@Directive({
  selector: '[appBetterHighlight]'
})
export class BetterHighlightDirective implements OnInit {

  @HostBinding('style.backgroundColor') backgroundColor: string = 'transparent';

  constructor(private elRef: ElementRef, private renderer: Renderer2) {
  }

  ngOnInit() {

  }

  @HostListener('mouseenter') mouseover(eventData: Event) {
    // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'orange')
    this.backgroundColor = 'orange'
  }

  @HostListener('mouseleave') mouseleave(eventData: Event) {
    // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'transparent')
    this.backgroundColor = 'transparent'
  }
}
```

همانطور که ملاحظه کردید یک ویژگی به نام backgroundColor در ابتدا تعریف کردیم و مقدار style.backgroundColorرا معادل آن قرار داده و توسط HostBinding آن را اعمال می‌کنیم.

بنابراین درون رویدادهای mouseover و mouseleave نیز این تغییرات را با دسترسی به این ویژگی و انتساب یک مقدار مشخص به آنها، ایجاد می‌کنیم. خروجی این روش نیز دقیقا مشابه روش استفاده از کلاس Renderer2 است.

ارسال (Bind) اطلاعات به یک ویژگی (Property) در انگولار

فرض کنید می‌خواهیم اطلاعات را سمت کامپوننت پردازش کرده و سپس به یک تگ به عنوانه ویژگی ارسال (Bind) کنیم. در این حالت باید مشابه قبل یک ویژگی دریافت شده با مفسر @Input ایجاد کرده و سپس مقادیر موجود در فایل دستورات (directive.ts) را متناسب با آن تنظیم کنیم.

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

بنابراین در فایل better-highlight.directive.ts مثال قبل تغییرات زیر را لحاظ خواهیم کرد:

```
import {Directive, ElementRef, HostBinding, HostListener, Input, OnInit, Renderer2} from '@angu
lar/core';

@Directive({
    selector: '[appBetterHighlight]'
})
export class BetterHighlightDirective implements OnInit {

    @Input() defaultColor: string;
    @Input() highlightColor: string;

    @HostBinding('style.backgroundColor') backgroundColor: string = this.defaultColor;

    constructor(private elRef: ElementRef, private renderer: Renderer2) {
    }

    ngOnInit() {

    }

    @HostListener('mouseenter') mouseover(eventData: Event) {
        // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'orange')
        this.backgroundColor = this.highlightColor;
    }

    @HostListener('mouseleave') mouseleave(eventData: Event) {
        // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'transparent')
        this.backgroundColor = this.defaultColor;
    }
}
```

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

و در نهایت امر داخل فایل app.component.ts که این دستور را اضافه کرده‌ایم، ویژگی را bind(ارسال) می‌کنیم:

```
<div class="container" dir="rtl" style="margin-top: 30px;">
    <div class="row">
        <div class="col-xs-12">
            <button class="btn btn-primary" (click)="onlyFalse = !onlyFalse">نمایش اعداد فرد</
button>
        </div>
        <br><br>
        <ul class="list-group">
            <div *ngIf="onlyFalse">
                <li class="list-group-item"
                    *ngFor="let odd of oddNumber"
                    [ngClass]="{odd: odd % 2 !== 0}"
                    [ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Yellow':'transparent'}"
                >
                    {{odd}}
                </li>
            </div>
            <div *ngIf="!onlyFalse">
                <li class="list-group-item"
                    *ngFor="let even of evenNumber"
                    [ngClass]="{even: even % 2 === 0}"
                    [ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Black':'transparent'}"
                >
                    {{even}}
                </li>
            </div>
            <li class="list-group-item" appBasicHighlight>این تگ تحت تاثیر دستور appBasicHighl
ight روکسو است. قرار گرفته</li>
            <li class="list-group-item" appBetterHighlight [defaultColor]="'pink'" [highlightCo
lor]="'brown'">این تگ تحت تاثیر دستور appBetterHighlight روکسو است. قرار گرفته</li>
        </ul>
    </div>
</div>
```

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

بنابراین در فایل component.ts خواهیم داشت:

```
import {Component, OnInit, Input, EventEmitter, Output, ViewChild, ElementRef} from '@angular/c
ore';

@Component({
    selector: 'app-server',
    templateUrl: './server.component.html',
    styleUrls: ['./server.component.css'],
})
export class ServerComponent implements OnInit {

    serverName: string = '';
    serverContent: string = '';
    serverList = [];

    @Input('initS') initServer = {name: "روکسو", content: "این سرور باید به کامپونتت والد ار
سال شود"};

    @Output() localServerCreated = new EventEmitter<{ serverName: string, serverContent: string
}>();
    @Output('exServerCreated') externalServerCreated = new EventEmitter<{ serverName: string, s
erverContent: string }>()

    @ViewChild('referenceServerName') referenceServerName: ElementRef;
    @ViewChild('referenceServerContent') referenceServerContent: ElementRef;


    constructor() {
    }


    ngOnInit() {
    }

    onCreateLocalServer() {

        this.localServerCreated.emit({
            serverName: this.referenceServerName.nativeElement.value,
            serverContent:  this.referenceServerContent.nativeElement.value
        });
    }

    onCreateExternalServer() {
        this.externalServerCreated.emit({
            serverName: this.referenceServerName.nativeElement.value,
            serverContent:  this.referenceServerContent.nativeElement.value
        });
    }
}
```

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

همانگونه که مشاهده خواهید کرد ویژگی defaultColor را برابر رنگ صورتی و ویژگی highlightColor را معادل رنگ قهوه‌ای قرار داده‌ایم. بنابراین خروجی ما در حالتی که صفحه بروز می‌شود به صورت زیر است:

ارسال اطلاعات به ویژگی‌ها (Property) و پاسخ به رویدادها (Event) در دستورهای صفتی

و هنگامیکه روی تگ موردنظر بیاستیم خروجی ما معادل تصویر زیر می‌باشد:

ایجاد دستورهای ساختاری (Structural Directive)

ساخت دستورهای صفتی (Attribute Directive) دلخواه در انگولار

دستور ساختاری (Structural Directive) : به دستوری گفته می‌شود که روی المان‌های DOM تاثیر مستقیم گذاشته و یک تگ را حذف یا اضافه می‌کند.

شروع با یک مثال

فرض کنید می‌خواهیم یک دستور ساختاری به نام unless تولید کنیم که در صورت اعمال آن به یک مجموعه تگ یا template در صورت برقرار نبودن شرط فعال شود و در غیر این صورت تگ را حذف کند.

بنابراین ابتدا در خط فرمان Angular CLI دستور زیر را تایپ می‌کنیم تا یک پوشه به همراه فایل unless.directive.ts برای ما تولید شود:

```
ng g d unless/unless
```

سپس فایل unless.directive.ts را باز کرده و عبارت‌های زیر را درون آن قرار می‌دهیم:

```
import {Directive, Input, TemplateRef, ViewContainerRef} from '@angular/core';

@Directive({
    selector: '[appUnless]'
})
export class UnlessDirective {

    @Input() set appUnless(condition: boolean){
        if(!condition){
            this.vcRef.createEmbeddedView(this.templateRef)
        }else{
            this.vcRef.clear();
        }
    }
    constructor(private templateRef: TemplateRef<any>, private vcRef: ViewContainerRef) {
    }

}
```

ایجاد دستورهای ساختاری (Structural Directive)

همانطور که ملاحظه می‌کنید ابتدا یک ویژگی به نام appUnless تعریف کرده‌ایم که مقدار موردنظر condition را به عنوان ورودی از قالب HTML دریافت می‌کند سپس داخل این ویژگی یک سری دستورهای شرطی قرار داده‌ایم. اما قبل از اینکه دستورات شرطی را بررسی کنیم به توضیح ویژگی‌های templateRef و vcRef می‌پردازیم.

ویژگی templateRef دقیقا مشابه elementRef است با این تفاوت که elementRef روی تگ‌ها یا المان‌ها اعمال می‌شود ولی templateRefروی یک دسته از تگ‌ها که تشکیل template می‌دهند اعمال خواهد شد.

همچنین ویژگی vcRef از نوع کلاس ViewContainerRef تعریف شده است که بیانگر مفهوم «کجایی» یا “ where” یک دستور ساختاری می‌باشد. به عبارت دیگر templateRef مفهوم چطور و ViewContainerRef مفهوم کجایی را می‌رساند.

همچنین ویژگی appUnless را به صورت set تعریف کرده‌ایم که تنها مقادیر را بپذیرد و معادل قرار دهد و نتوان از روی آن مقداری را خواند.

حال با این تغییرات درون فایل app.component.html نیز باید دستورهایی مشابه ذیل قرار دهیم:

```
<div class="container" dir="rtl" style="margin-top: 30px;">
    <div class="row">
        <div class="col-xs-12">
            <button class="btn btn-primary" (click)="onlyFalse = !onlyFalse">نمایش اعداد فرد</
button>
        </div>
        <br><br>
        <ul class="list-group">
            <!--<div *ngIf="onlyFalse">-->
                <!--<li class="list-group-item"-->
                    <!--*ngFor="let odd of oddNumber"-->
                    <!--[ngClass]="{odd: odd % 2 !== 0}"-->
                    <!--[ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Yellow':'transparent'}"--
>
                <!--&gt;-->
                    <!--{{odd}}-->
                <!--</li>-->
            <!--</div>-->
            <!--<div *ngIf="!onlyFalse">-->
                <!--<li class="list-group-item"-->
                    <!--*ngFor="let even of evenNumber"-->
                    <!--[ngClass]="{even: even % 2 === 0}"-->
                    <!--[ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Black':'transparent'}"-->
                <!--&gt;-->
                    <!--{{even}}-->
                <!--</li>-->
            <!--</div>-->
            <div *appUnless="onlyFalse">
                <li class="list-group-item"
                    *ngFor="let even of evenNumber"
                    [ngClass]="{even: even % 2 !== 0}"
                    [ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'Black':'transparent'}"
                    >
                    {{even}}
                </li>
            </div>
            <li class="list-group-item" appBasicHighlight>این تگ تحت تاثیر دستور appBasicHighl
ight روکسو است. قرار گرفته</li>
            <li class="list-group-item" appBetterHighlight [defaultColor]="'pink'" [highlightCo
lor]="'brown'">این تگ تحت تاثیر دستور appBetterHighlight روکسو است. قرار گرفته</li>
        </ul>
    </div>
</div>

<!--<div class="container" dir="rtl">-->
<!--<div class="row">-->
<!--<div class="col-xs-12">-->
<!--<app-lifecycle></app-lifecycle>-->
<!--</div>-->
<!--</div>-->
<!--</div>-->


<!--<div class="container" dir="rtl">-->
<!--<div class="row">-->
<!--<div class="col-xs-12">-->
<!--<app-servers></app-servers>-->
<!--</div>-->
<!--</div>-->
<!--</div>-->
```

ایجاد دستورهای ساختاری (Structural Directive)

در این مجموعه‌ی کد یک سری کدها را به صورت کامنت قرار داده‌ایم که دقیقا در جریان تغییرات باشید.

بنابراین در این بخش یک دستور ایجاد کردیم که شرط درون آن به صورت خلاف شرط معمولی if تعریف شده است.

دستور ساختاری ng-switch در انگولار

در صورتیکه با یکی از زبان‌های برنامه‌نویسی سطح بالا کار کرده باشید همگی با مفهوم دستور شرطی switch آشنا هستید. این دستور متناسب با مقداری که به عنوان value دریافت می‌کند یکی از شروط را اجرا خواهد کرد و در صورتیکه هیچ یک از case های آن با شرط موردنظر همسان نباشد یا از شرط خارج شده و یا یک مقدار پیش‌فرض (default) را ارائه می‌دهد.

برای درک بهتر این دستور در انگولار یک مثال خدمت شما عزیزان ارائه خواهیم داد. بنابراین فایل app.component.ts را باز می‌کنیم و دستور زیر را درون آن قرار می‌دهیم که شامل یک ویژگی به نام value است تا به عنوان مقادیر و شروط به دستور switch ارسال شود:

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

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']
})
export class AppComponent {

    value: number = 10;

    onLoadOdd() {

    }
}
```

ایجاد دستورهای ساختاری (Structural Directive)

همانطور که ملاحظه می‌کنید یک ویژگی به نام value تعریف کرده و سپس مقدار معادل آن را برابر ۱۰ قرار داده‌ایم. حال باید دستور ساختاری ngswitch را اعمال کنیم. بنابراین فایل app.component.ts را باز کرده و سپس:

```
<div class="col-xs-12" [ngSwitch]="value">
    <p *ngSwitchCase="10"> مقدار برابر ۱۰ می باشد</p>
    <p *ngSwitchCase="15"> مقدار برابر ۱۵ می باشد</p>
    <p *ngSwitchCase="100"> مقدار برابر ۱۰۰ می باشد</p>
    <p *ngSwitchDefault> مقدار برابر حالت پیشفرض می باشد</p>
</div>
```

با بررسی مجموعه دستور بالا باید حدس زده باشید که فرآیند اجرا به چه صورت است. در ابتدا یک ویژگی به نام value به ویژگی توکار ngSwitch اعمال می‌شود (Binding) سپس متناسب با مقادیری که از کنترلر کامپوننت app.component.ts دریافت می‌کند تصمیم گیری خود را در دستورهای ستاره‌دار *ngSwitchCase انجام می‌دهد. در صورتیکه مقدار value برابر هیچ یک از مقادیر فوق آنگاه تگ با دستور *ngSwitchDefault اجرا می‌شود.