Directive Deep Dive

Structural Directives:

Structural Directives change the structure of the Dom around this element

If you have ngIf on a paragraph if that condition is false, this paragraph is removed from the Dom, so overall view container is affected

Attribute Directives,you never destroy an element from the Dom, you only change Properties of that Element

Eg: background Color

92) ngFor and ngIf

app.component.html

ngFor

```
      <li
    class="list-group-item"
    *ngFor="let number of numbers">
    {{ number}}
   </li>
```

ngIf

```
     <ul class="list-group">
            <div *ngIf="onlyOdd">
                   <li
                   class="list-group-item"
                   *ngFor="let odd of oddNumbers">
                     {{ odd }}
                   </li>
             </div>
             <div *ngIf="!onlyOdd">
                    <li
```

```
                                        class="list-group-item"
                                    *ngFor="let even of evenNumbers">
                                       {{ even }}
                                       </li>
                         </div>
             </ul>
```

app.component.ts

```
             export class AppComponent {
               // numbers = [1, 2, 3, 4, 5];
               oddNumbers = [1, 3, 5];
               evenNumbers = [2, 4];
               onlyOdd = false; }
```

93)ngClass and ngStyle

app.component.css

ngClass:

```
.container {
  margin-top: 30px;
}
.odd {
  color: red;
}
```

app.component.html

```
       <div *ngIf="onlyOdd">
                     <li
                                   class="list-group-item"
                                   [ngClass]="{odd: odd % 2 !== 0}"
                                            *ngFor="let odd of oddNumbers">
                                   {{ odd }}
                     </li>
```

```
        </div>
        <div *ngIf="!onlyOdd">
                <li
                        class="list-group-item"
                        [ngClass]="{odd: even % 2 !== 0}"
                         *ngFor="let even of evenNumbers">
                        {{ even }}
                </li>
        </div>
```

ngStyle:

```
[ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'yellow' : 'transparent'}"
 [ngStyle]="{backgroundColor: even % 2 !== 0 ? 'yellow' : 'transparent'}"
```

94) Creating a Basic Attribute Directive

Create a new directive(basic-highlight)-basic-highlight.directive.ts

basic-highlight.directive.ts

```
import { Directive, ElementRef, OnInit } from '@angular/core';
@Directive({
  selector: '[appBasicHighlight]'
})
export class BasicHighlightDirective implements OnInit {
  constructor(private elementRef: ElementRef) {
  }
  ngOnInit() {
    this.elementRef.nativeElement.style.backgroundColor = 'green';
  }
}
```

app.module.ts

```
import { BasicHighlightDirective } from './basic-highlight/basic-highlight.directive';
@NgModule({
  declarations: [
    AppComponent,
    BasicHighlightDirective ],
```

app.component.html

```
     <p appBasicHighlight>Style me with basic directive!</p>
```

95)Using the Renderer to build a better Attribute Directive

Create a new directive(better-highlight)-better-highlight.directive.ts

app.module.ts

```
import { BetterHighlightDirective } from './better-highlight/better-highlight.directive';
@NgModule({
  declarations: [
    AppComponent,
    BasicHighlightDirective,
    BetterHighlightDirective,
  ],
```

better-highlight.directive.ts

render-give the template to the DOM

```
import {Directive, OnInit, Renderer2,ElementRef } from '@angular/core';
@Directive({
             selector: '[appBetterHighlight]'
            })
export class BetterHighlightDirective implements OnInit {
constructor(private elRef: ElementRef, private renderer: Renderer2) { }
ngOnInit() {
   this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'blue');
  }
```

app.component.html

```
<ul class="list-group">
            <div *ngIf="onlyOdd">
            <li
              class="list-group-item"
              [ngClass]="{odd: odd % 2 !== 0}"
              [ngStyle]="{backgroundColor: odd % 2 !== 0 ? 'yellow' : 'transparent'}"
              *ngFor="let odd of oddNumbers">
              {{ odd }}
            </li>
        </div>
        <div *appUnless="onlyOdd">
            <li
              class="list-group-item"
              [ngClass]="{odd: even % 2 !== 0}"
              [ngStyle]="{backgroundColor: even % 2 !== 0 ? 'yellow' : 'transparent'}"
              *ngFor="let even of evenNumbers">
              {{ even }}
            </li>
         </div>
    </ul>
    <p appBasicHighlight>Style me with basic directive!</p>
    <p appBetterHighlight>Style me with a better directive!</p>
```

97)Using HostListener to Listen to Host Events

better-highlight.directive.ts

```
import {Directive,Renderer2,OnInit,ElementRef,HostListener } from '@angular/core';
@HostListener('mouseenter') mouseover(eventData: Event) {
  this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'blue',false,false); }
@HostListener('mouseleave') mouseleave(eventData: Event) {
  this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'transparent',false,false);}
```

98)Using HostBinding to Bind to Host Properties

better-highlight.directive.ts

```
import {Directive,Renderer2,OnInit,ElementRef,HostListener,HostBinding} from '@angular/core';
export class BetterHighlightDirective implements OnInit {
  @HostBinding('style.backgroundColor') backgroundColor: string='transparent';
@HostListener('mouseenter') mouseover(eventData: Event) {
this.backgroundColor ='blue'; }
@HostListener('mouseleave') mouseleave(eventData: Event) {
 this.backgroundColor ='transparent'; }
  }
```

99)Binding to Directive Properties

better-highlight.directive.ts

```
import {Directive,Renderer2,OnInit,ElementRef,HostListener,HostBinding,Input} from '@angular/core';
export class BetterHighlightDirective implements OnInit {
  @Input() defaultColor: string = 'transparent';
  @Input() highlightColor: string = 'blue';
 @HostBinding('style.backgroundColor') backgroundColor: string=this.defaultColor;
  constructor(private elRef: ElementRef, private renderer: Renderer2) { }
@HostListener('mouseenter') mouseover(eventData: Event) {
    // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'blue');
    this.backgroundColor = this.highlightColor;
  }
  @HostListener('mouseleave') mouseleave(eventData: Event) {
    // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'transparent');
    this.backgroundColor = this.defaultColor;
  }
}
```

o/p: default-transparent

hover-blue

```
 @HostBinding('style.backgroundColor') backgroundColor: string;
 ngOnInit() {
   this.backgroundColor = this.defaultColor;
   // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'blue');
  }
@HostListener('mouseenter') mouseover(eventData: Event) {
   // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'blue');
   this.backgroundColor = this.highlightColor;
  }
  @HostListener('mouseleave') mouseleave(eventData: Event) {
   // this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'transparent');
   this.backgroundColor = this.defaultColor;
  }
```

app.component.html

```
     <p appBetterHighlight [defaultColor]="’yellow’" [highlightColor]=”'red’”>Style me with a better
directive!</p>
```

o/p: default-yellow,hover-red

```
     <p [appBetterHighlight]="'red'" defaultColor="yellow">Style me with a better directive!</p>
```

better-highlight.directive.ts

```
export class BetterHighlightDirective implements OnInit {
  @Input() defaultColor: string = 'transparent';
  @Input('appBetterHighlight') highlightColor: string = 'blue';
}
```

o/p:default-yellow,hover-red

100)what happens behind the scenes on structural Directive

-structural Directive,without but code was Complicated

app.component.html

```
<ng-template [ngIf]="!onlyOdd">
         <div>
           <li
           class="list-group-item"
           [ngClass]="{odd: even % 2 !== 0}"
           [ngStyle]="{backgroundColor: even % 2 !== 0 ? 'yellow' : 'transparent'}"
           *ngFor="let even of evenNumbers">
           {{ even }}
         </li>
       </div>
</ng-template>
```

101)Building a Structural Directive

//vc-view Container

Create a new directive(unless)-unless.directive.ts,delete a test file

Unless.diective.ts

```
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) { }
}
```

o/p: oddnumber-color-yellow change

app.module.ts

```
import { UnlessDirective } from './unless.directive';
```

```
@NgModule({
 declarations: [
  AppComponent,BasicHighlightDirective,BetterHighlightDirective,UnlessDirective ],
```

app.component.html

```
<div *appUnless="onlyOdd">
     <li
      class="list-group-item"
      [ngClass]="{odd: even % 2 !== 0}"
      [ngStyle]="{backgroundColor: even % 2 !== 0 ? 'yellow' : 'transparent'}"
      *ngFor="let even of evenNumbers">
      {{ even }}
     </li>
    </div>
```

102)ngSwitch

app.component.ts

```
export class AppComponent {
 // numbers = [1, 2, 3, 4, 5];
 oddNumbers = [1, 3, 5];
 evenNumbers = [2, 4];
 onlyOdd = false;
 value = 5;
}
```

app.component.html

```
<div [ngSwitch]="value">
    <p *ngSwitchCase="5">Value is 5</p>
    <p *ngSwitchCase="10">Value is 10</p>
    <p *ngSwitchCase="100">Value is 100</p>
    <p *ngSwitchDefault>Value is Default</p>
   </div>  o/p: value is 10          5 -> 5
```