



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Angular Basics & A first Component

Alex Thalhammer

Outline

- Motivation
- First steps
- HTTP access with Angular HTTP client
- Your first Component
- Built-in Angular Directives

Motivation



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

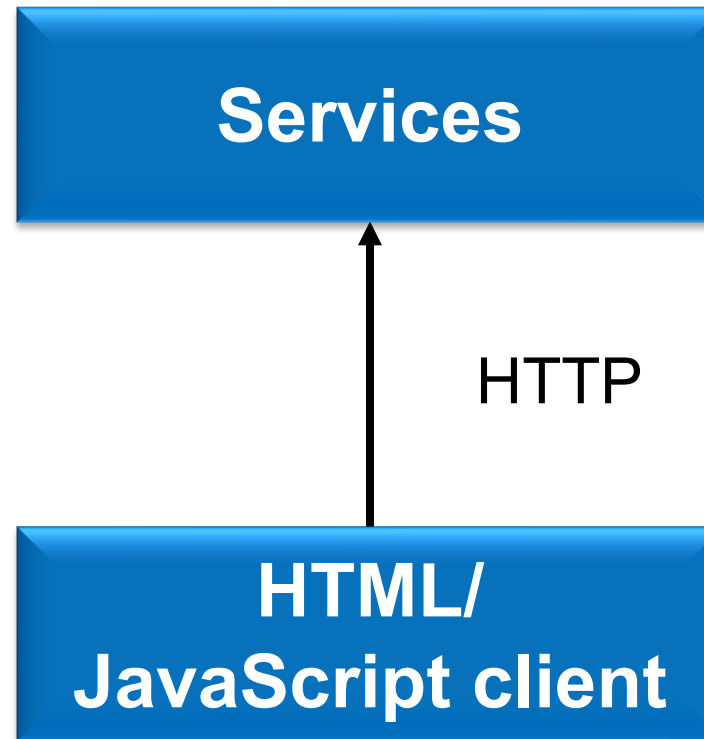
Platforms and Usability



HTML + JavaScript



Single Page Application (SPA)





HTML + JavaScript =
Complexity



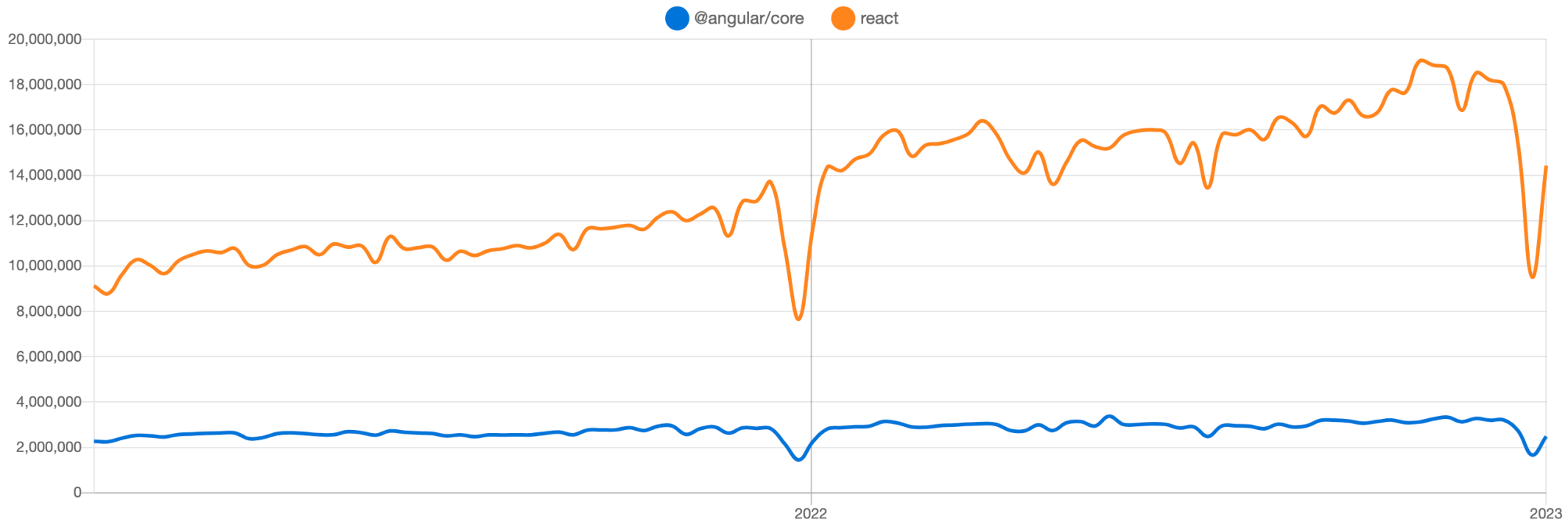
Frameworks make SPA manageable

@angular/core vs react

Enter an npm package...

@angular/core x react x + angular + vue

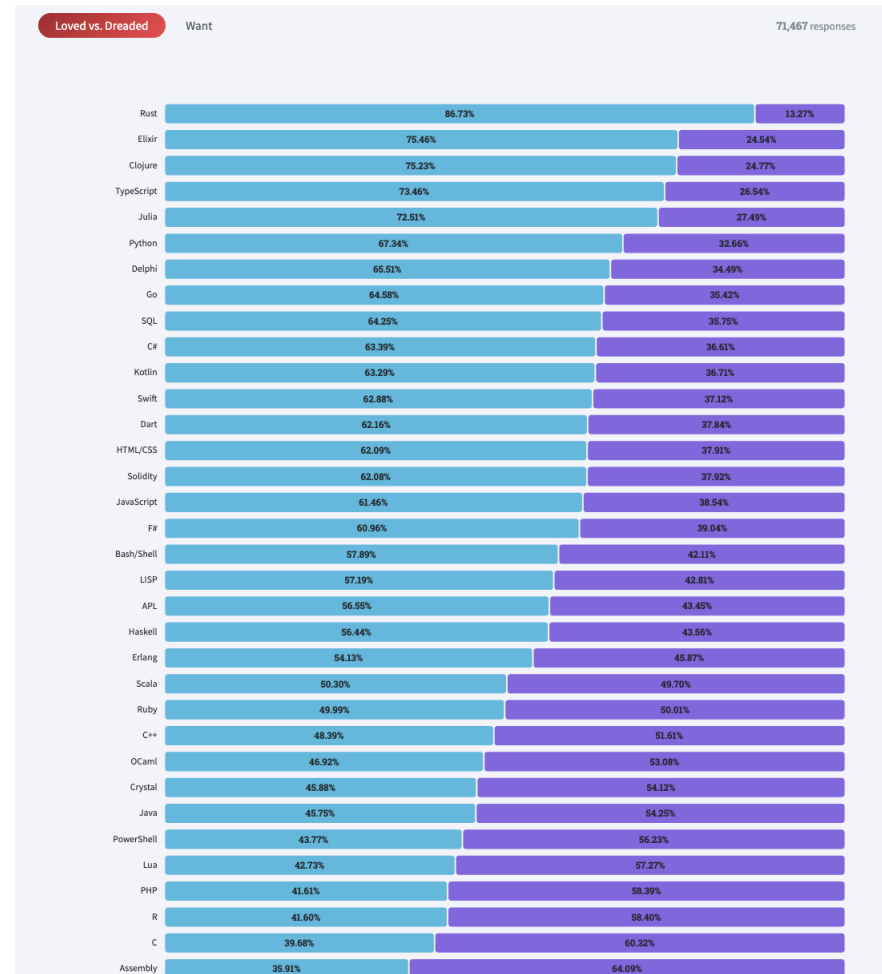
Downloads in past 2 Years ▾



What is TypeScript?

- Superset of EcmaScript 2022 or newer
- Compiles to EcmaScript 2022 or older (e.g. 2020, or 2015, or even 5 → IE11)
- Introduces static typing (like known from C++, Java and others)
- Advantages
 - **Awesome Code Completion (IDE)**
 - Easier Refactoring
 - **Static Code Analyzing (Compiler)**
 - Less Bugs!

Devs <3 TypeScript (StackOverflow Survey'22)



Separation of Web Technologies



<https://sametcelikbicak.com/html-css-typescript>



Google

TypeScript

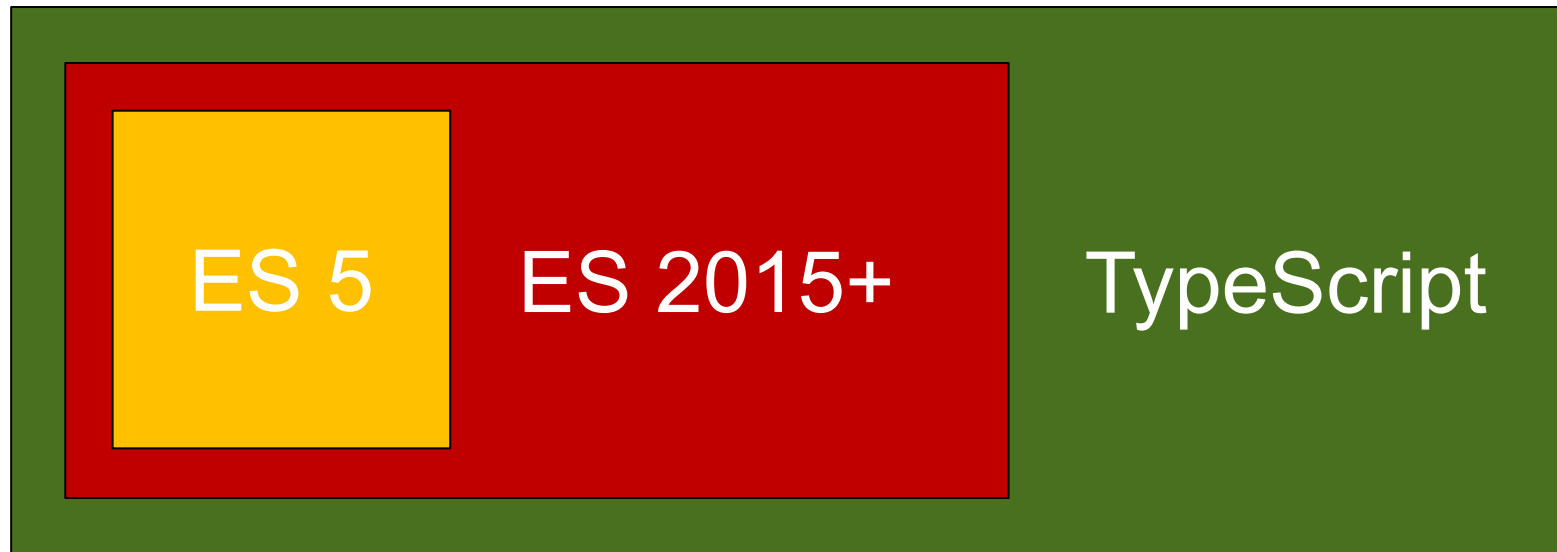
Separation

Community

angular- connect, London



JavaScript vs TypeScript

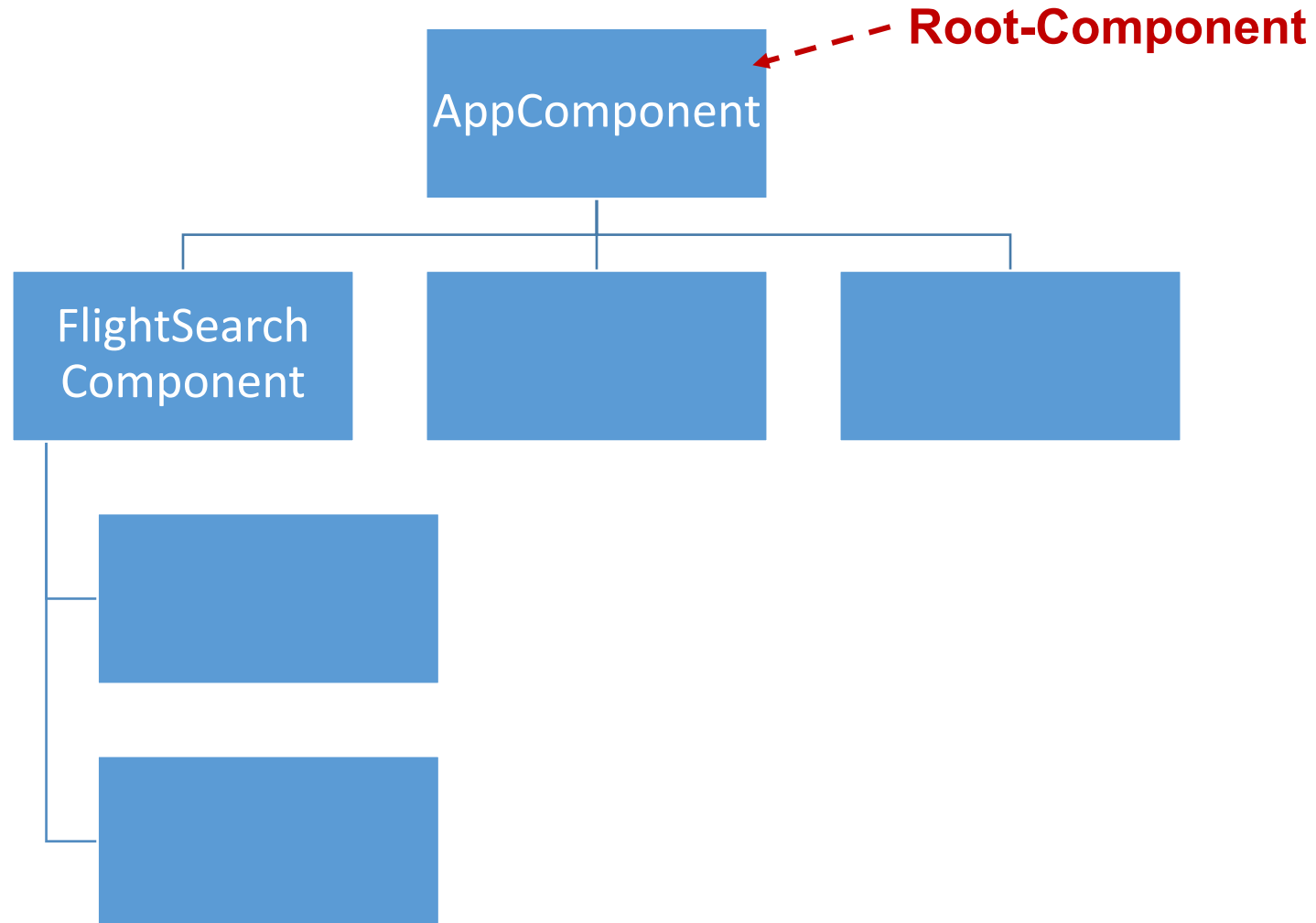


←
compilation



First steps with Angular

App == component tree



Angular Dev Tools

- <https://chrome.google.com/webstore/detail/angular-devtools/ienfalfjdbdpebioblackkekamfmbnh>

DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

AppComponent

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



AppComponent

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

```
@Component({  
  selector: 'app-flight-app',  
  templateUrl: './app.component.html'  
})
```

```
export class AppComponent {  
  title = 'Hello World!';  
}
```

Library

E.g.: @angular/core

Own project

E.g.: ../entities/flight
No ending ".ts"



AppComponent

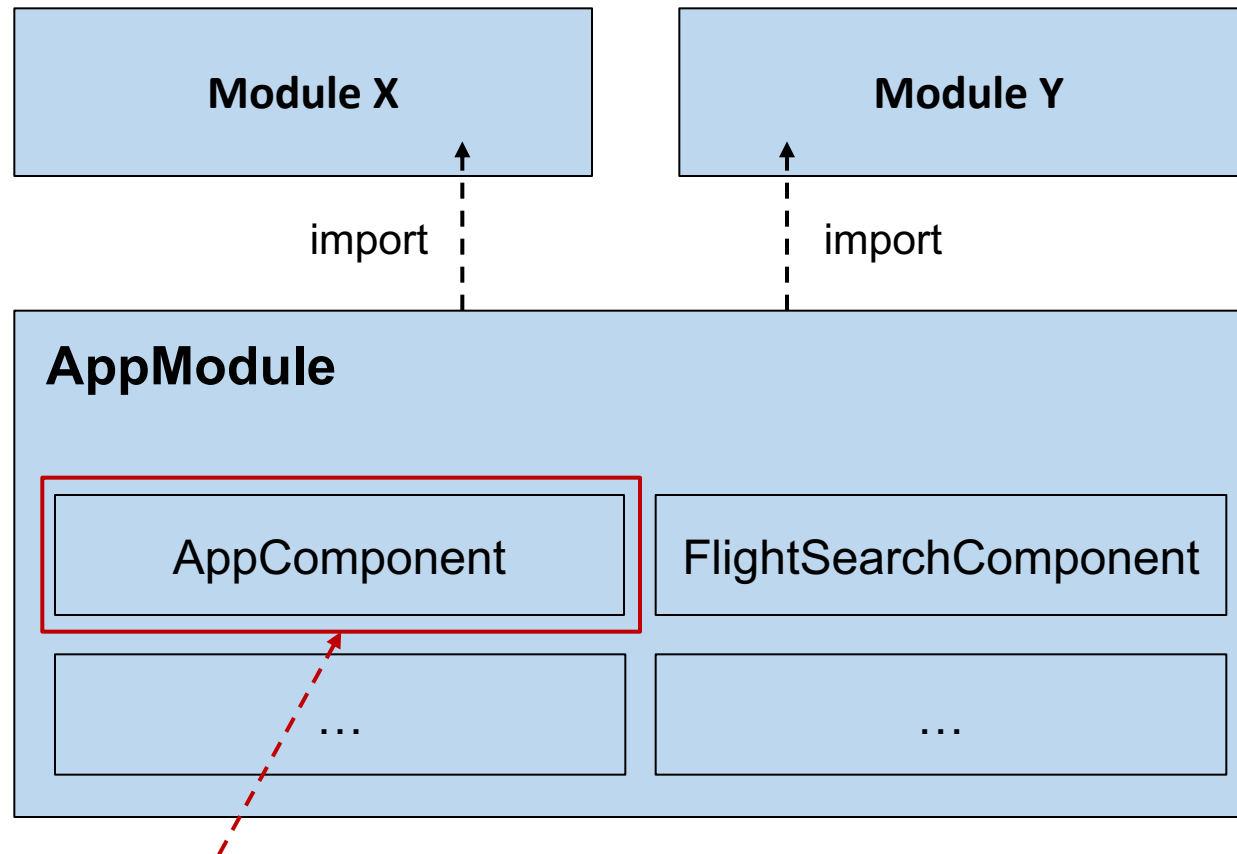
```
import { Component } from '@angular/core';

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

```
<h1>{{title}}</h1>
<div class="container">
  <app-flight-search />
</div>
```



Module



Root-Component

AppModule

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



Bootstrapping

- Starting Angular
- Loading
 - RootModule/AppModule with
 - RootComponent/AppComponent

Bootstrapping

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



index.html

```
[...]  
<body>  
  <app-flight-app></app-flight-app>  
  <script src="..."></script>  
</body>  
[...]
```



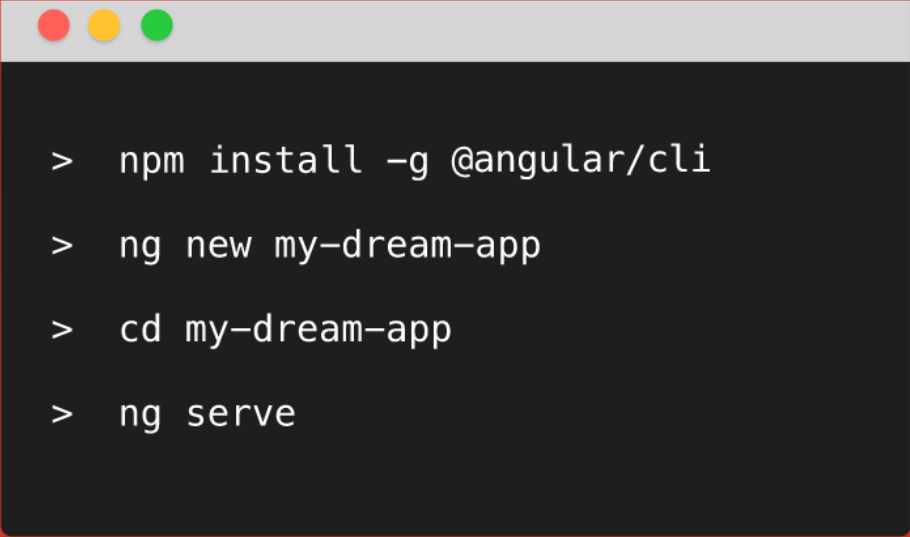
A new Angular project



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

Our Starterkit

- `npm i -g @angular/cli`
- `ng new essentials && cd essentials`
- `npm i bootstrap --save`
- Adding global styles in *angular.json*

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

DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT



Your first component

Component as TypeScript class

```
@Component({  
  selector: 'app-flight-search',  
  templateUrl: './flight-search.html'  
})  
export class FlightSearchComponent {  
  from = '';  
  to = '';  
  flights: Flight[] = [];  
  
  search(): void { [...] }  
  select(flight: Flight): void { [...] }  
}
```



Template

Two Way Binding

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

Event (/Output) Binding

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

Property (/Input) Binding

```
<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>
```

?



Falsy values

- false
- null
- undefined
- "" // empty string
- 0 // number
- NaN



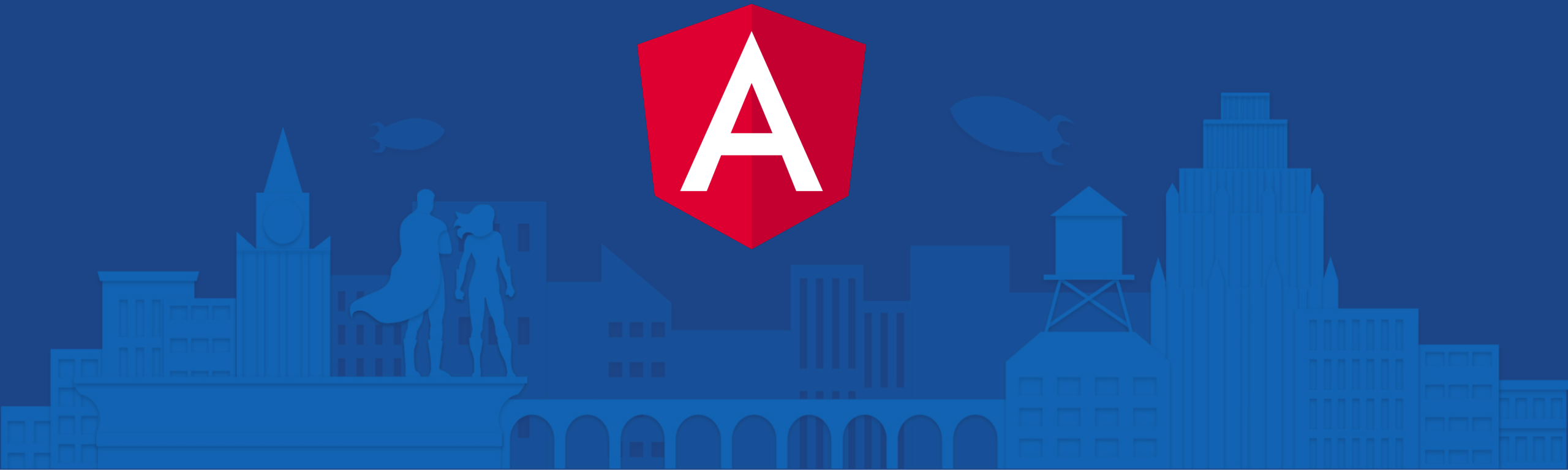
DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT



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)`
- ...



Inject HttpClient

```
@Component({  
  selector: 'app-flight-search',  
  templateUrl: './flight-search.html'  
})  
export class FlightSearchComponent {  
  from = '';  
  to = '';  
  flights: Flight[] = [];  
  
  constructor(private http: HttpClient) { [...] }  
  
  search(): void { [...] }  
  select(flight: Flight): void { [...] }  
}
```



Inject HttpClient - V14

```
@Component({  
  selector: 'app-flight-search',  
  templateUrl: './flight-search.html'  
})  
export class FlightSearchComponent {  
  from = '';  
  to = '';  
  flights: Flight[] = [];  
  
  private http = inject(HttpClient);  
  
  search(): void { [...] }  
  select(flight: Flight): void { [...] }  
}
```



Use HttpClient (I)

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



Use HttpClient (II)

```
const url = 'http://www.angular.at/api/flight';  
  
const params = new HttpParams().set('from', this.from).set('to', this.to);
```



Use HttpClient (III)

```
const url = 'http://www.angular.at/api/flight';  
  
const params = new HttpParams().set('from', this.from).set('to', this.to);  
  
const headers = new HttpHeaders().set('Accept', 'application/json');
```



Use HttpClient (IV)

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

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

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

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



Use HttpClient (V)

```
const url = 'http://www.angular.at/api/flight';  
  
const params = new HttpParams().set('from', this.from).set('to', this.to);  
  
const headers = new HttpHeaders().set('Accept', 'application/json');  
  
this.http.get<Flight[]>(url, { params, headers })
```

Short hand



Use HttpClient (VI)

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

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

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

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



Use HttpClient (VII)

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

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

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

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



Use HttpClient (VIII)

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

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

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

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



Use HttpClient (IX)

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

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

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

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



DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT

Use HttpClient (X)

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

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

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

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

←----- **Observable**



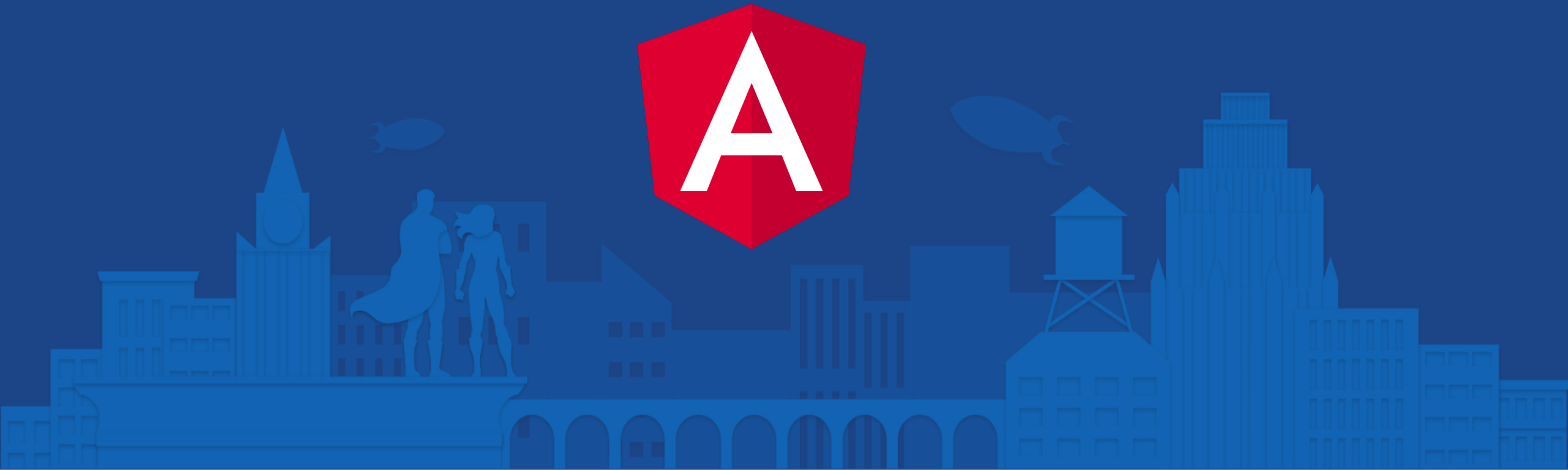
LAB



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT



Use Angular Directives

What are Directives?

- Add behaviour to html elements
- Are used as html attributes
- Examples:
 - `<input [(ngModel)]="from">`
 - `<div *ngFor="let flight of flights">...</div>`

Types of Directives?

- Structural Directives
 - *ngIf="statement"
 - *ngFor="let element of array"
 - *ngSwitch="something"
- Attribute directives
 - Built-ins
 - [(ngModel)]
 - [ngClass] or [ngStyle]
 - Custom ones

Examples (I)

```
<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>
```



Examples (II)

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



DEMO



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE



SOFTWARE
ARCHITECT