

Lab 3: Databinding & eventbinding

2 way binding

Binnen Angular kan je gebruik maken van 1-way en 2-way binding functies. Als je een attribuut tussen gewone haakjes plaatst, wordt er data verzonden van de view naar de component. Plaats je een attribuut tussen vierkante haken, dan wordt er data van de component naar de view verstuurd. Gebruik je beide, dan krijg je 2-way binding:

```
[(ngModel)]="variableName"
```

Pas de **contact.component.html** aan en voorzie volgende inhoud:

```
<div>
  <h2>{{ name }}</h2>
  <ul>
    <li>Email: {{ email }}</li>
    <li>Phone: {{ phone }}</li>
  </ul>
</div>

<input type="text" name="name"
  placeholder="Name" [(ngModel)]="name">
<input type="email" name="email"
  placeholder="name@domain.com" [(ngModel)]="email">
<input type="tel" name="phone"
  placeholder="Phone number" [(ngModel)]="phone">
<input type="checkbox" name="isFavorite" id="isFavorite">
<label for="isFavorite">Favorite</label><br>
<button type="submit">Submit</button>
```

Vergeet niet om de Formsmodule te importeren in de app.module.ts file! De checkbox en button doen voorlopig nog niets, maar we kunnen wel al testen met de data binding van de input velden. Als je de tekst aanpast, zie je de list items wijzigen van content.

Vervolgens voorzien we een nieuwe property bij onze component klasse:

```
isFavorite: boolean = false;
```

Koppel de checkbox aan de property van de component door middel van 2-way binding:

```
<input type="checkbox" name="isFavorite" id="isFavorite"  
[(ngModel)]="isFavorite">
```

Event binding

Maak in je component onderstaande methode. Deze methode toont de status van de isFavorite boolean:

```
onClick(): void{  
    console.log('Button clicked. Status van favorite is:' +  
    this.isFavorite);  
}
```

Pas de HTML van de component aan zodat de event gekoppeld wordt aan de button:

```
<button type="button" (click)="onClick()">Submit</button>
```

Test het resultaat. Je ziet in de developer console de log boodschap verschijnen.

@Input binding

Momenteel blijven we binnen éénzelfde component. Vaak is het zo, dat data vanuit een bovenliggende component doorgestuurd wordt. Daarnaast is het ook vaak nodig dat een child component data terug stuurt naar zijn parent.

De input binding kunnen we gebruiken om lijsten van components aan te maken. Om hiervan gebruik te maken, moet je Input importeren uit @angular/core:

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

We gaan er voor zorgen dat onze contact data input gebonden is. Importeer de klasse uit hoofdstuk 2 in je component:

```
import { Contact } from '../models/contact.model';
```

Aan de klasse van de component voegen we nu een property toe van het type Contact:

```
export class ContactComponent {  
    @Input() contact: Contact;  
}
```

We gaan er voor zorgen dat de data in de ContactComponent binnenkomt via de parent (AppComponent). Ga naar **app.component.ts** en voorzie opnieuw de Contact klasse import. Daarnaast voorzie je een property myContact die aangemaakt wordt in de ngOnInit methode:

app.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Contact } from './models/contact.model';

...

export class AppComponent{
  myContact: Contact;

  ngOnInit(): void {
    this.myContact = new Contact(
      'John Doe',
      'john.doe@gmail.com',
      '01166424893',
      true,
      'assets/avatar.png'
    );
  }
}
```

Tenslotte kunnen we de 2 html files aanpassen:

app.component.html:

```
<app-contact [contact]="myContact"></app-contact>
```

contact.component.html:

```
<div>
  <h2>{{ contact.name }}</h2>
  <ul>
    <li>Email: {{ contact.email }}</li>
    <li>Phone: {{ contact.phone }}</li>
  </ul>
</div>

<input type="text" name="name"
  placeholder="Name" [(ngModel)]="contact.name">
<input type="email" name="email"
  placeholder="name@domain.com" [(ngModel)]="contact.email">
<input type="tel" name="phone"
  placeholder="Phone number" [(ngModel)]="contact.phone">
<input type="checkbox" name="isFavorite" id="isFavorite"
[(ngModel)]="contact.isFavorite">
<label for="isFavorite">Favorite</label><br>
<button type="button" (click)="onClick()">Submit</button>
```

@Output binding

Naast data ontvangen, kan een component ook data verzenden naar zijn parent. Om iets terug te sturen naar de parent werken met de @Output() binding en een EventEmitter.

Voorzie beide imports in de **contact.component.ts**:

```
import { Component, OnInit, Input, Output, EventEmitter } from
'@angular/core';
```

Voorzie een extra property in de component klasse:

```
@Output() onSubmit: EventEmitter<Contact> = new EventEmitter();
```

Vervolgens voorzien we een methode die uitgevoerd wordt bij het drukken op de submit knop. Hierin laten we de EventEmitter data terug naar de parent sturen:

contact.component.html:

```
<button type="button" (click)="submit()">Submit</button>
```

contact.component.ts:

```
submit(){  
  this.onSubmit.emit(this.contact);  
}
```

Het contact object wordt nu verzonden naar de parent, maar deze wordt nog niet opgevangen in de parent zelf. Pas de **app.component.html** file aan als volgt:

```
<app-contact [contact]="myContact"
(onSubmit)="handleData($event)"></app-contact>
```

(onSubmit) verwijst naar de EventEmitter in de child component. De handleData methode maken we later aan in de app.component.ts file. Het \$event object bevat de data die de emitter doorstuurt. Bij een EventEmitter is dit altijd \$event.

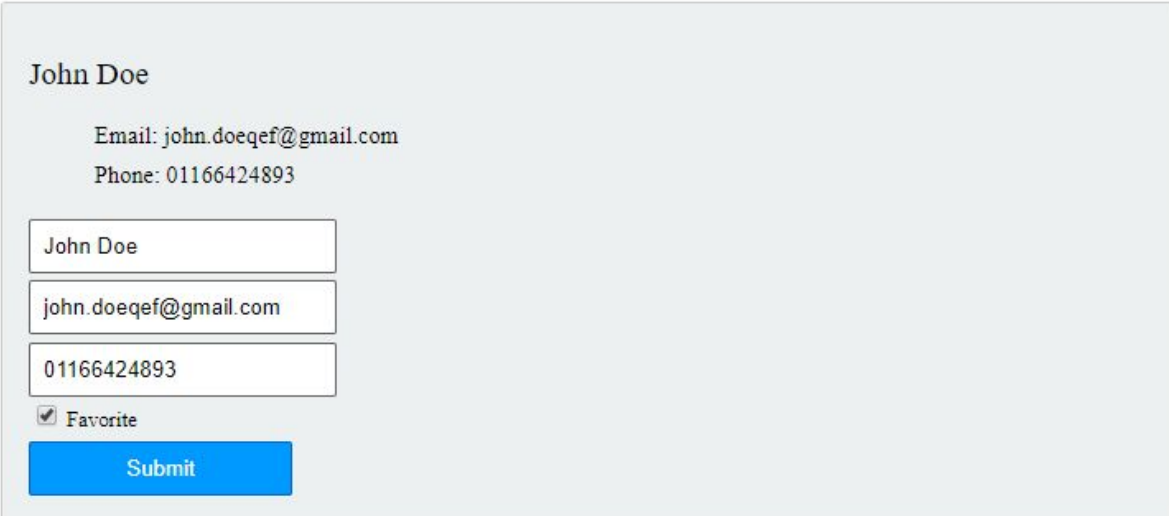
Voorzie de handleData methode in de **app.component.ts** file:

```
handleData(event: Contact){
  console.log('Recieved data!', event);
}
```

De methode krijgt als parameter 'event' van het datatype Contact. Dit datatype werd bepaald bij het aanmaken van de EventEmitter: onSubmit: EventEmitter<Contact>.

Het eindresultaat zou er als volgt moeten uitzien:

Contacts



The screenshot shows a contact form with the following elements:

- Name: John Doe
- Email: john.doeqef@gmail.com
- Phone: 01166424893
- Input fields for Name, Email, and Phone, each containing the same text as the labels above them.
- A checkbox labeled "Favorite" which is checked.
- A blue "Submit" button.

Bij het klikken op de submit knop, krijg je de melding "recieved data: ..." in je console.

Bij het aanpassen van de tekstvelden, wordt de data in de view ook aangepast.