# 183. Module Introduction

Vì chúng ta đang tạo Single Page Application (SPA) nên sẽ không submit Form tới Server. Thay vào đó chúng ta cần sử lý Form thông qua Angular.

Angular giúp lấy data từ các trường input mà người dung đã nhập và validate các data này. Angular cũng cho phép chúng ta thay đổi cách hiển thị của một Form (Tạo các red borders cho các trường invalid…). Tóm lại Angular cung cấp các tool mạnh để làm việc với Form.

# 184. Why do we Need Angular’s Help?

Tiếp cận gần hơn để thấy rõ: Angular làm việc với Form như thế nào?

Như đã nhắc ở phần trước. Angular cho phép ta lấy các data mà người dung nhập vào các trường input, và check valid của các trường này…

Angular lấy các data mà người dung nhập vào Form và biến đổi chúng thành java script Obj đại diện cho Form đó. Ví dụ về Obj đại diện ở bên dưới

{

Value: {

Name: ‘Nhat’,

Email: ‘123@gmail.com’

}

Valid: true

}

Từ Obj đại diện chúng ta có thể dễ dàng thao tác với Form và nắm bắt được trạng thái của nó

# 185. Template-Driven (TD) vs Reactive Approach

Một điều vô cùng quan trọng cần phải hiểu được rằng, Angular đề xuất 2 hướng tiếp cận khác nhau để sử lý Form:

* **Template-Driven:** Angular infers the Form Objet form the DOM – Đơn giản là chúng ta sẽ set up Form bằng HTML Code và angular sẽ tự động hiểu được cấu trúc của Form. Nó sẽ biết được trường nào có trong Form.
* **Reactive:** Form is created programmatically ad synchoronized with the DOM – Cấu trúc của Form sẽ được định nghĩa bằng TypeScript code. Mặt dù cách này khá phức tạp nhưng sau cùng nó sẽ cung cấp cho chúng ta những công cụ kiểm soát Form vô cùng mạnh – bạn có thể tùy chính từng phần nhỏ của Form tùy thích.

# 186. An Example Form (Có code demo)

Vì Form trong Angular App không submit thẳng data tới Server nên ta có thể thấy trong thẻ HTML <form> sẽ không có attr “action” cũng như METHOD.

Thay vì Submit thẳng angular sẽ sử lý trước dữ liệu ở Form (như đã đề cập). Chính vì lý do trên nên khi ấn vào nút submit (trong code demo) sẽ không có bất kỳ hành động nào xảy ra trên trang web (do form không gữi đe request).

# 187. TD: Creating the Form and Registering the Controls

Ở phần này chúng ta sẽ cùng xem cách mà Angular hiểu và chuyển dữ liệu trong Form thành Obj đại diện cho Form như thế nào (Theo hướng tiếp cận Template Driven).

Để làm được điều này hãy đảm bảo rằng: Tại App Module, FormsModule đã được import. Module này cung cấp rất nhiều chức năng liên quan tới Form. Nó là bắt buộc để xử lý Form theo hướng tiếp cận Driven Template.

Cách thức: Khi angular phát hiện ra có cấu trúc form trong template của Component (thông qua thẻ Form đóng vai trò như selector) và nó sẽ tạo ra Object đại diện.

Tuy nhiên Angular vẫn chưa hiểu được các trường input trong Form. Để Angular hiểu đâu là các trường input chúng ta cần phải đăng ký các trường này với angular.

Cách thức đăng ký các input controller cho Form:

* Thêm “ngModel” vào các trường input.
* Thêm attr “name” và giá trị của name vào trường input.

# 188. TD: Submitting and Using the Form

Ở phần này chúng ta sẽ cùng làm cho Form có thể “Submit” được – hay nói cách khác là có thể thu thập được dữ liệu mà người dung nhập và trạng thái của Form thông qua event Submit.

Để có thể Form có thể Submit được, chúng ta sẽ bind một event vào thẻ Form (chứ không phải nút submit) bằng directive “ngSubmit”. Sự kiện này sẽ kích hoạt mỗi khi Form được Submit. Ví dụ minh họa <from (ngSubmit)=”onSubmit()”>. Tới đây thì form đã có khả năng Submit.

Tuy nhiên làm thể nào để lấy dữ liệu từ form và chuyển nó thành Obj đại diện cho form. Chúng ta sẽ dùng #Reference\_Tempate tại thẻ <Form> và gán cho #Local\_Ref này giá trị là “ngForm”, sau đó truyền #Local\_Ref này vào hàm sử lý Submit đã bind vào sự kiện (ngSubmit). Ví dụ minh họa <from (ngSubmit)=”onSubmit(f)” #f=”ngForm”>.

Để xem nội dung của Obj đại diện cho Form, trong hàm sử lý submit chúng ta có thể làm như sau:

onSubmit(f: ngForm) {

console.log(f)

}