Front End Development 1
Sesi 24

Angular Form

2 Macam Jenis Angular Form

Form sangat berguna untuk mengirim/ mengupdate data ke Database maupun ke API. Dalam Angular, terdapat 2 Jenis Form yang patut diketahui:

- **Template-driven forms**: Hampir semua logic akan diproses di file .component.html (Component template)
- **Reactive forms**: Hampir semua fungsionalitas dan logic akan dilakukan di file .component.ts (Component class).

Referensi: https://angular.io/guide/forms-overview

Perbedaan Reactive & Template Driven Form

	REACTIVE	TEMPLATE-DRIVEN
Setup of form model	Explicit, created in component class	Implicit, created by directives
Data model	Structured and immutable	Unstructured and mutable
Data flow	Synchronous	Asynchronous
Form validation	Functions	Directives

Referensi: https://angular.io/guide/forms-overview

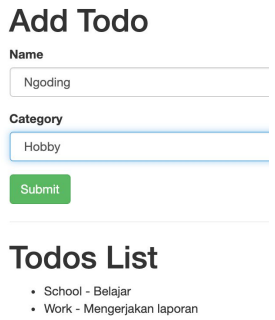
Template-driven form

Template-driven Form membutuhkan directive-directive yang ada di dalam **FormsModule**.
Berikut directive yang digunakan template driven form:

- **NgModel:** Menghandle perubahan value dalam elemen form seperti input. JUga menghandle validasi input dan error handling.
- **NgForm**: mem-bind <form> element, serta mentrack form value dan status validasi. Dapat digunakan dan secara default aktif di semua tag form setelah meng-import **FormsModule**.
- **NgModelGroup** mem-bind instance FormGroup ke DOM element.

Referensi: https://angular.io/guide/forms

Template-driven form: Contoh Todo List



Membuat Form Menambahkan Task.

1. Import FormsModule di Module.
2. Buat 1 class bernama “Task”

app.module.ts M

```
src > app > app.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3
4  import { AppComponent } from './app.component';
5  import { FormsModule } from '@angular/forms';
6  import { TemplateDrivenFormComponent } from './template-driven-form/template-driven-form.component';
7
8  @NgModule({
9    declarations: [
10     AppComponent,
11     TemplateDrivenFormComponent
12   ],
13   imports: [
14     BrowserModule,
15     FormsModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
20 export class AppModule { }
```

task.ts U

```
src > app > task.ts > ...
1  export class Task {
2    static nextId = 1;
3
4    constructor(
5      public task: string,
6      public isCompleted: boolean,
7      public category: string,
8      public id: number = 0,
9    ) {
10     this.id = id ? id : Task.nextId++;
11   }
12
13 }
```

Referensi: https://angular.io/guide/forms

Template-driven form: Menambah NgForm dan NgModel

3. Integrasikan ngModel dengan value di dalam component. Properti **name** HARUS ada dalam template input.
4. Buat 1 method yang akan menghandle form submit.

```ts
import { Component } from '@angular/core';
import { Task } from '../task';
import { NgForm } from '@angular/forms';

@Component({…
})

export class TemplateDrivenFormComponent {
  tasks: Task[] = []
  categories = ['School', 'Work', 'Hobby']

  onSubmit(form: NgForm) {
    const {taskName, category} = form.value
    this.tasks = [...this.tasks, new Task(taskName, false, category)]
    form.reset()
  }
}
```

```html
<form (ngSubmit)="onSubmit(todoForm)" #todoForm="ngForm">
  <div class="form-group">
    <label for="name">Task Name</label>
    <input type="text" class="form-control" id="taskName"
           required name="taskName" ngModel>
  </div>

  <div class="form-group">
    <label for="category">Category</label>
    <select class="form-control" id="category" name="category" ngModel>
      <option *ngFor="let category of categories" [value]="category">{{category}}</option>
    </select>
  </div>

  <button type="submit" class="btn btn-success">Submit</button>
</form>
```

Referensi: https://angular.io/guide/forms

Template-driven form: Penjelasan Form template

```
<form (ngSubmit)="onSubmit(todoForm)" #todoForm="ngForm">
  <div class="form-group">
    <label for="name">Task Name</label>
    <input type="text" class="form-control" id="taskName"
           required name="taskName" ngModel>
  </div>

  <div class="form-group">
    <label for="category">Category</label>
    <select class="form-control" id="category" name="category" ngModel>
      <option *ngFor="let category of categories" [value]="category">{{category}}</option>
    </select>
  </div>

  <button type="submit" class="btn btn-success">Submit</button>
</form>
```

- Pada setiap elemen **input**, atribut **name** dan **ngModel** wajib dicantumkan agar dapat dideteksi sebagai form value.
- Pada tag **Form,** variable template **todoForm** berisi **ngForm**, agar dapat menggunakan method atau atribut dalam ngForm.
- Pada tag form, terdapat **ngSubmit**, yang mengarah pada method onSubmit di dalam component.ts. Di dalamnya dikirim 1 params variabel template **todoForm**.

Referensi: https://angular.io/guide/forms

Template-driven form: Penjelasan NgForm

```
import { Component } from '@angular/core';
import { Task } from '../task';
import { NgForm } from '@angular/forms';

@Component({…
})

export class TemplateDrivenFormComponent {
  tasks: Task[] = []
  categories = ['School', 'Work', 'Hobby']

  onSubmit(form: NgForm) {
    const {taskName, category} = form.value
    this.tasks = [...this.tasks, new Task(taskName, false, category)]
    form.reset()
  }
}
```

Ketika form di submit, maka diarahkan ke method onSubmit yang telah dibuat:

- onSubmit method menerima 1 params, yaitu form yang bertipe data NgForm, dari component template. (slide sebelumnya.)
- Maka, di dalam komponen, kita pun bisa memakai properti-properti dalam NgForm.
- Disini, properti yang digunakan adalah **.value** untuk mengambil value-value input dalam form.
- Serta method **.reset()** untuk clear element input dalam form pasca submit.

Referensi: https://angular.io/api/forms/NgForm#description