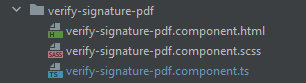
ANGULAR

I,Overview

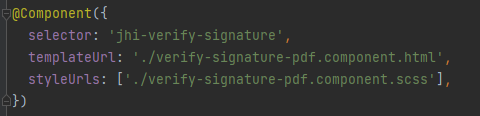
* Là một JavaScript Framework do gg phát triển để xây dựng các SPA(Single Page App) bằng JavaScript, HTML, TypeScript.
* Cung cấp các tính năng tích hợp cho animation, http service…
* Code đc viết bằng TypeScript , biên dịch thành JavaScript và hiển thị tương tự trong các trình duyệt

II,Component

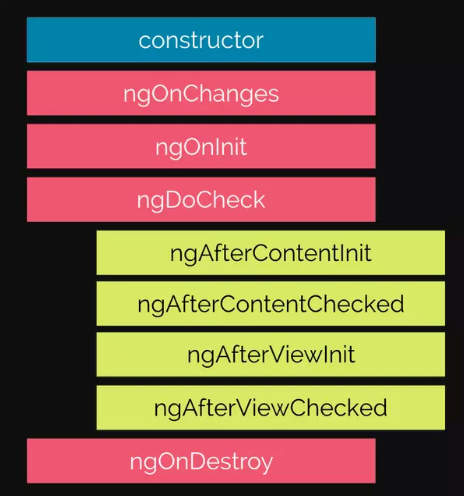
* Khối code trong app Angular, là sự kết hợp template HTML và nhúng kèm code TypeScript.
* Các Component độc lập với nhau và độc lập với hệ thống
* 1 component là 1 control trên màn hình hiển thị gồm giao diện HTML và code sử lý sự kiện kèm control đó
* Cấu trúc của 1 component :
  + HTML template : bao gồm phần hiển thị giao diện cho component
  + CSS selector : là file css format giao diện cho HTML template của component
  + 1 class typescript: được sử dụng để khai báo các thành phần của component và còn là file thực hiện chứa các biến dữ liệu của component.
  + Ngoài ra component còn có 1 số thành phần khác như:
    - Service : thực hiện xử lí logic cho component, thực thiện call api,…
    - Module : gọi component, service,.. thành 1 phần 1 nhóm để có thể thực hiện export ra ngoài.



* @Component :



* + Selector : name của component, dùng để gọi vào html
  + TemplateUrl : đường dẫn đến file html của component
  + styleUrls : đường dẫn đến file css của component
* Lifecycle trong html :



* + Constructer : Hàm khởi tạo của component
  + ngOnchanges : được gọi khi giá trị nào đó được binding vào component
  + ngOnInit : được gọi khi component được gọi lần đầu tiên
  + ngDoCheck : được gọi khi component có sự thay đổi
  + ngAfterContentInit : đc gọi khi component đc tạo thành công
  + ngAfterContentChecked : đc gọi khi angular kiểm tra xong nội dung của component
  + ngAfterViewInit : tương tự AfterContentInit nhưng được gọi khi component đó và các component con của nó được tạo thành công.
  + ngAfterViewChecked : tương tự ngAfterContentChecked nhưng được gọi khi component đó và các component con của nó được kiểm tra xong.
  + ngOnDestroy: được gọi đến khi component hoặc derective bị phá bởi angular.

III, Template

* Là file html trong component, sử dụng để hiển thị giao diện web
* Interpolation : cách thức truyền dữ liệu từ .ts -> html
* Binding : truyền dữ liệu từ .ts và html, có 2 cách :
  + One way binding : Thực hiện truyền dữ liệu 1 chiều, từ html->.ts hoặc ngược lại, khi dữ liệu thay đổi thì web refresh
  + Two way binding : Thực hiện truyền dữ liệu 2 chiều từ ts->html và ngược lại, giá trị ở html gắn với biến trong .ts, nên khi giá trị ở html thay đổi thì gía trị ở .ts cũng thay đổi.Web sẽ không refresh
* Pipe : là 1 tính năng xây dựng với mục tiêu biến đổi dữ liệu đầu ra , hiển thị trên template đúng với ý tưởng thiết kế lập trình . EX : hiển thị kiểu datetime , uppercase,LowerCase,…

IV, Directive

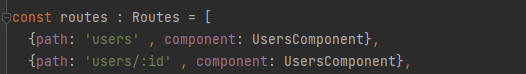
* Dùng để thao tác với các element của DOM
* Gồm 3 loại :
  + Component directive : component chính là directive, vì rõ ràng là component cho phép định nghĩa selector và gọi ra như một thẻ html tag (<component-name></component-name>)
  + Structure directive : Đúng như tên gọi, structural directive hay directive cấu trúc sẽ quyết định DOM element nào được thực thi. Các structural directive thường có dấu '\*' ở trước của directive. Ví dụ điển hình của structural directive chính là \*ngIf và \*ngFor
    - ngFor : dùng để lặp lại các phần HTML cho mỗi item. Nó giống như vòng lặp for. Mỗi lần lặp nó sẽ chạy lại các đoạn code trong vòng lặp
    - ngIf : chúng ta sử dụng ngIf để thêm và xóa các thành phần HTML dựa vào điều kiện có thoả mãn không. Điều kiện phải đúng mới thực hiện các câu lệnh bên trong.
    - ngSwitch : chúng ta dùng để quyết định thêm hoặc xóa các thành phần của web phụ thuộc vào điều kiện có thoả mãn hay không. Chúng ta thường dùng chung với ngSwitchCase và ngSwitchDefault
  + Attribute directive : Attribute Directive được dùng để thay đổi sự hiển thị hoặc hành vi của một thành phần trên web.
    - ngModel : được sử dụng cho việc binding 2 chiều như ta đã học trong bài databinding.
    - ngClass : được sử dụng để thêm hoặc xoá một class của một thành phần web.
    - ngStyle : dùng để thêm nhiều thuộc tính css cho một thành phần của web.

V,Dependence Injection

* Dependency Injection là một phần quan trọng trong bộ core của Angular. Sử dụng cơ chế Dependency Injection giúp chúng ta có thể nhúng service vào các component hoặc các service với nhau.
* Có 5 thành phần chính trong Angular Dependency Injection Framework :
  + Consumer : nơi nhúng service
  + Dependency : Những service được nhúng vào component
  + DI Token : Được sinh ra là một dãy ký tự tượng trưng cho ID và là duy nhất khi Service đăng ký là Dependency Injection với Framework
  + Provider : quản lý danh sách các dependencies và token của nó
  + Injector : quản lý việc nhúng các đối tượng Dependency vào các consumer
* Cơ chế hoạt động của dependence :
  + Dependency đăng ký với Provider
  + Sau đó Angular Provider sẽ nhúng các dependency vào các module consumer tương ứng.
  + Consumer sẽ khởi tạo các đối tượng Dependency thông qua constructor.

VI, Router

* Angular Router là module được tích hợp sâu vào Angular, giúp bạn dễ dàng tạo các route cho ứng dụng.
* Thực hiện nhiệm vụ chính là chuyển trang, thay đổi một số thành phần mà không cần phải tải lại trang.
* Sử dụng [RouterLink] để có thể thực hiện định dạng cho các thẻ html biết đường dẫn đi đến các component.
* Sử dụng <router-outlet></router-outlet> để khai báo vị trí mà component được chi định bởi đường dẫn chèn vào
* Thực hiện config path trước để có thể thực hiện chuyển đường dẫn đến đúng component và truyền dữ liệu



* Có thể sử dụng snapshot, paramMap, … để có thể lấy ra được dữ liệu khi thực hiện truyền qua url
* Mỗi path nên có 1 unique title đẻ có thể dễ dàng phân biệt trong lịch sử trình duyệt
* Lazy loading : là 1 tính năng có thể set cho angular chỉ load các module cần thiết.
* Route guards : là 1 tính năng thực hiện nang các user có thể truy cập vào app mà chưa có anthorization
* RouterModule.forRoot() : là 1 singleTon services đảm bảo rằng trong ứng dụng chỉ tồn tại 1 instance của routerModule.

VII, Form

* Có 2 cách để sử dụng form :
  + React Form : React Form xây dựng cấu trúc của 1 form trong component class. Chúng ta sử dụng Form Groups, Form Controls và From Array để xây dựng lên một form.

Các bước :

* + - import ReactiveFormsModule vào NgModule.



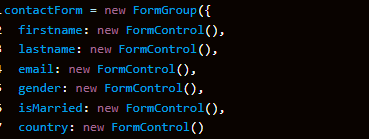
* + - Tạo Model cho Form chúng ta sử dụng FormGroup, FormControl và Validator.



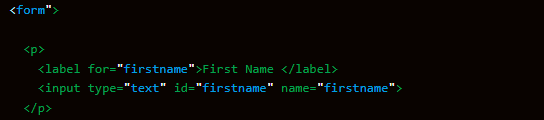
* + - tạo FormGroup. FormGroup được sử dụng để tạo các form control. Form control là các input, button mà ta thấy trên giao diện form.



* + - Tiếp đến tạo các form control



* + - tạo form bên template html :



* + - Binding template html vào model form bên component class.



* + - Binding các trường trong form vào FormControl models.



* + - Submit form, sử dụng ngSubmit.



* Template Form :
  + form nhỏ , đơn giản (ex: đăng nhập), bất đồng bộ
  + flow :

1. User truyền dữ liệu vào thẻ input trong form
2. Thẻ input gửi sự kiện chứa giá trị nhập vào đén controlValueAccessor.
3. controlValueAccessor sẽ thực hiện gán giá trị vào hàm setValue của formControl.
4. component đăng kí nhận giá trị sẽ vào valueChange để nhận giá trị
5. controlValueAccessor sẽ gọi đến ngModel.viewToModelUpdate để thực hiện cập nhật giá trị input và gửi đến component nhận

s