ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Components Deep Dive

Alex Thalhammer

Contents

- Interacting with Content
- Interacting with View
- Working with Handles
- Providers vs. ViewProviders

Why is this Intersting?

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

Interacting with a Component's Content

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Case Study #1: Tabbed Pane

Upcoming Flights | Operated Flights | Cancelled Flights

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

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

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

View vs. Content

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

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

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Handles

Handles

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

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

DEMO

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

ViewProviders

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Providers

```
@Component({
    providers: [NavigatorService]
})
export class TabbedPaneComponent {
    […]
}
```

Providers

```
@Component({
    providers: [NavigatorService]  // Visible in View and Content
})
export class TabbedPaneComponent {
    […]
}
```

View Providers

```
@Component({
    viewProviders: [NavigatorService] // Visible only in View
})
export class TabbedPaneComponent {
    […]
}
```

LAB

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Thought experiment

- What if <app-flight-card> would handle use case logic?
  - e.g. communicate with API
- Number of requests ==> Performance?
- Traceability?
- Reusability?

Smart vs. Dumb Components

Smart / Controller

- 1 per feature / use case / route
- Business logic
- Container

Dumb / Presentational

- Independent of Use Case
- Reusable
- Leave

Summary

- Content vs. View
- [Content|View][Child|Children]
- Handles
- Providers vs. ViewProviders
- Smart vs Dumb Components