



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Directives Deep Dive

[ANGULARarchitects.io](https://ANGULARarchitects.io)

# Contents

- Attribute Directives
- Templates and View Containers
- Structural Directives

# Directives

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

# Attribute Directives



# Case Study

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

# Simple Example

```
@Directive({  
  selector: '[appClickWithWarning]'  
})  
export class ClickWithWarningDirective implements OnInit {  
  
  constructor(private elementRef: ElementRef) { }  
  
  ngOnInit(): void {  
    this.elementRef  
      .nativeElement.setAttribute('class', 'btn btn-danger');  
  }  
}
```

# Calling a Directive

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



**Host-Element**

# Bindings

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



# Bindings

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

# Bindings

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

  @Input() warning = 'Are you sure?';
  @Output() appClickWithWarning = new EventEmitter();

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

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

}
```

# Calling the Directive

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

# 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>
```

# Implementation

```
@Directive({
  selector: '[appTooltip]'
})
export class TooltipDirective implements OnInit {

  @Input('appTooltip') template: TemplateRef<unknown>;

  constructor(private viewContainer: ViewContainerRef) { }

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

}
```

# Implementation

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



# 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>  
  <ol>  
    <li>Don't tell everything!</li>  
  </ol>  
</ng-template>
```

# Mouse-Events

```
@HostListener('mouseover')  
mouseOver(): void {  
    [...]  
}
```

```
@HostListener('mouseout')  
mouseOut(): void {  
    [...]  
}
```

# Passing Parameters to Templates

```
this.viewContainer.createEmbeddedView(this.template, {  
  $implicit: 'Tooltip!',  
  helpLink: 'http://www.google.com'  
});
```

← Context

# Using Parameters in Template

```
<ng-template #tmp1 let-title let-link="helpLink">  
  <h3>{{title}}</h3>  
  
  <p>  
    Deletes EVERYTHING!  
  </p>  
  
  <a [href]="link">More</a>  
  
</ng-template>
```

# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# LAB



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Adding Templates and Components on the Fly



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# ViewContainer

- `createEmbeddedView`
- `createComponent`



# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# TemplateOutletDirective

Insert Template in Placeholder

```
<div *templateOutlet="tpl"></div>  
<ng-template #tpl>Hallo Welt!</ng-template>
```

# ComponentOutletDirective

Insert Component in Placeholder

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

# ContentChildren

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

# 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

# Structural Directive

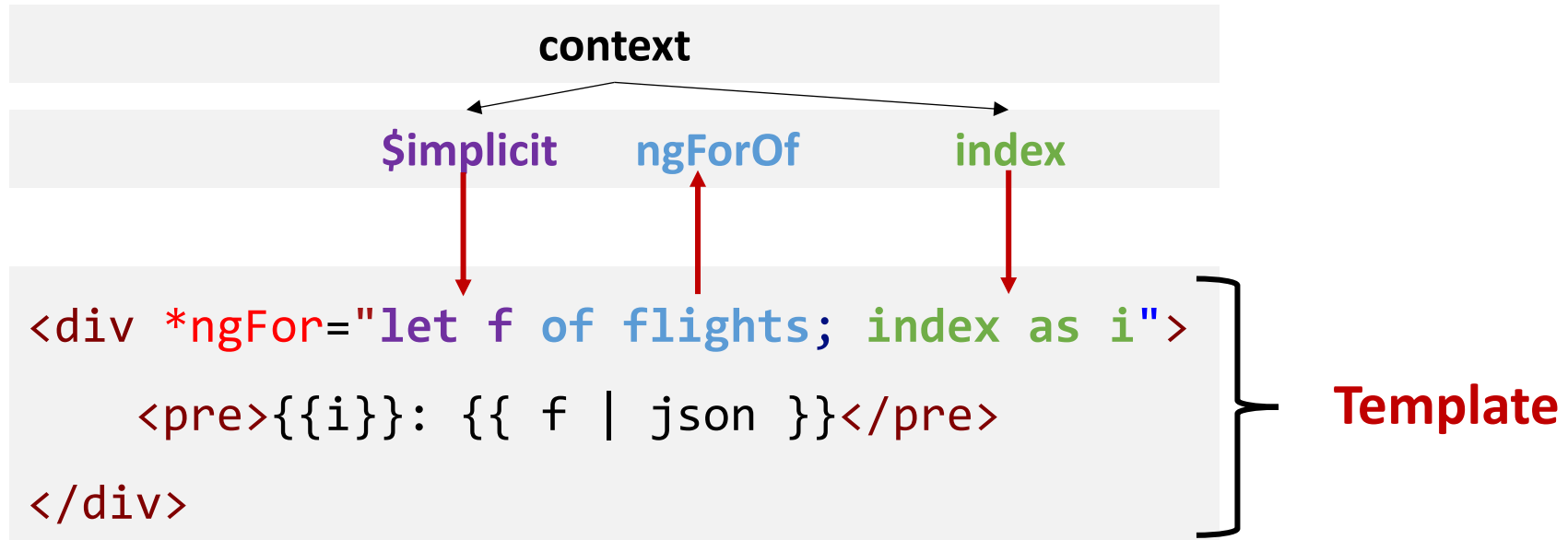
## ngFor Implementation

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

Template



# Structural Directive



# 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>
```

# 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

# 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>
```

# DEMO



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# LAB



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

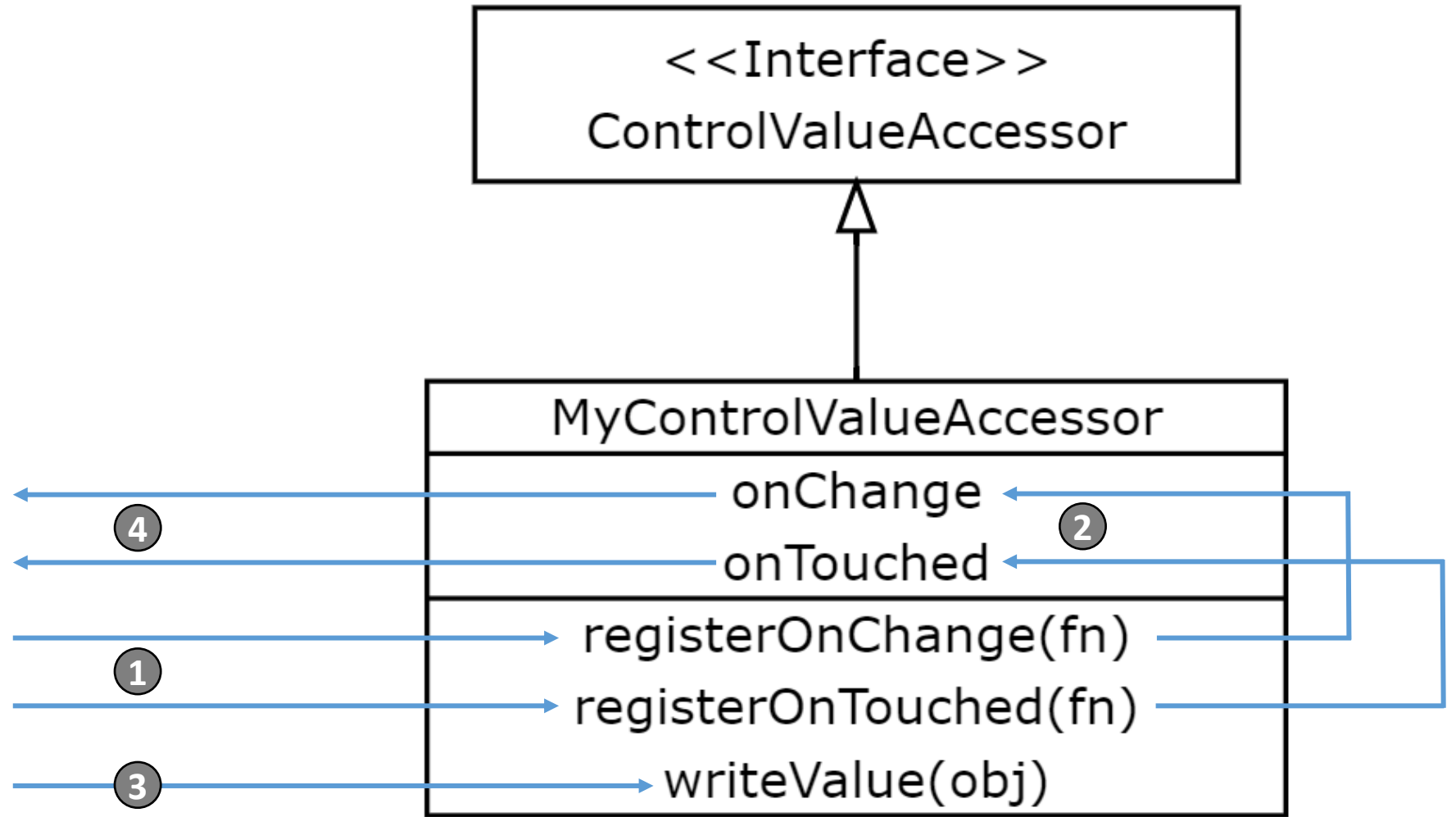
# Summary

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

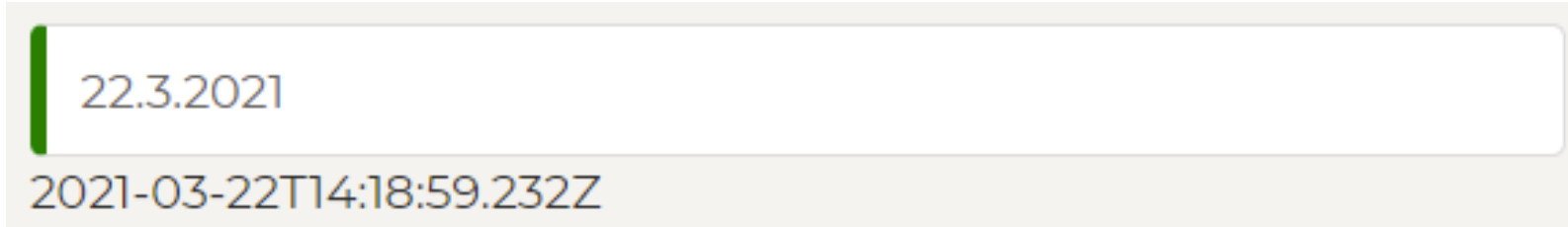
# Custom Form Controls with Control Value Accessors







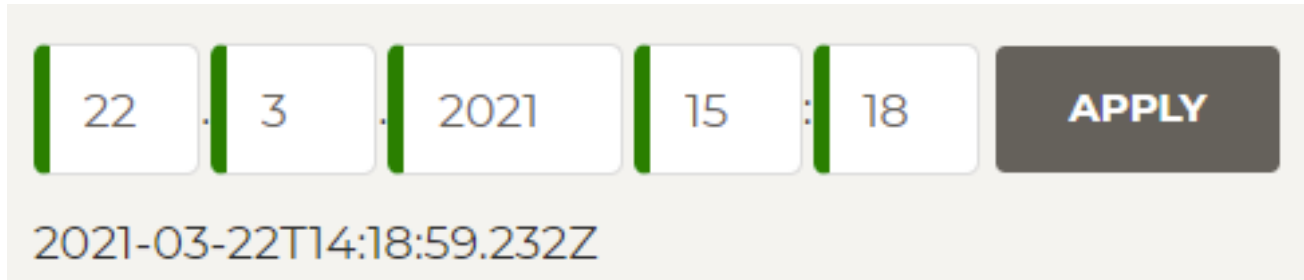
# Case Study #3: Formatting/Parsing Dates



A screenshot of a web form element. It features a white input field with a green vertical bar on the left. The input field contains the text '22.3.2021'. Below the input field, on a light gray background, is the ISO 8601 date string '2021-03-22T14:18:59.232Z'.

```
<input [(ngModel)]="date" appDate name="date">
```

# Case Study #3a: DateControl



22 . 3 . 2021 15 : 18 APPLY

2021-03-22T14:18:59.232Z

```
<app-date [(ngModel)]="date"></app-date>
```