Angular: Erste Schritte

Hosted by Alex Thalhammer

Inhalt

- Motivation
- Erste Schritte
- Eine erste Komponente
- HTTP-Zugriff

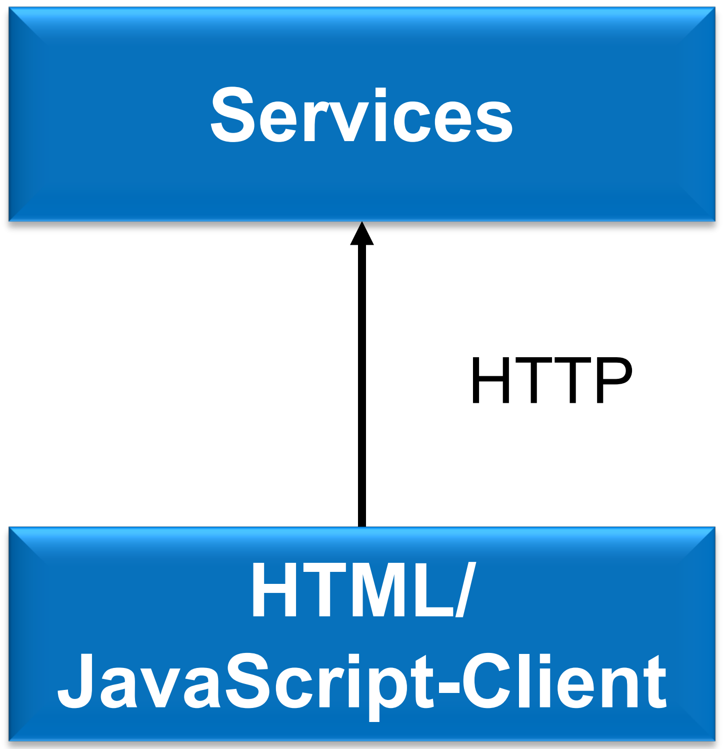
Motivation

Plattformen und Usability

HTML + JavaScript

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Single Page Application (SPA)



HTML + JavaScript = Komplexität

Frameworks machen SPA beherrschbar

Google

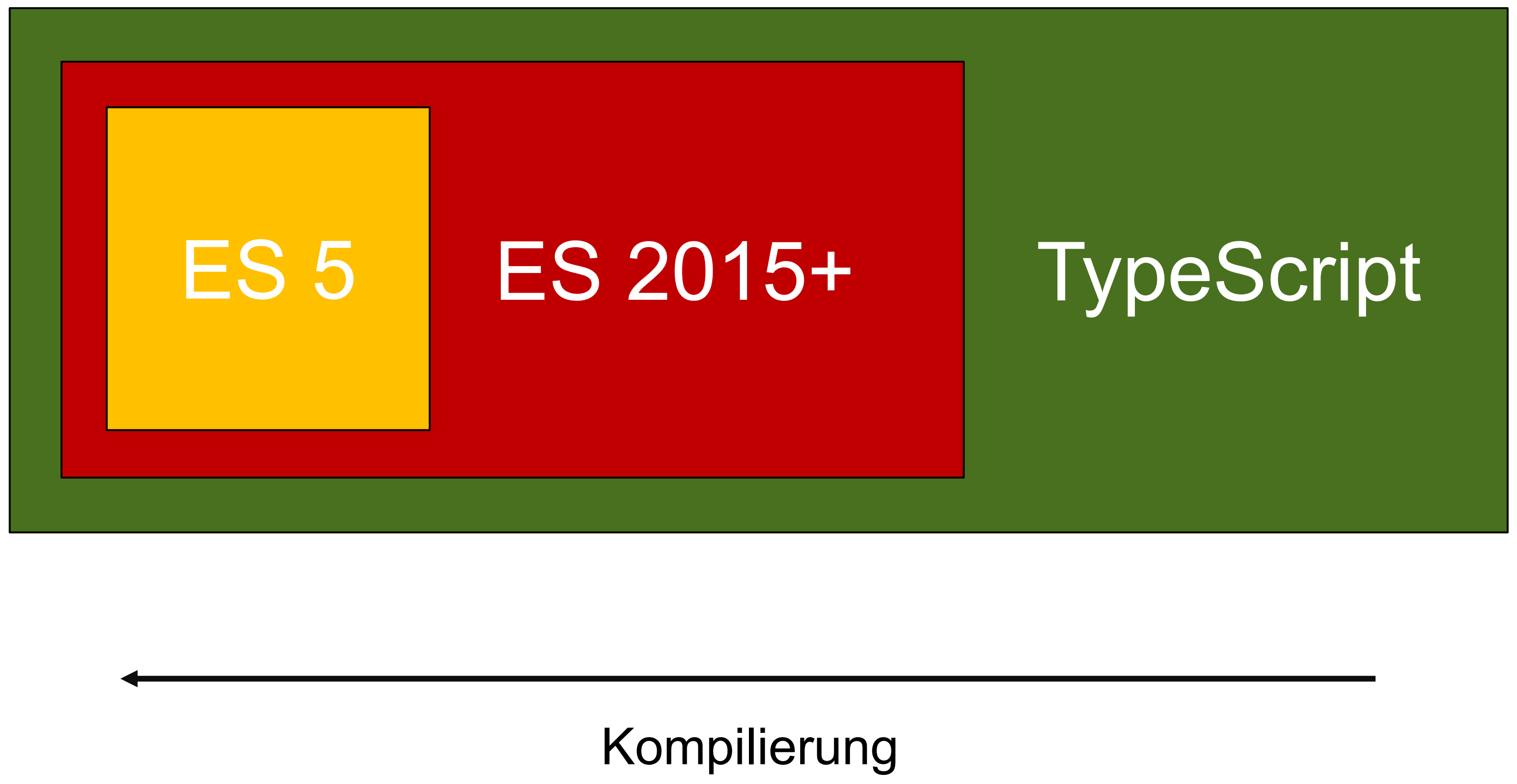
Community

Angular (2/4/5): >1,2M Devs

angular-connect, London

angular-connect, London

Sprachen



Erste Schritte mit Angular

AppComponent

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

AppComponent

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

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

Bibliothek

Beispiel: @angular/core

Eigenes Projekt

Beispiel: ../entities/flug

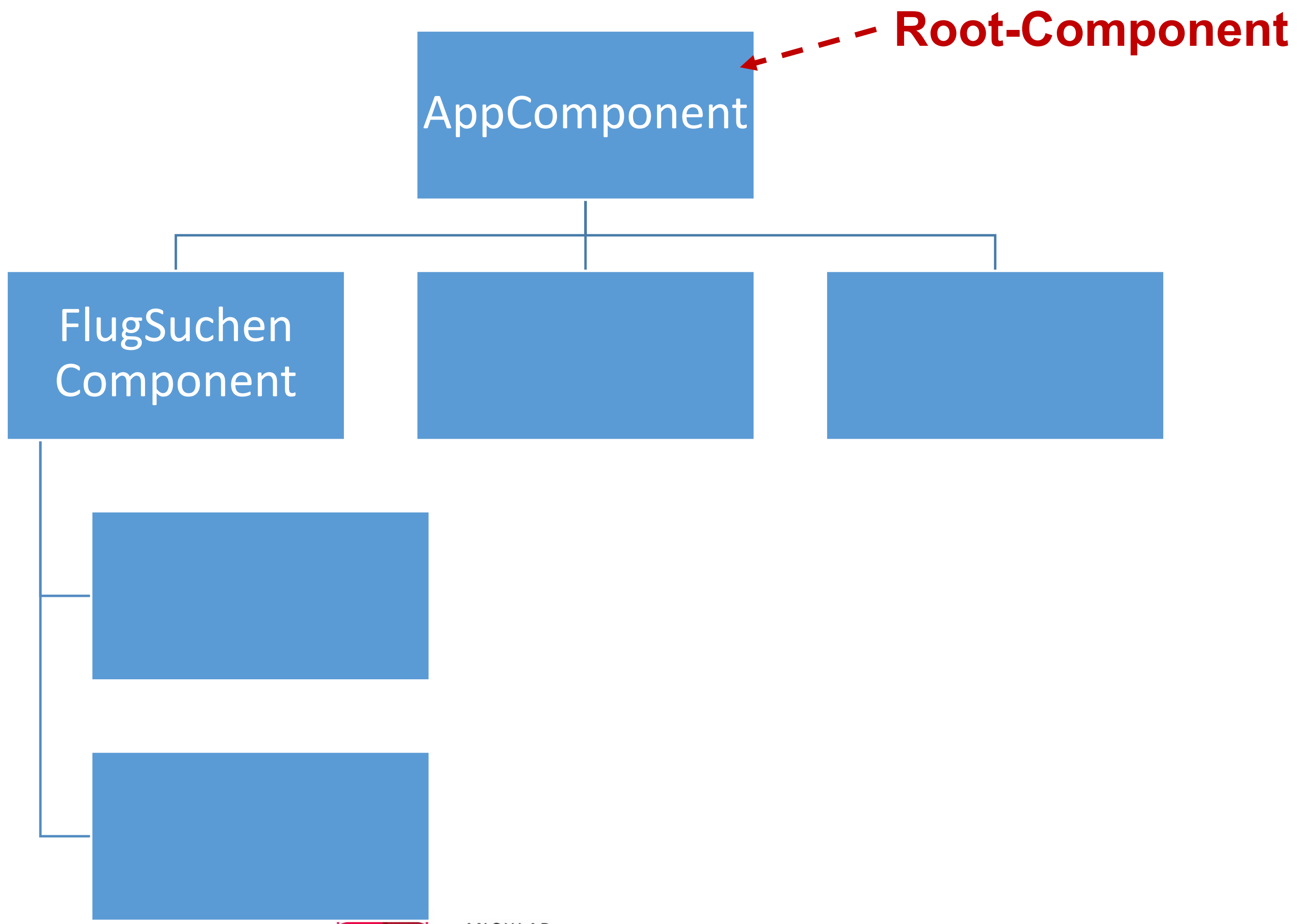
Keine Endung .ts

AppComponent

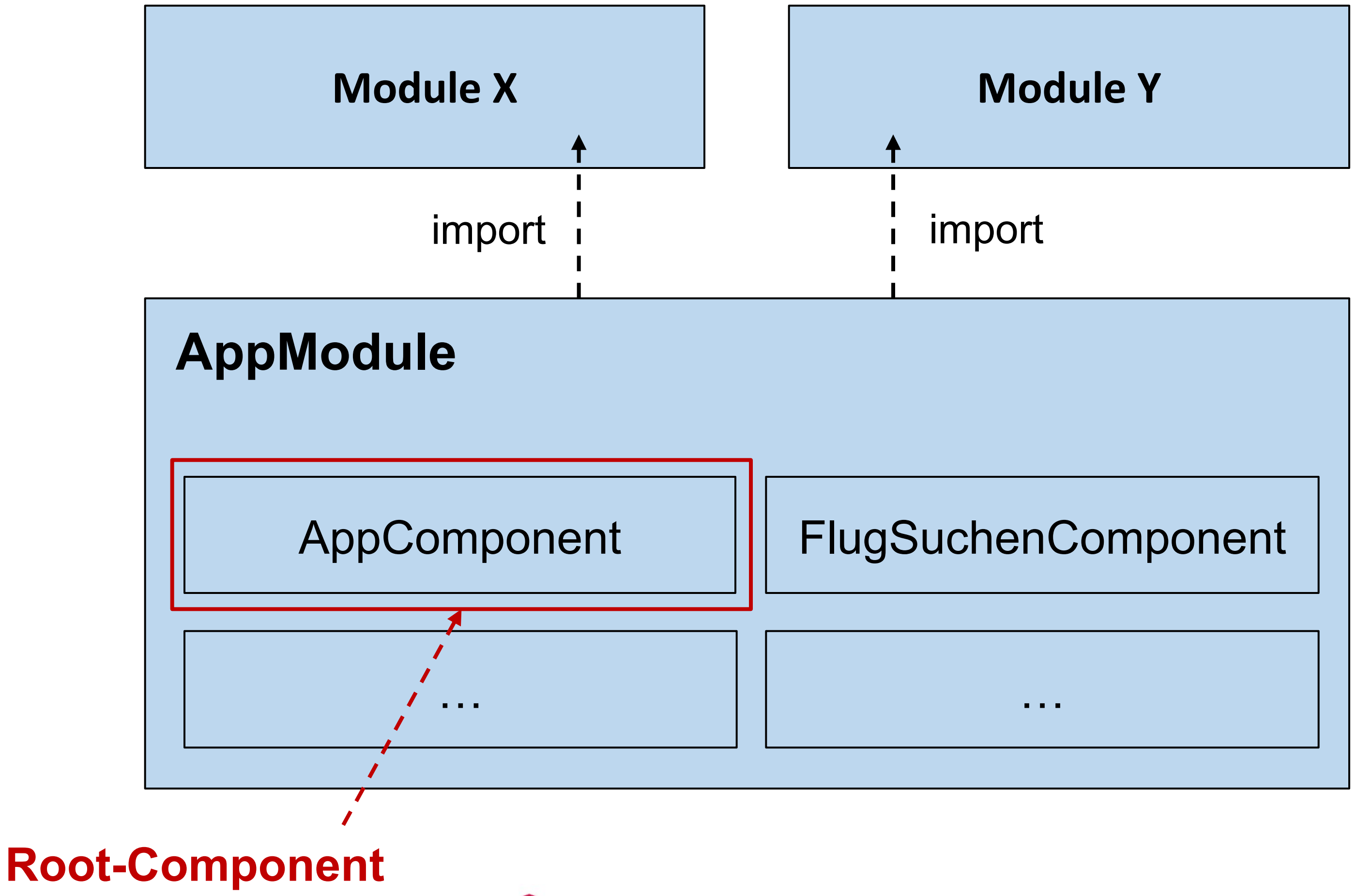
```
import { Component } from '@angular/core';

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

```
<h1>{{title}}</h1>

<div class="container">
    <flug-suchen></flug-suchen>
</div>
```

Anwendung == Kompontentenbaum



Module


Module X
Module Y
import
import
AppModule
AppComponent
FlugSuchenComponent
…
…
Root-Component

AppModule

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

Bootstrapping

- Angular starten
- RootModule mit RootComponent bekannt geben

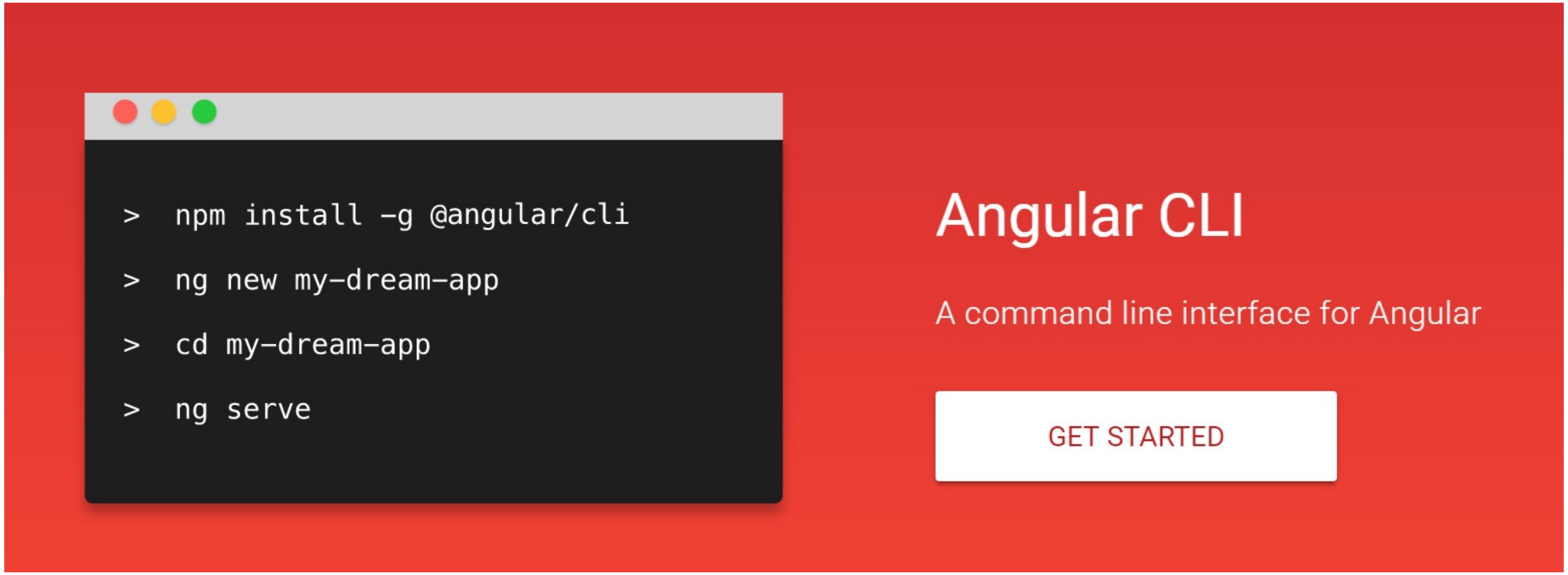
Bootstrapping

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

index.html

```
[…]

<body>

  <flug-app></flug-app>

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

</body>

[…]
```

Projektstart



```
> npm install -g @angular/cli
> ng new my-dream-app
> cd my-dream-app
> ng serve
```

Angular CLI

A command line interface for Angular

Angular CLI

Unser Starterkit

- ng new starter
- cd starter
- npm i bootstrap --save
- Globale Styles in *angular.json* eintragen

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

DEMO

LAB

Let’s get it on

- Pull the repo https://github.com/L-X-T/uni-muenster
- Please get started with lab 00_getting_started
- Yarn (or npm i) and then Yarn start (or npm start)
- Take a closer look at the starter kit
- Optional: You may add the plugins mentioned

Eine erste Komponente

Komponente als TypeScript-Klasse

```
@Component({
    selector: 'flug-suchen',
    templateUrl: './flug-suchen.html'
})
export class FlugSuchenComponent {

    von: string;
    nach: string;
    fluege: Array<Flug>;

    search(): void { [...] }
    select(flug: Flug): void { [...] }
}
```

Template

Two-Way-Binding

Event-Binding

Property-Binding

Template

```
<input [(ngModel)]="von">
<input [(ngModel)]="nach">

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

<table>
    <tr *ngFor="let flug of fluege">
        <td>{{flug.id}}</td>
        <td>{{flug.datum}}</td>
        <td>{{flug.von}}</td>
        <td>{{flug.nach}}</td>
    </tr>
</table>
```

DEMO

Auf HTTP-Ressourcen zugreifen

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)
- …

HttpClient injizieren

```
@Component({
    selector: 'flug-suchen',
    templateUrl: './flug-suchen.html'
})
export class FlugSuchenComponent {

    von: string;
    nach: string;
    fluege: Array<Flug>;

    constructor(http: HttpClient) { […] }

    search(): void { [...] }
    select(flug: Flug): void { [...] }
}
```

HttpClient nutzen

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

HttpClient nutzen

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

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

HttpClient nutzen

```
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');
```

HttpClient nutzen

```
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: params, headers: headers })
```

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

HttpClient nutzen

```
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 })
```

HttpClient nutzen

```
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) { […] }
    );
```

HttpClient nutzen

```
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;
        }
);
```

HttpClient nutzen

```
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;
        }
);
```

HttpClient nutzen

```
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('Fehler beim Laden', err); }
    );
```

DEMO

LAB

HttpClient nutzen

```
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('Fehler beim Laden', err); }
    );
```

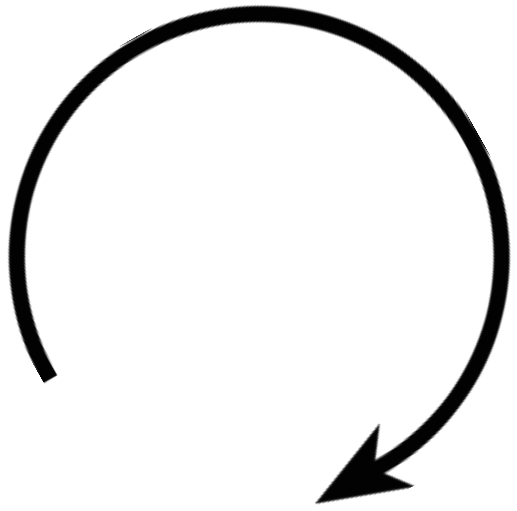
← Observable

Observable
„Quelle“

Operator
(z. B. map)

Observer
„Senke“

Observable



Observable

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

Observer

Direktiven nutzen

Was sind Direktiven?

- Fügen Verhalten zu Elementen hinzu
- Werden häufig in Form von Attributen verwendet
- Beispiele:
  - <input **[(ngModel)]**="from">
  - <div ***ngFor**="let flight of flights">…</div>

Beispiele

```
<tr *ngFor="let flight of flights">
    <td>{{flight.id}}</td>
</tr>
```

```
<table *ngIf="flights.length > 0">
…
</table>
```

```
<tr [ngClass]="{ 'active': flight === selectedFlight }">
…
</tr>
```

```
<tr [ngStyle]="{ 'background-color':
     (flight === selectedFlight) ?
     'orange' : 'blue' }">
</tr>
```

INSIDE KNOWLEDGE

Beispiele

```
<tr *ngFor="let flight of flights">
    <td>{{flight.id}}</td>
</tr>
```

```
<table *ngIf="flights.length > 0">
…
</table>
```

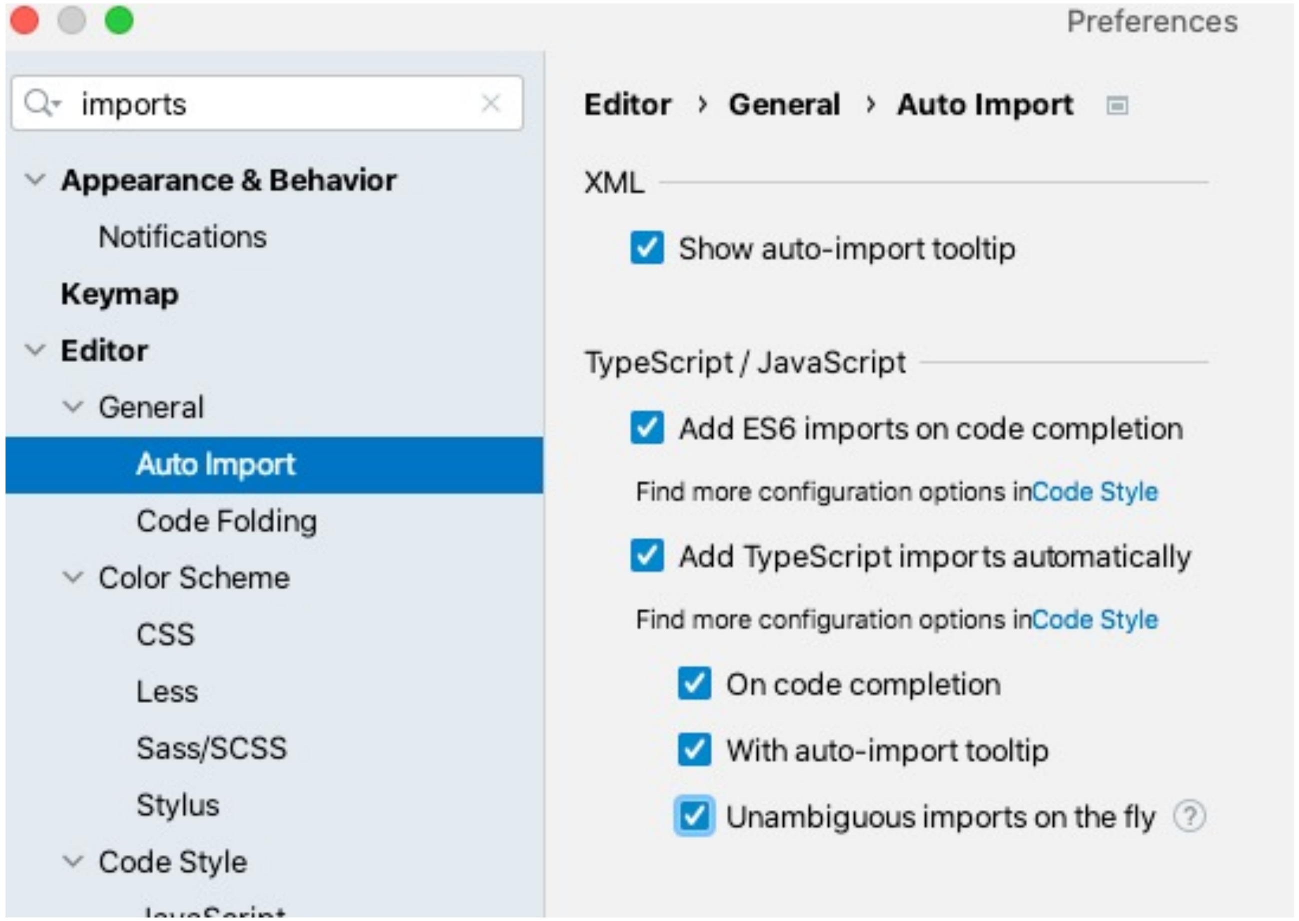
```
<tr [ngClass]="{ 'active': flight === selectedFlight }">
…
</tr>
```

```
<tr [class.active]="flight === selectedFlight">
</tr>
```

```
<tr [ngStyle]="{ 'background-color':
     (flight === selectedFlight) ?
     'orange' : 'blue' }">
</tr>
```

DEMO

One more thing: auto import



Preferences

imports

- Appearance & Behavior
  - Notifications
- Keymap
- Editor
  - General
    - Auto Import
    - Code Folding
  - Color Scheme
    - CSS
    - Less
    - Sass/SCSS
    - Stylus
  - Code Style

Editor › General › Auto Import

XML

- [✓] Show auto-import tooltip

TypeScript / JavaScript

- [✓] Add ES6 imports on code completion

  Find more configuration options inCode Style

- [✓] Add TypeScript imports automatically

  Find more configuration options inCode Style

  - [✓] On code completion
  - [✓] With auto-import tooltip
  - [✓] Unambiguous imports on the fly