Angular: First Steps

Manfred Steyer
SOFTWARE_architekt.at_

Agenda

- Motivation
- First steps
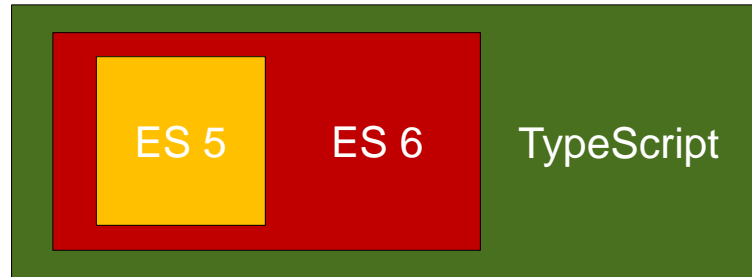- Our first Components
- HTTP access

SOFTWARE_architekt.at_

Motivation

SOFTWAREarchitekt.at

Google	Community
AngularJS (1): >1,3 Mio Devs	Angular (2+): >1M Devs

Programming languages

ES 5	ES 6	TypeScript

← Compiling

SOFTWARE_architekt.at_

First steps with Angular

AppComponent

```
@Component({
    selector: 'flight-app',
    templateUrl: './app.component.html'
})
export class AppComponent {
    title = 'Hello World!';
}
```

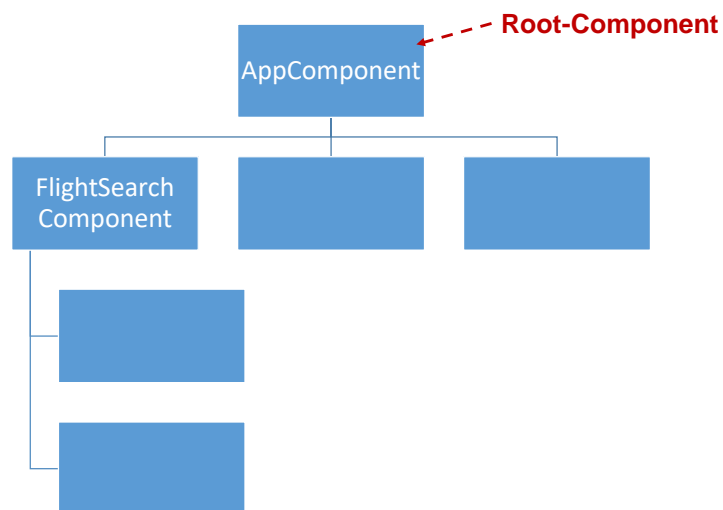**SOFTWARE** *architekt.at*

AppComponent

```
import { Component } from '@angular/core';

@Component({
    selector: 'flight-app',
    templateUrl: './app.component.html'
})
export class AppComponent {
    title = 'Hello World!';
}
```

Library
Example: @angular/core

Custom project
Example: ../entities/flight
No file extension .ts

SOFTWARE *architekt.at*

AppComponent

```typescript
import { Component } from '@angular/core';

@Component({
    selector: 'flight-app',
    templateUrl: './app.component.html'
})
export class AppComponent {
    title = 'Hello World!';
}
```
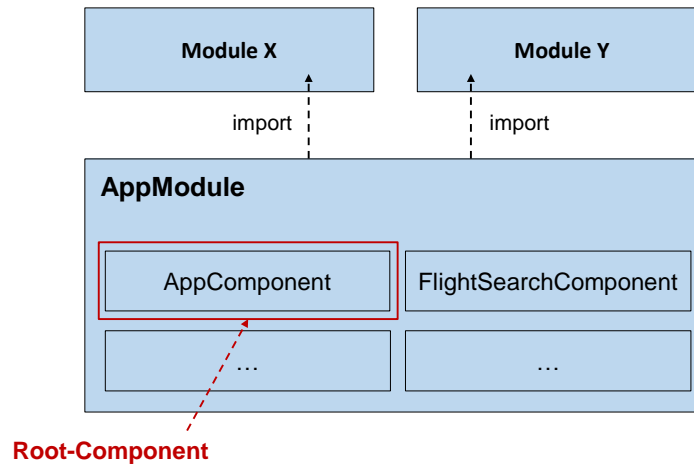
```html
<h1>{{title}}</h1>
<div class="container">
    <flight-search></flight-search>
</div>
```

SOFTWARE *architekt.at*

Application == Component Tree

SOFTWARE *architekt.at*

5

Module

| Module X | Module Y |

import ⋮   import ⋮

AppModule

| AppComponent | FlightSearchComponent |
| … | … |

Root-Component

SOFTWARE*architekt.at*

AppModule

```
@NgModule({
    imports: [
        BrowserModule, HttpClientModule, FormsModule
    ],
    declarations: [
        AppComponent, FlightSearchComponent
    ],
    bootstrap: [
        AppComponent
    ]
})
export class AppModule {
}
```

SOFTWARE*architekt.at*

Bootstrapping

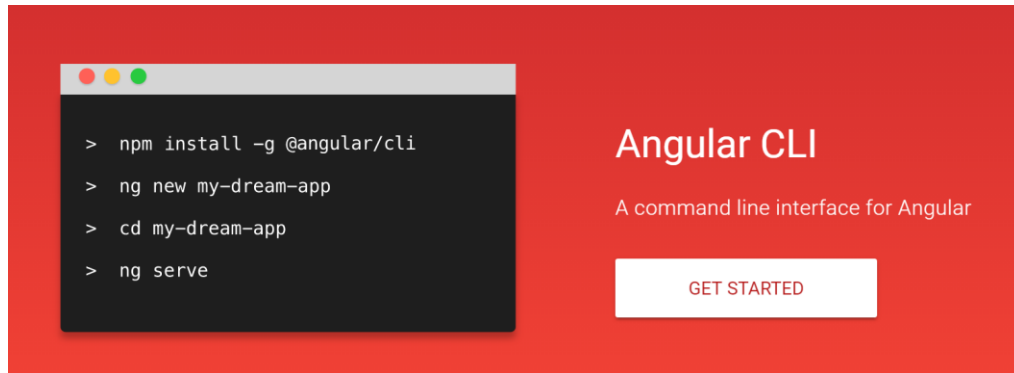- Starting Angular
- Associate RootModule with RootComponent

SOFTWARE*architekt.at*

Bootstrapping

```
platformBrowserDynamic().bootstrapModule(AppModule);
```

SOFTWARE*architekt.at*

index.html

```
[…]

<body>

  <flight-app></flight-app>

  <script src="..."></script>

</body>

[…]
```

SOFTWARE*architekt.at*

Project Start

SOFTWARE*architekt.at*

Angular CLI

How this starter kit was created

- ng new starter
- cd starter
- npm i bootstrap --save
- Add Files for Bootstrap Theme to Folder *assets*
- Insert global styles in .angular-cli.json

```
[…]
"styles": [
    "styles.css",
    "../node_modules/bootstrap/dist/css/bootstrap.css",
    […]
],
[…]
```

SOFTWARE *architekt.at*

DEMO

SOFTWARE*architekt.at*

Our first component

Component as TypeScript Class

```typescript
@Component({
    selector: 'flight-search',
    templateUrl: './flight-search.html'
})
export class FlightSearchComponent {

    from: string;
    to: string;
    flights: Array<Flight>;

    search(): void { [...] }
    select(flight: Flight): void { [...] }
}
```

SOFTWARE *architekt.at*

Template

Two-Way-Binding

Event-Binding

```html
<input [(ngModel)]="from">
<input [(ngModel)]="to">

<button [disabled]="!from || !to" (click)="search()">
    Search
</button>

<table>
    <tr *ngFor="let flight of flights">
        <td>{{flight.id}}</td>
        <td>{{flight.date}}</td>
        <td>{{flight.from}}</td>
        <td>{{flight.to}}</td>
    </tr>
</table>
```
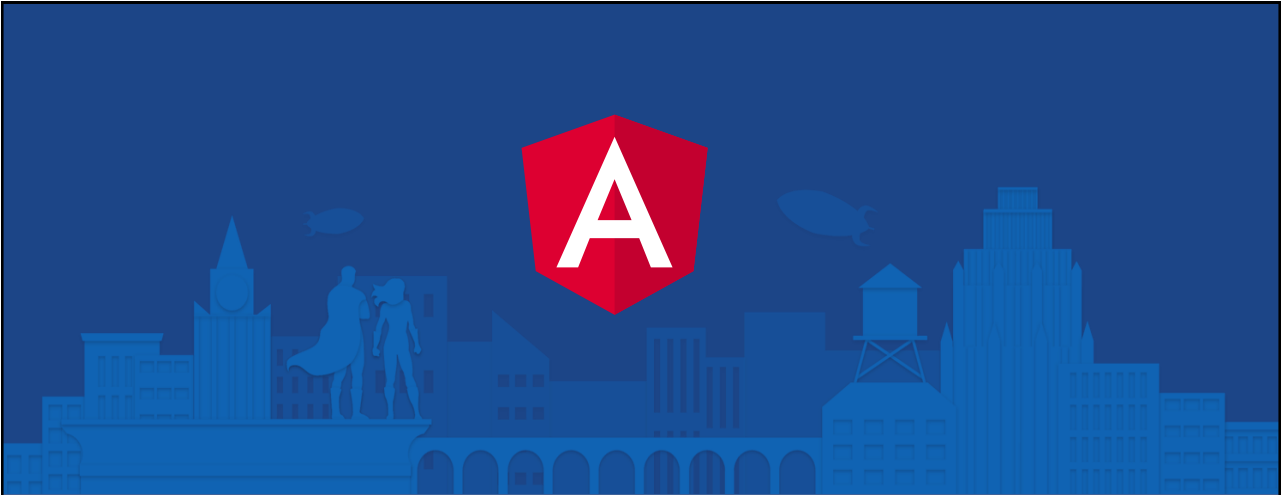
Property-Binding

Template

SOFTWARE *architekt.at*

Access HTTP resources

HttpClient

- get(url, options)
- post (url, body, options)
- put(url, body, options)
- delete(url, options)
- …

HttpClient

- get<T>(url, options)
- post<T>(url, body, options)
- put<T>(url, body, options)
- delete<T>(url, options)
- …

SOFTWARE*architekt.at*

Inject HttpClient

```
@Component({
    selector: 'flight-search',
    templateUrl: './flight-search.html'
})
export class FlightComponent {

    from: string;
    to: string;
    flight: Array<Flight>;

    constructor(http: HttpClient) { […] }

    search(): void { [...] }
    select(flight: Flight): void { [...] }
}
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');

this.http
    .get<Flight[]>(url, { params, headers })
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');

this.http
    .get<Flight[]>(url, { params, headers })
    .subscribe(
        function(flights) { … }
);
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');

let that = this;
this.http
    .get<Flight[]>(url, { params, headers })
    .subscribe(
        function(flights) {
                that.flights = flights;
        }
);
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');

this.http
    .get<Flight[]>(url, { params, headers })
    .subscribe(
        flights => {
                this.flights = flights;
        }
);
```

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');

this.http
    .get<Flight[]>(url, { params, headers })
    .subscribe(
        flights => { this.flights = flights; },
        err => { console.error('Error on loading flights', err); }
    );
```

SOFTWARE*architekt.at*

DEMO

SOFTWARE*architekt.at*

Use HttpClient

```
let url = 'http://www.angular.at/api/flight';

let params = new HttpParams()
                    .set('from', this.from)
                    .set('to', this.to);

let headers = new HttpHeaders()
                    .set('Accept', 'application/json');

this.http
    .get<Flight[]>(url, { params, headers })
    .subscribe(
        flights => { this.flights = flights; },          ◂-- Observable
        err => { console.error('Error on loading flights', err); }
    );
```

SOFTWARE*architekt.at*

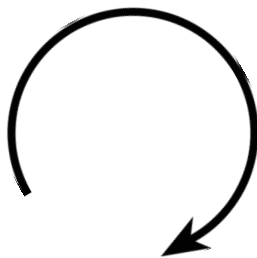Observable „source"

Operator (e.g. map)

Observer „trough"

SOFTWARE *architekt.at*

Observable

Observable

```
.subscribe(
  (result) => { … },
  (error) => { … },
  () => { … }
);
```

Observer

SOFTWARE *architekt.at*

Map operator

```
this
    .http
    .get(…)
    .map(flights => mapToOffers(flights))
    .subscribe(
        (offers) => { … },
        (err) => { console.error(err); }
    );
```

SOFTWARE*architekt.at*

DEMO

SOFTWARE*architekt.at*