



Angular: Erste Schritte

Hosted by Alex Thalhammer

Inhalt

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

Motivation



Plattformen und Usability



HTML + JavaScript

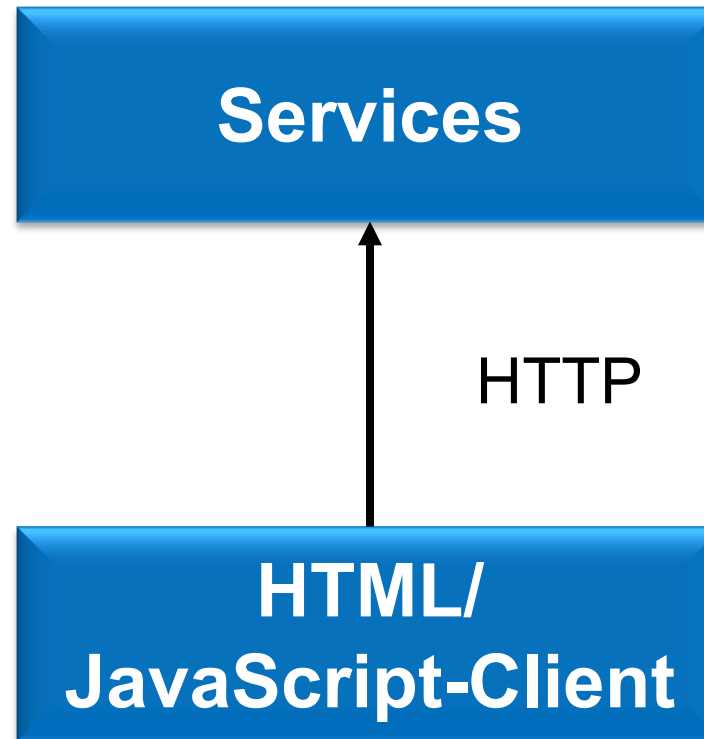


ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

Single Page Application (SPA)





HTML + JavaScript =
Komplexität



Frameworks machen SPA beherrschbar



Google

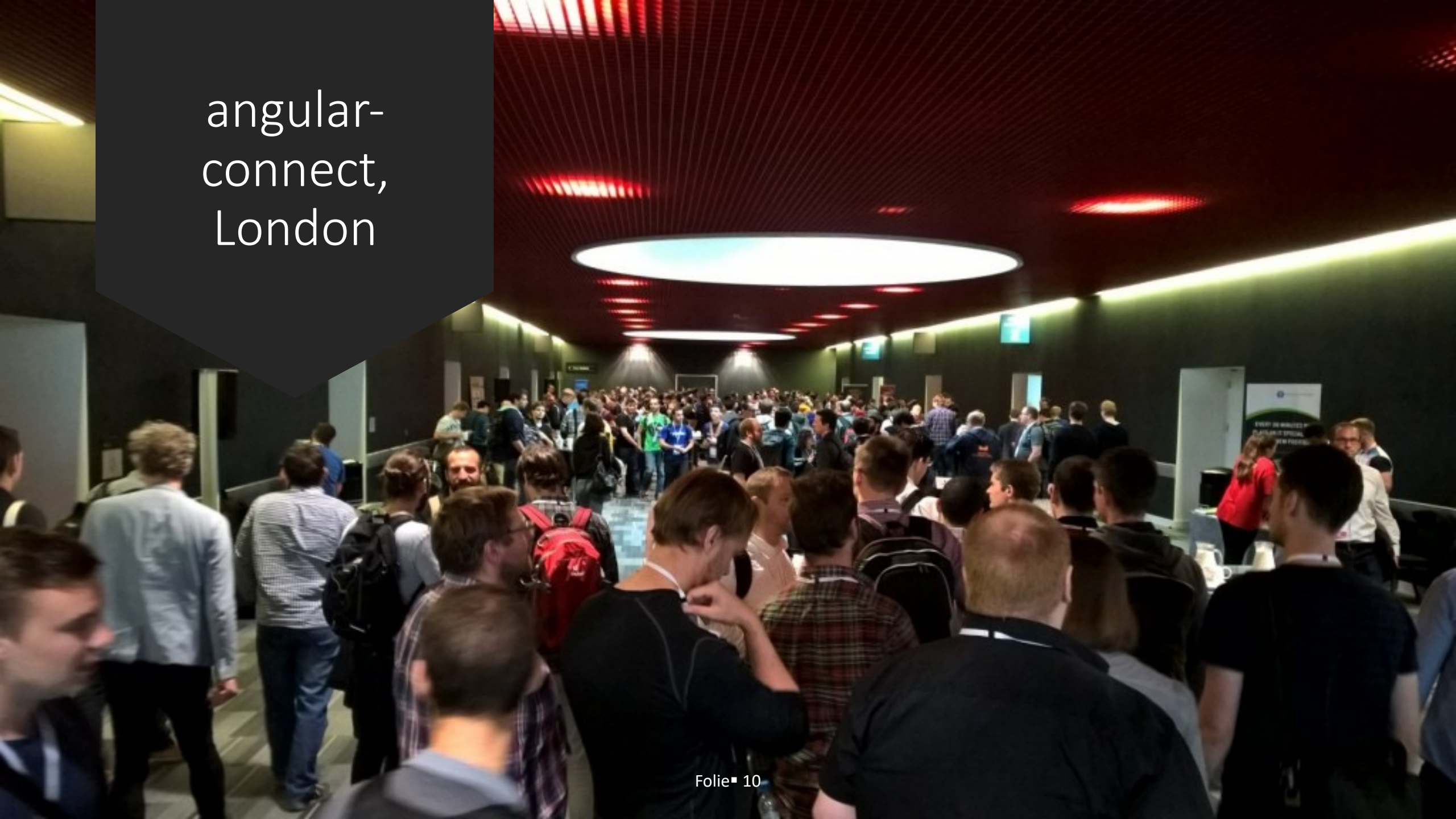
Community

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

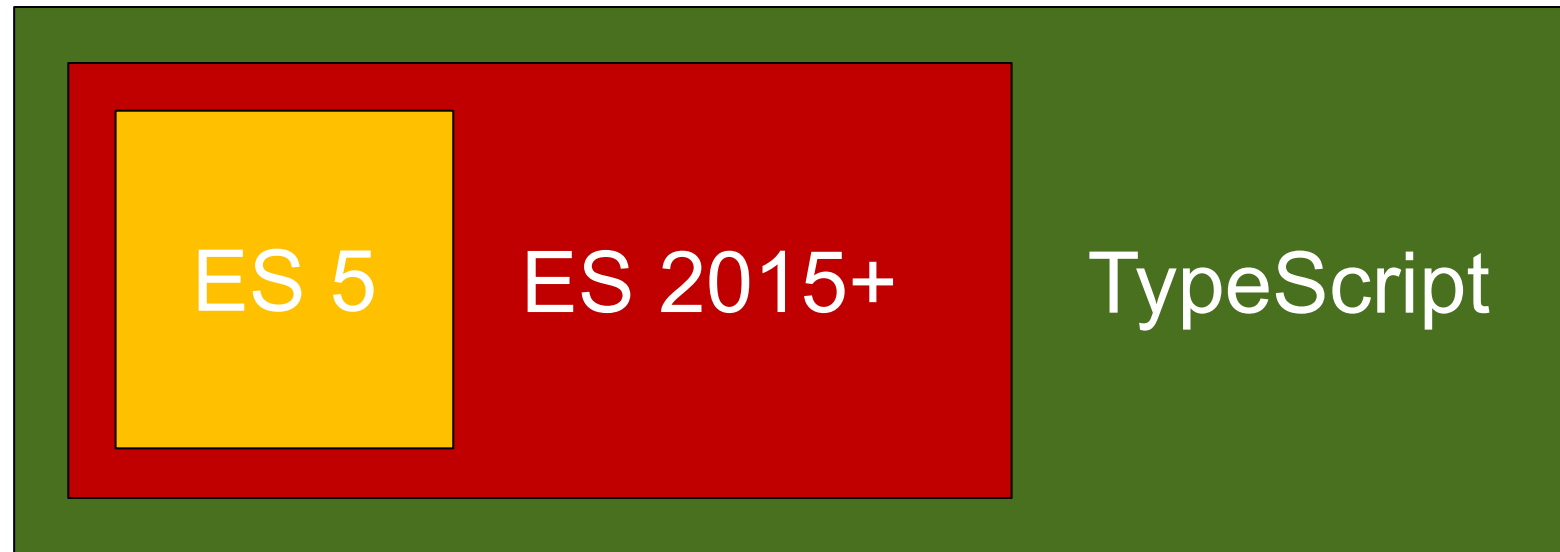
angular- connect, London



angular- connect, London



Sprachen



Kompilierung



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

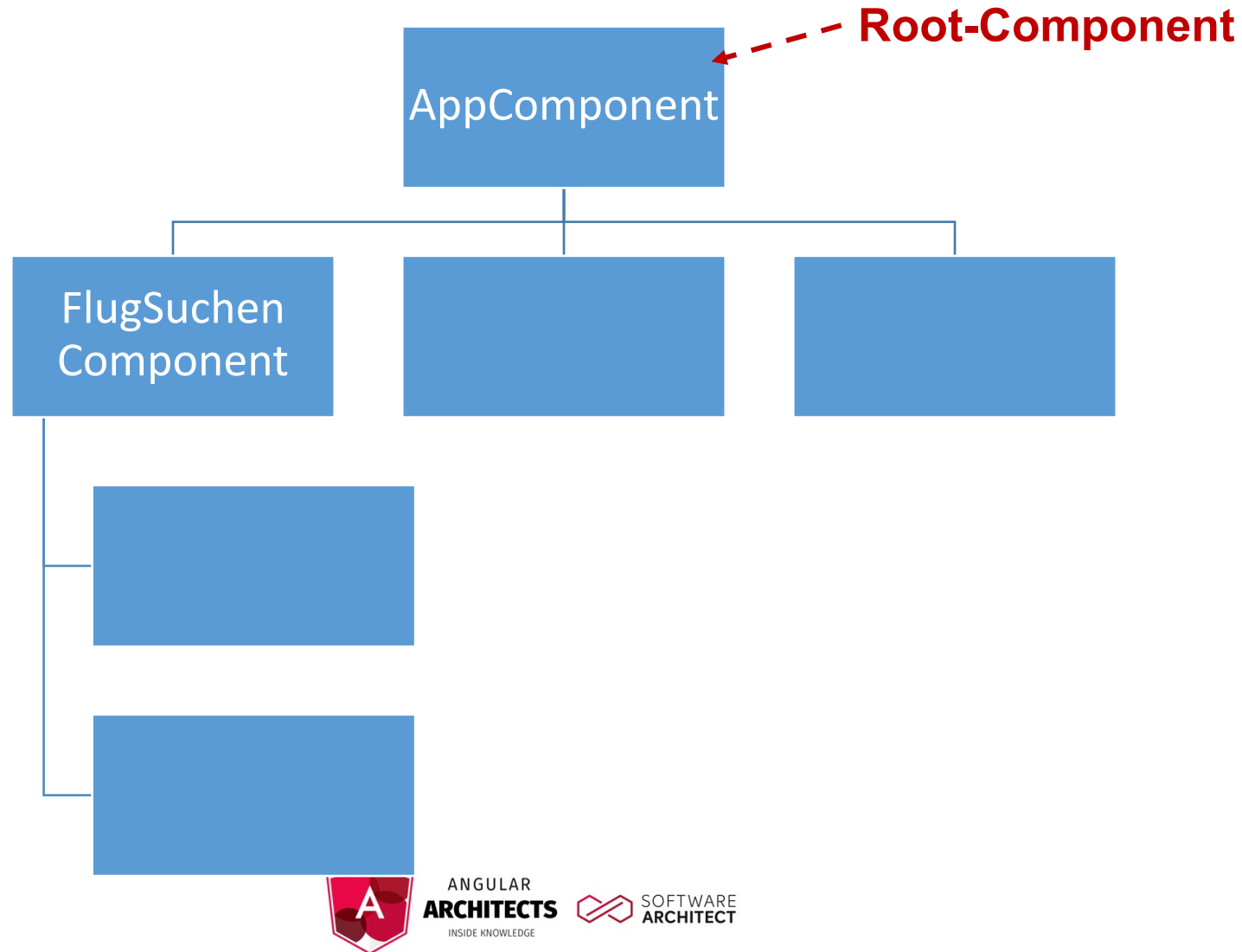
```
import { Component } from '@angular/core';

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

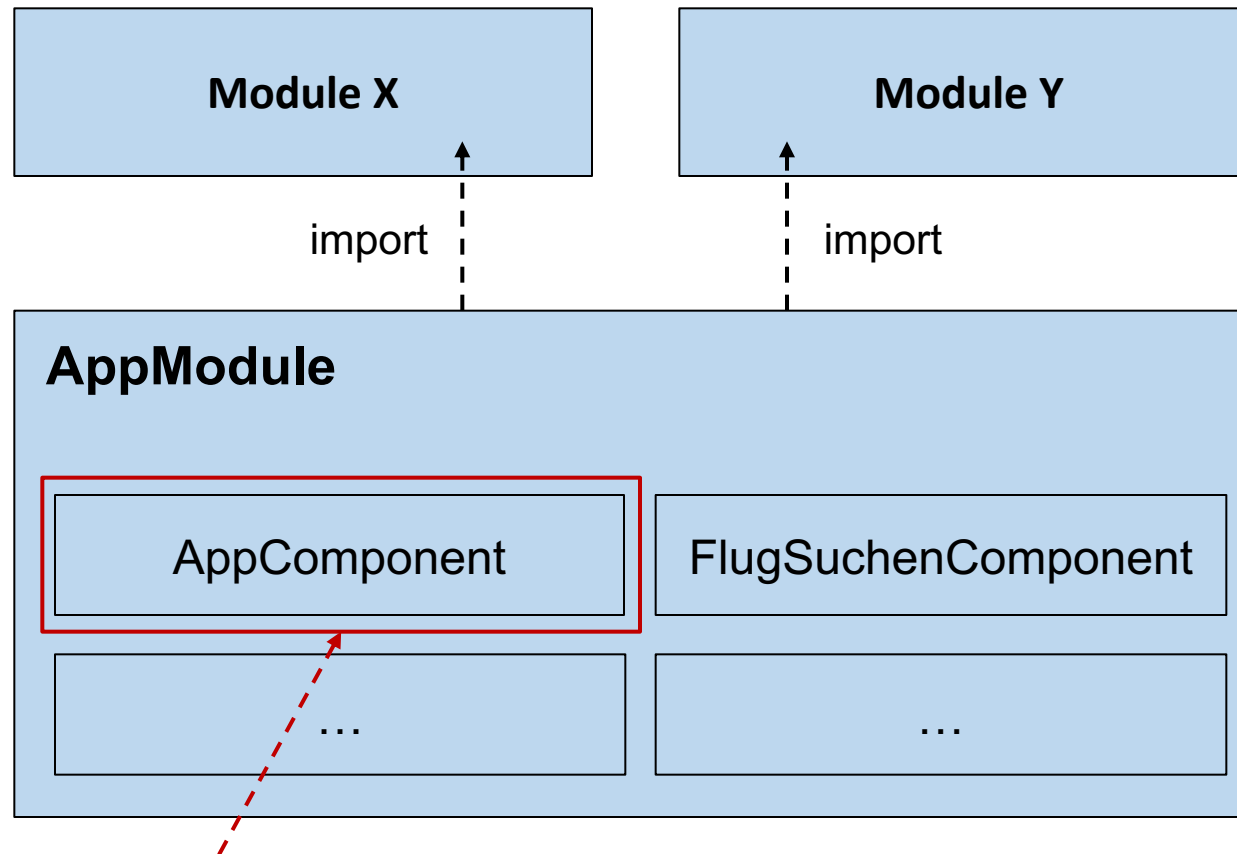
```
<h1>{{title}}</h1>
<div class="container">
  <flug-suchen></flug-suchen>
</div>
```



Anwendung == Komponentenbaum



Module



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>  
[...]
```



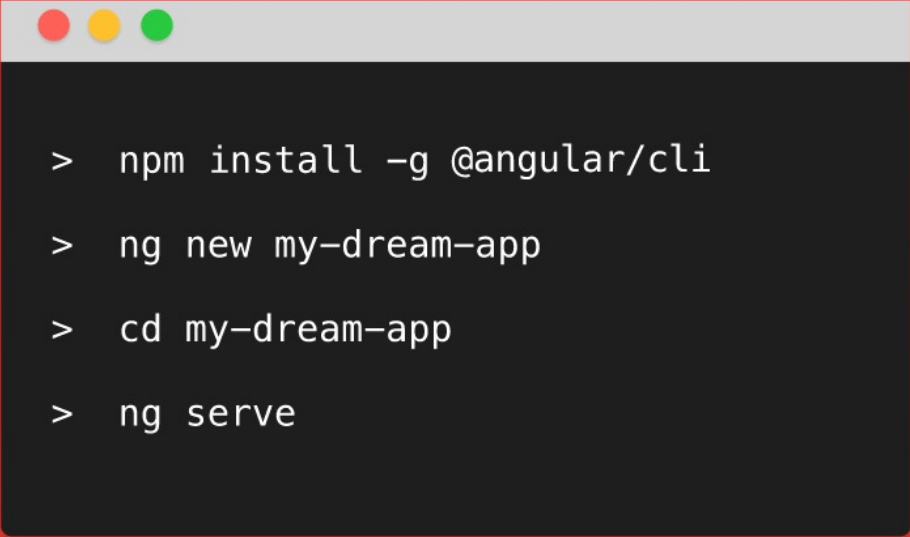
Projektstart



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT



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

Angular CLI

A command line interface for Angular

GET STARTED

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",  
  [...]  
],  
[...]
```



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

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

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

Event-Binding

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

Property-Binding

```
<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>
```

Template



DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT



Auf HTTP-Ressourcen zugreifen



HttpClient

- `get(url, options)`
- `post (url, body, options)`
- `put(url, body, options)`
- `delete(url, options)`
- ...



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

HttpClient

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



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

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';
```



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);
```



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');
```



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: 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 })
```



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 })
    .subscribe(
        function(flights) { [...] }
    );
```



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');

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); }
    );
```



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

LAB



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 })
    .subscribe(
        flights => { this.flights = flights; },
        err => { console.error('Fehler beim Laden', err); }
    );
```

←----- Observable



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

Observable
„Quelle“



Operator
(z. B. map)

Observer
„Senke“

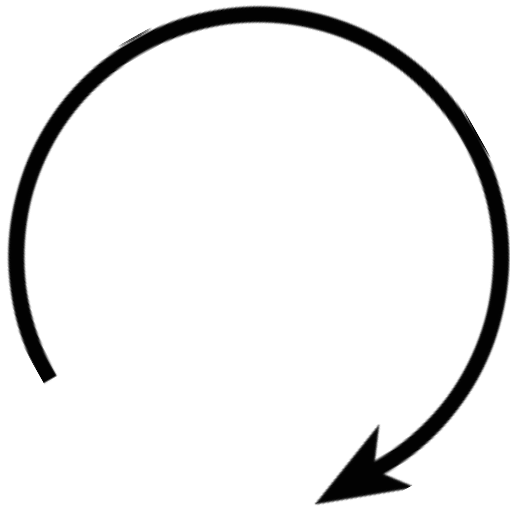


ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

Observable



Observable

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

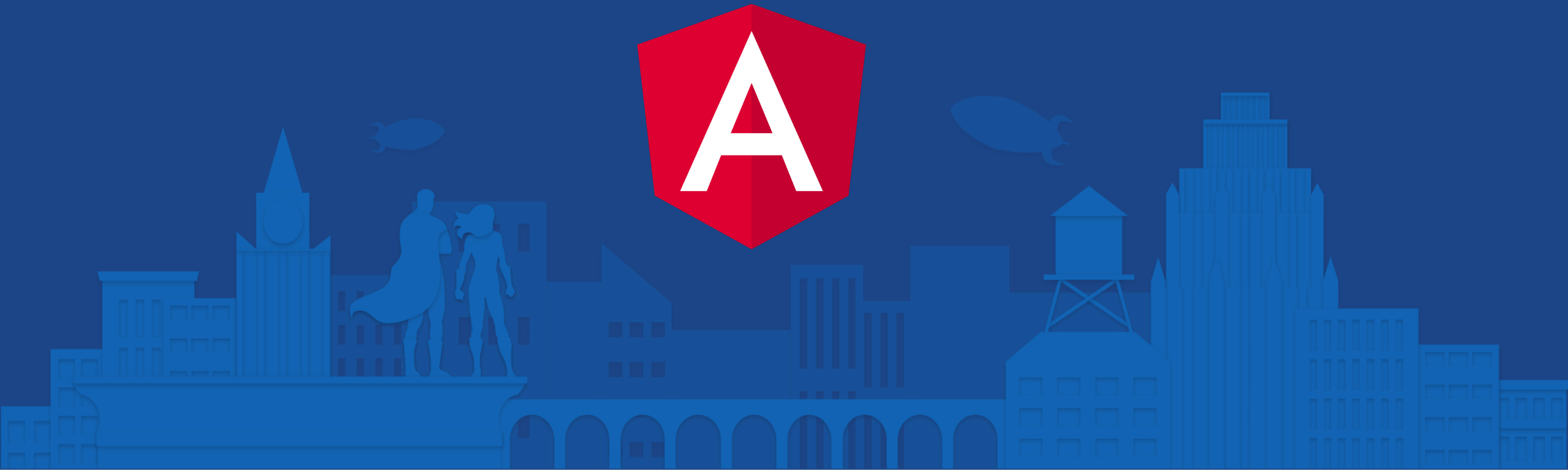
Observer



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT



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>
```



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

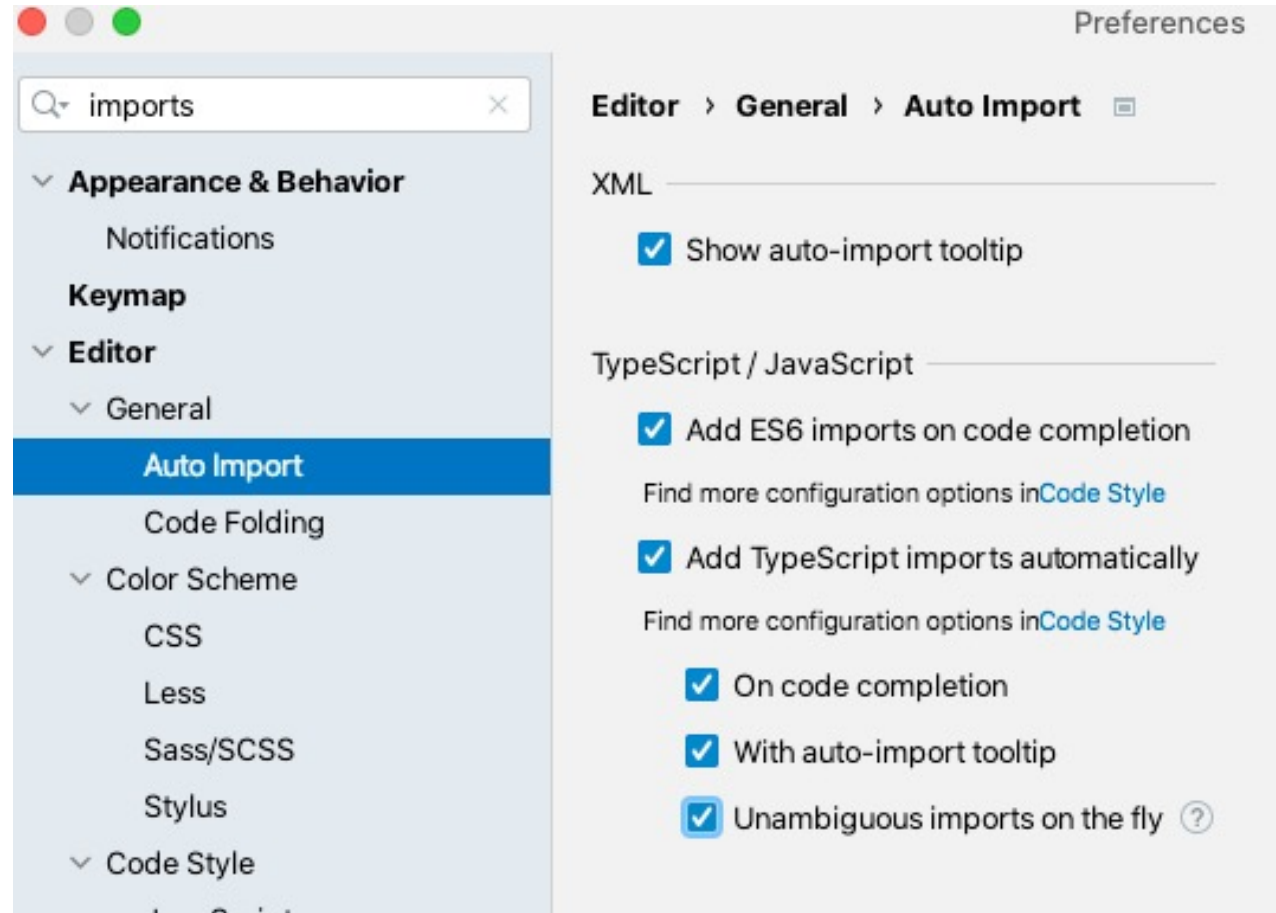


ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

One more thing: PhpStorm auto import



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

