



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Directives Deep Dive

Alex Thalhammer



**ARCHITECTS**  
INSIDE KNOWLEDGE



**ARCHITECT**

# Contents

- Attribute Directives
- Templates and View Containers
- Structural Directives



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Directives

- „Components without Templates“
- Add behavior to an element
- Examples: ngModel, ngClass, ngStyle



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Attribute Directives



# Case Study

```
<button appClickWithWarning>Delete</button>
```



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Simple Example

```
@Directive({
  selector: '[appClickWithWarning]'
})
export class ClickWithWarningDirective implements OnInit {
  constructor(private elementRef: ElementRef) {}

  ngOnInit(): void {
    this.elementRef
      .nativeElement.setAttribute('class', 'btn btn-danger');
  }
}
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Calling a Directive

```
<button appClickWithWarning>Delete</button>
```



**Host-Element**



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Bindings

```
@Directive({  
  selector: '[appClickWithWarning]'  
})  
export class ClickWithWarningDirective implements OnInit {  
  @Input() warning = 'Are you sure?';  
  @Output() appClickWithWarning = new EventEmitter<void>();  
}
```



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**



# Bindings

```
@Directive({
  selector: '[appClickWithWarning]'
})
export class ClickWithWarningDirective implements OnInit {
  @Input() warning = 'Are you sure?';
  @Output() appClickWithWarning = new EventEmitter();

  @HostBinding('class') classBinding = 'btn btn-danger';
}
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Bindings

```
@Directive({
  selector: '[appClickWithWarning]'
})
export class ClickWithWarningDirective implements OnInit {
  @Input() warning = 'Are you sure?';
  @Output() appClickWithWarning = new EventEmitter<void>();

  @HostBinding('class') classBinding = 'btn btn-danger';

  @HostListener('click', ['$event'])
  handleClick($event: MouseEvent): void { ... }
}
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Calling the Directive

```
<button (appClickWithWarning)="delete()" warning="Sure?">Delete</button>
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Templates and ViewContainers



# Example: Tooltip

```
[<input [appTooltip]="tmp1">] ← - ViewContainer
```

```
<ng-template #tmp1>  
  <h3>2 Tips for Success</h3>  
  <ol>  
    <li>Don't tell everything!</li>  
  </ol>  
</ng-template>
```



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Implementation

```
@Directive({
  selector: '[appTooltip]'
})
export class TooltipDirective implements OnInit {
  @Input('appTooltip') template: TemplateRef<unknown>;

  constructor(private host: ElementRef,
               private viewContainer: ViewContainerRef) { }

  ngOnInit(): void {
    this.viewContainer.createEmbeddedView(this.template);
  }
}
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Implementation

```
export class TooltipDirective implements OnInit {  
    private viewRef: EmbeddedViewRef<unknown>;  
    @Input('appTooltip') template: TemplateRef<unknown>;  
  
    constructor(  
        private host: ElementRef,  
        private viewContainer: ViewContainerRef) {}  
  
    ngOnInit(): void {  
        this.viewRef = this.viewContainer.createEmbeddedView(this.template);  
        [...]  
    }  
}
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT



# Mouse-Events

```
const elm = this.host.nativeElement as HTMLElement;

elm.addEventListener('mouseover', () => {
  [...]
});

elm.addEventListener('mouseout', () => {
  [...]
});
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Iterate over Projected Root Nodes

```
this.viewRef.rootNodes.forEach(nativeElement => {  
  nativeElement.hidden = true;  
});
```

```
<input [appTooltip]="tmp1">
```

```
<ng-template #tmp1>
```

```
  <h3>2 Tips for Success</h3> ← == -- Root Nodes
```

```
  <ol> ← -----  
    <li>Don't tell everything!</li>
```

```
  </ol>
```

```
</ng-template>
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Adding Templates and Components on the Fly



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# ViewController

- `createEmbeddedView`
- `createComponent`



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# TemplateOutletDirective

Insert Template in Placeholder

```
<div *ngTemplateOutlet="#tmpl"></div>  
<ng-template #tmpl>Hello world!</ng-template>
```



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# ComponentOutletDirective

Insert Component in Placeholder

```
<div *ngComponentOutlet="FlightSearchComponent"></div>
```



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# ContentChildren

- **@ContentChildren(MyComponentOrDirective, { read: ElementRef | ViewContainerRef })**
- Same for @ContentChild, @ViewChildren, @ViewChild



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**



# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Structural Directives



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Structural Directive

## Micro Syntax

```
<div *ngFor="let f of flights; index as i">  
  <pre>{{i}}: {{ f | json }}</pre>  
</div>
```

Template



ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Structural Directive

## ngFor Implementation

```
<div *ngFor="let f of flights; index as i">  
  <pre>{{i}}: {{ f | json }}</pre>  
</div>
```

Template

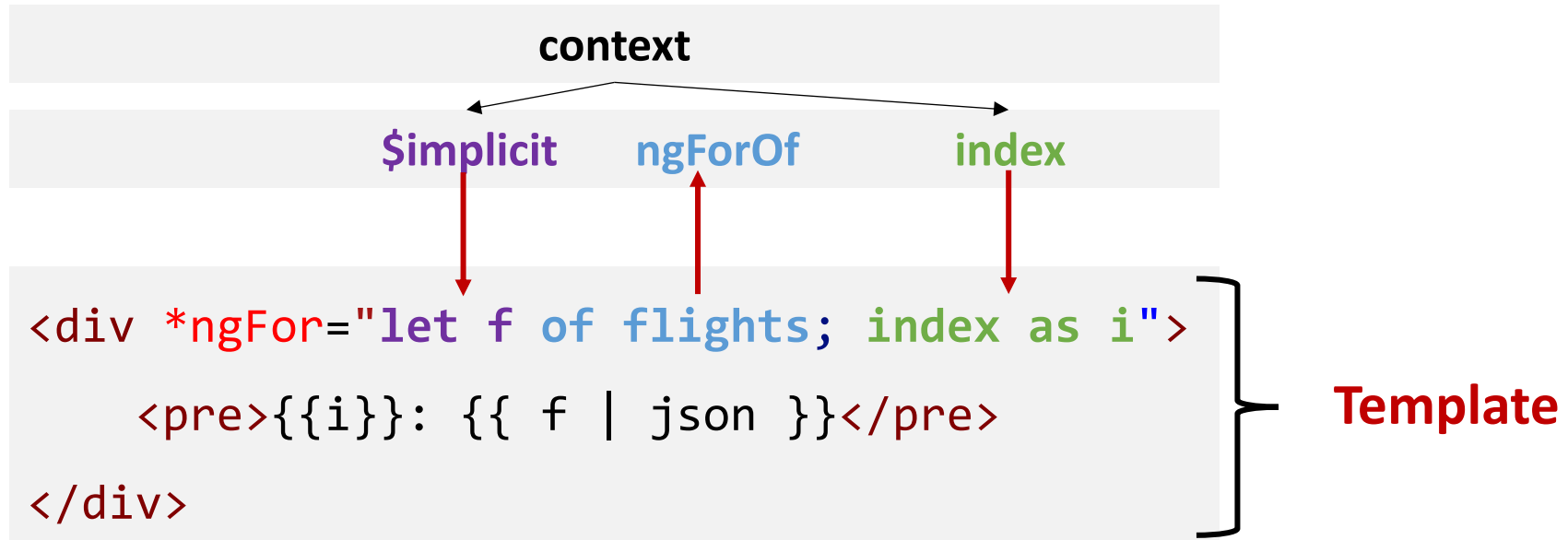


ANGULAR  
ARCHITECTS  
INSIDE KNOWLEDGE



SOFTWARE  
ARCHITECT

# Structural Directive



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Syntax Sugar

```
<div *ngFor="let f of flights; index as i">  
  <pre>{{i}}: {{ f | json }}</pre>  
</div>
```

```
<ng-template ngFor let-f [ngForOf]="flights" let-i="index">  
  <div>  
    <pre>{{i}}: {{ f | json }}</pre>  
  </div>  
</ng-template>
```



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# Case Study #2: DataTable

## Upcoming Flights

1	Hamburg	Berlin	01.02.2025 17:00
2	Hamburg	Frankfurt	01.02.2025 17:30
3	Hamburg	Mallorca	01.02.2025 17:45



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**

# DataTable

```
<app-data-table [data]="flights">
  <div *appTableField="let data as 'id'">{{data}}</div>
  <div *appTableField="let data as 'from'">{{data}}</div>
  <div *appTableField="let data as 'to'">{{data}}</div>
  <div *appTableField="let data as 'date'">
    {{data | date:'dd.MM.yyyy HH:mm'}}
  </div>
</app-data-table>
```



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**



# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# LAB



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Summary

- Attribute Directive with Input, Output, HostBinding, HostListener
- ElementRef, TemplateRef
- ViewContainerRef, EmbeddedViewRef
- \*ngTemplateOutlet, \*ngComponentOutlet
- Structural Directive: Template + Micro Syntax



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE



SOFTWARE  
**ARCHITECT**