



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Components Deep Dive

Alex Thalhammer

# Contents

- Interacting with Content
- Interacting with View
- Working with Handles

# Why is this interesting?

- Reusable Components (Component Libraries)
- Better Understanding for Angular

# Interacting with a Component's Content

# Case Study #1: Tabbed Pane

Upcoming Flights	Operated Flights	Cancelled Flights
<h2>Upcoming Flights</h2>		
1	Hamburg	Berlin
2	Hamburg	Frankfurt
3	Hamburg	Mallorca

# Tabbed Pane

```
<app-tabbed-pane>
  <app-tab title="Upcoming Flights">
    <p>No upcoming flights!</p>
  </app-tab>

  <app-tab title="Operated Flights">
    <p>No operated flights!</p>
  </app-tab>

  <app-tab title="Cancelled Flights">
    <p>No cancelled flights!</p>
  </app-tab>
</app-tabbed-pane>
```

# DEMO

# View vs. Content

# View vs. Content

```
@Component({  
  selector: 'tab',  
  template: `  
    <div *ngIf="visible">  
      <h1>{{title}}</h1>  
      <div>  
        <ng-content></ng-content>  
      </div>  
    </div>  
` })  
export class TabComponent {  
  @Input() title = "";  
  protected visible = true;  
}
```

View

```
<tab title="Booked">  
  Sample Text ...  
</tab>
```

Content

# Hooks

- 1) ngOnChanges
- 2) ngOnInit
- 3) ngDoCheck
- 4) ngAfterContentInit
- 5) ngAfterContentChecked
- 6) ngAfterViewInit
- 7) ngAfterViewChecked
- 8) ngOnDestroy

# Hooks

- 1) ngOnChanges
- 2) ngOnInit
- 3) ngAfterContentInit**
- 4) ngAfterViewInit**
- 5) ngOnDestroy

# DEMO

# Handles



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Template reference variables

```
<app-tabbed-pane #pane>  
  [...]  
</app-tabbed-pane>
```

```
Current Page: {{ pane.currentPage }}
```

# Don't confuse them with local tpml. vars

```
@let page = 0;
```

# DEMO

# LAB

# Summary

- Content vs. View
- [Content|View][Child | Children]
- Handles