

# Praktikum #3

## Ionic Components & Page

---

### Hybrid Web Development

Dosen Pengampu: Aditya Rizki Yudiantika, M.Eng.

### Mengenal Typescript

Javascript (JS) merupakan salah satu bahasa utama di dunia web programming di sisi client. JS memiliki keterbatasan dalam membuat aplikasi yang kompleks dan berskala besar, sehingga membutuhkan Typescript.

Typescript diperkenalkan oleh Microsoft untuk memberi kemudahan dalam mengembangkan aplikasi yang kompleks menggunakan basis pemrograman JS. Typescript merupakan bahasa pemrograman berbasis JS yang bisa bekerja dengan konsep OOP (object oriented programming).

### Collecting Data

Pada contoh praktikum sebelumnya, kita sudah membuat form registrasi sebagai latihan menggunakan layout dan beberapa komponen pada Ionic. Pada latihan kali ini kita akan mencoba untuk menangkap dan memproses data-data dari input pengguna pada form yang sudah dibuat sebelumnya.

Pada masing-masing input component kita tambahkan component angular [(ngModel)] yang berguna untuk menangkap dan mengirim objek pada suatu class. Berikut ini code di [home.page.html](#)

```
1. <ion-item>
2.   <ion-label position="floating">Nama Lengkap</ion-label>
3.   <ion-input [(ngModel)]="info.fullname" type="text"></ion-input>
4. </ion-item>
5. <ion-item>
6.   <ion-label position="floating">Email</ion-label>
7.   <ion-input [(ngModel)]="info.email" type="email"></ion-input>
8. </ion-item>
9. <ion-item>
10.  <ion-label position="floating" type="email">Tanggal Lahir</ion-label>
11.  <ion-datetime value="2019-10-01T15:43:40.394Z" [(ngModel)]="info.dob" display-
    timezone="utc"></ion-datetime>
12. </ion-item>
```

Ketiga components tersebut ditambahkan atribut sebagai berikut:

- [(ngModel)]="info.fullname"
- [(ngModel)]="info.email"
- [(ngModel)]="info.dob"

Selanjutnya kita perlu mendefinisikan variable info pada file [home.page.ts](#). Tambahkan code berikut ini di dalam class Homepage{ } pada baris awal:

```
info: any = {};
```

Untuk menampilkan data tersebut, kita memerlukan sebuah event yang ditambahkan pada button Submit. Kembali tambahkan atribut berikut ini pada elemen ion-button:

```
<ion-button color="success" (click)="saveData()" expand="block">Submit</ion-button>
```

Pada elemen di atas, event yang kita gunakan adalah (click) yang mana event tersebut akan menjalankan function saveData(). Untuk menambahkan function ini kita perlu edit lagi file [home.page.ts](#) kemudian tambahkan code berikut ini pada class Homepage{ }

```
1. saveData() {
2.   console.log(this.info);
3. }
```

Pada gambar di bawah ini adalah hasil ketika kita isikan data pada form kemudian kita tekan button submit, hasilnya terlihat pada console di developer mode browser.

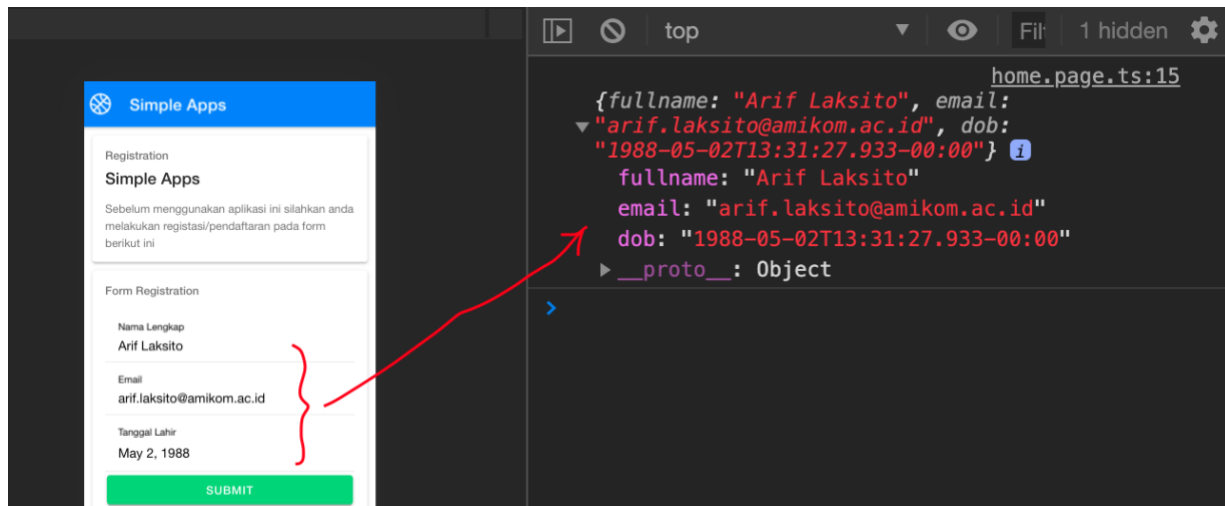


Figure 1 Tampilan developer mode di browser untuk collect data dari form

## Membuat page/halaman baru

Dengan adanya ionic CLI yang sudah terinstall di perangkat kita(laptop/pc), kita dapat dengan mudah menggunakan ionic command untuk membuat page. Langsung saja untuk membuat page baru, kita buka kembali cmd/terminal, turn off service ionic dengan cara tekan ctrl + c, kemudian kita bisa menuliskan perintah untuk membuat page berikut

```
C:\> ionic generate page score
```

Ionic generate adalah perintah untuk membuat suatu component baru bisa berupa *page*, *service*, maupun *directives*. Disini type kita pilih **page**, dan **score** merupakan nama page baru yang ingin kita buat. Tunggu beberapa saat dan sekarang terdapat folder baru yaitu src/app/score.

Edit file **score.page.html** dengan menuliskan kode berikut:

```
1. <ion-header>
2.   <ion-toolbar color="primary">
3.     <ion-buttons slot="start">
4.       <ion-icon size="large" name="basketball-outline">
5.     </ion-icon>
6.   </ion-buttons>
7.   <ion-title>Score Apps</ion-title>
8. </ion-toolbar>
9. </ion-header>
10.
11. <ion-content>
```

```
12.     <ion-card>
13.
14.     </ion-card>
15. </ion-content>
```

Selanjutnya untuk menghubungkan dari /home ke /score kita bisa menambahkan 1 button di `home.page.html` dengan kode berikut:

```
<ion-button color="warning" (click)="scorePage()" expand="block">Score Apps</ion-button>
```

Seperti pada code sebelumnya, kita perlu menambahkan function `scorePage()` pada file `home.page.ts`, terdapat 3 bagian yang perlu ditambahkan yaitu:

1. Bagian pertama tambahkan import untuk library router


```
import {Router} from '@angular/router';
```

2. Edit class constructor, tambahkan parameter berikut:

```
constructor(private route: Router) {}
```

3. Buat fungsi `scorePage()` pada class `HomePage{ }` untuk mengarahkan ke page /score

```
scorePage() {
    this.route.navigate(['/score']);
}
```

 Simple Apps

Registration

**Simple Apps**

Sebelum menggunakan aplikasi ini silahkan anda melakukan registrasi/pendaftaran pada form berikut ini

Form Registration

Nama Lengkap

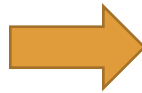
Email


Tanggal Lahir

SUBMIT

SCORE APPS

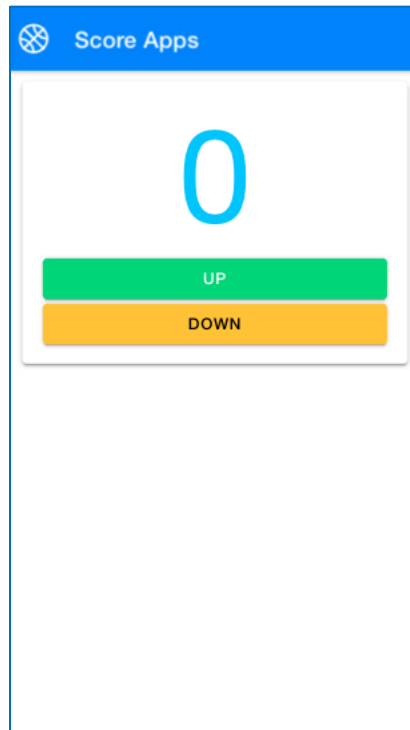
Footer



 Score Apps

## Membuat papan score

Di page /score tersebut kita akan membuat papan score. Yang perlu kita lakukan pertama kali adalah membuat layout untuk tampilan tersebut menjadi seperti pada gambar dibawah ini:



Jika kita ingin menambahkan style pada ionic, misalkan pada contoh di atas ingin mengubah ukuran font angka 0 menjadi 116px, kita dapat menambahkan pada file .scss

Buka file score.page.scss dan tambahkan kode dibawah ini:

```
.ion-2x { font-size: 116px !important;}
```

Berikutnya tambahkan code dibawah ini pada file **score.page.html**:

```
1. <ion-card>
2.   <ion-card-content>
3.     <ion-text color="secondary" class="ion-text-center">
4.       <h1 class="ion-2x">{{no}}</h1>
5.       <ion-button color="success" (click)="up()" expand="block">Up</ion-button>
6.       <ion-button color="warning" (click)="down()" expand="block">Down</ion-
   button>
7.     </ion-text>
8.   </ion-card-content>
9. </ion-card>
```

Perhatikan pada baris kode-4 di atas, untuk menggunakan style yang telah kita buat, tambahkan atribut class dengan value nama class yang telah kita definisikan pada file .scss, disini kita gunakan class="ion-2x".

Masih di baris ke-4, terdapat double bracket dengan keyword `no` `{{no}}` digunakan untuk membinding variable pada view. Variable yang digunakan disini adalah `no`, lebih jauh lagi akan kita tambahkan pada file `typescript`.

Pada kedua button kita tambahkan event (click) dengan fungsi `up()` untuk menaikkan angka dan fungsi `down()` untuk menurunkan angka. Pada fungsi `down()` kita tambahkan validasi jika angka kurang dari 1 atau  $= 0$  maka penurunan nilai tidak dijalankan.

## Latihan

Silahkan anda tambahkan kode pada file `score.page.ts` supaya papan score tersebut bisa bertambah nilainya jika ditekan tombol UP dan nilainya berkurang jika ditekan tombol DOWN

Tips:

1. Buat variable baru dengan nama `no`, set nilai awal 0
2. Tambahkan 2 function `up()` dan `down()`
3. Pada fungsi `up()` buat increment untuk menaikkan 1 angka
4. Pada fungsi `down()` buat decrement untuk menurunkan 1 angka
5. Tambahkan validasi di fungsi `down()` jika nilai `no > 0` akan dijalankan

Selamat mencoba!