# CGDN-Angular-StackBlitz

# Luồng đi của ứng dụng

+ Step 1 : Bắt đầu từ file main.js để bootstrap module

+ Step 2 : Từ file main.js sẽ bootstrap file app.module.ts

+ Step 3 : Từ app.module.ts sẽ gọi app.component.ts

+ Step 4 : Trong app.component.ts ta khai báo app-root selector. Cái mà ta khai báo trong file index.html

là root của cả chương

## Các điểm chính của ứng dụng

### Component

- Component là chịu trách nhiệm trong việc hiển thị (UI). Component cho phép bạn sử dung lại ở nhiều modules khác nhau trong dự án.

- Component có 3 phần chính :

+ Component Class : quản lý dữ liệu , các methods , thao tác với dữ liệu và truyền dữ liệu cho html (view)

+ HTML template : hiển thị giao diện (button, layout, ect)

+ Styles : CSS cho component <br>

Trong mỗi component có 3 files : ví dụ như trong component top-bar . Ta có top-bar.component.html ; top-bar.component.ts; top-bar.component.css

### Vòng đời của 1 component

+ Vòng đời của một component xảy ra khi mình tạo , cập nhật , xoá các phần tử trên angular

+ Khi một component được tạo ra và phá huỷ đi. Angular sẽ tạo ra các vòng đời theo thứ tự sau. Dự vào các thứ tự này ta có thể hook vào và thay đổi

+ ngOnChanges(): Xảy ra khi Angular cập nhật giá trị hoặc binding giá trị cho phần tử. Nó gọi trước hàm ngOnInit();

+ ngOnInit() : Xảy ra khi mình khởi tạo các component và sau khi Angular hiện thị data binding cho các component. Được gọi 1 lần sau khi ngOnChanges được gọi

+ ngDoCheck() : Xảy ra khi có sự thay đổi được phát hiện trên các component.

+ ngAfterContentInit() :

### Angular Teamplate

- Cú pháp của Angular template được kế thừa từ HTML và Javascript . Trong phân này chúng ta sẽ giới thiệu cách sử dụng cú pháp trong module Product

- Để hiểu được template thì trong folder product-list. Mở file product-list.component.html

```

<h2>Products</h2>

<div \*ngFor="let product of products">

<h3>

{{ product.name }}

</h3>

</div>

```

+ Chúng ta có \*\*ngFor\*\* . dùng để repeat (lập lại) cac thẻ div. Chúng ta khai báo 1 biến product để duyệt các phần tử trong tập hợp products. Tập hợp products sẽ được truyền từ controller sang cho html. Nếu products có 10 phần tử thì chúng ta sẽ có 10 cái div.

+ Để hiển thị tên của từng sản phẩm thi mình dùng {{ product.name }} để hiển thị ra trang web

### Truyền giá trị vào component bằng @Input

```

Trong file src/app/product-list/product-alerts.component.ts

Ta thấy có anotation @Input product. Thì biến product sẽ nhận được giá trị từ file HTML truyền vào dưới đây.

export class ProductAlertsComponent implements OnInit {

@Input() product;

constructor() { }

ngOnInit() {

}

}

```

+ Ví dụ ta có component class có nhận đầu vào là product. Thì giá trị này sẽ được truyền vào bằng cách sau

```

<app-product-alerts

[product]="product">

</app-product-alerts>

```

+ Ta sử dụng thẻ [product] để truyền giá trị product vào alert-product component

### @Output sự kiện từ component con ra ngoài component cha

+ Ta mởi 4 files sau product-list.component.html, product-list.component.ts , product-alerts.component.html,product-alerts.component.ts . Trong đó product-list là component cha chứ component con la product-alert. Câu hỏi đặt ra làm sao mình truyền sự kiện từ component con lên component cha để xử lý. <br>

+ Đầu tiên ta mở file product-alerts.component.html

```

<p \*ngIf="product.price > 700">

<button (click)="notifyAA.emit()">Notify Me</button>

</p>

```

+ Chúng ta thấy khi người dùng click vào component con (click) thì chúng ta gọi method notifyAA trong component class (controller)

```

export class ProductAlertsComponent implements OnInit {

@Output() notifyAA = new EventEmitter();

}

```

+ Compoent Class Alert con sẽ Emit sự kiện lên cho component cha (product-list) bằng cách sử dụng @Output() và EventEmitter()

+ Như vậy Component class cha (controller) sẽ nhận được sự kiện bởi vì trong file html của component cha chúng ta khai báo nhận sự kiện từ con là (notifyAA) . Lúc này component sẽ gọi sự kiện onNotify trong controller

```<app-product-alerts [product]="product" (notifyAA)="onNotify()">

</app-product-alerts>

```

```

export class ProductListComponent {

products = products;

share() {

window.alert('The product has been shared!');

}

onNotify() {

window.alert('You will be notified when the product goes on sale');

}

}

```

## Routing cho các component trong ứng dụng

+ Để đăng ký rounting cho ứng dụng thì chúng ta khai báo trong file app.module.ts. Gồm 2 phần , trong đó phần 1 là path (đường URL. Phần 2 là component . Như ta thấy nếu người dùng gõ vào http://localhost:4200/cart thì nó sẽ hiển thị Component Cart

```

@NgModule({

imports: [

RouterModule.forRoot([

{ path: '', component: ProductListComponent},

{ path: 'products/:productId', component: ProductDetailsComponent },

{ path: 'cart', component : CartComponent},

{ path: 'user', component : UserComponent},

{ path: 'other', component : OthersProductComponent},

])

],

```

+ Để sử dụng được router thì trong file app.component.html ta sử dụng thẻ <router-outlet>

```

<div class="container">

<router-outlet></router-outlet>

</div>

```

## Databinding

### One way biding sử dụng {{ }}

+ Step 1 trong file user.component.ts trong component user. Ta có biến là name với giá trị

là angular

+ Step 2 để hiển thị được giá trị angular lên trên front end cho người dùng thấy thì ta

truyền vào biến nam trong 2 dấu ngoăc nhọn trong file user.component.html với cú

pháp sau {{ name }}

+ Như vậy chúng ta có thể kết luận để hiển thị giá trị từ ts qua html thì ta dùng {{ }}

### Truyền giá trị vào sử dụng như 1 attribute

+ Step 1

Trong file user.component.html. Ta có thể dùng attribute của thẻ để hiển thị giá trị

<input value="{{ name }}" />

+ Steo 2 chúng ta có thể Sử dụng atrribute để biding dư liệu cách thứ 2. Cái này được khuyên nên

<input [value]="name" />

### Truyền sự kiện

+ Step 1 . Trong file html ta khai báo nút button có sự kiện onclick <button (click)="onButtonClicked()">Change Name</button>. Như ta thấy sự kiện được bọc trong 2 dấu ngoại tròn () .

```

<button (click)="onButtonClicked()">Change Name</button>

```

Sự kiện có thể là click, dbClick etc. Sau đó là tên sự kiện.

+ Step 2 . Ta khai báo hàm onButtonCLick trong file user.component.ts. Để sử lý sự kiện khi người dùng click vào

### 2 way binding

+ Step 1 : Chúng ta phải sử dụng ngModule để thực hiện 2 way binding. Vì ngModule là 1 phần của FormsModule. Nên muốn sử dụng được

ngModule thì chúng ta phải import bằng tay FormsModule vào file app.module.

```

<input [(ngModel)]="name" />

```

## Kết nối webservice

+ Trong file cart.service.js import thư viện HttpClientmport.

+ Để gọi webservice phương thức get http.get('http://164.132.226.137:1441/feed') . Trong đó http://164.132.226.137:1441/feed là đường dẫn tới webservice

+ Ví dụ các phương thức get , post , put (cái này mọi người phải đọc thêm rsjs để hiểu vê pipe . Ngoài ra còn khái niệm Promise sử lý bất đồng bộ

+ Post

```

sendJobNotificationManually(){

return this.http.post('/jobs/alert-queues','')

.pipe(

map((response: any) => response)

);

}

```

+ Get

```

getHotJobDetail(jobId) {

return this.http.get(`/hotjobs/${jobId}`)

.pipe(

map((response: any) => response)

);

}

```

+ Put

```

updateHotJobStatus(jobId, status, params = {}) {

return this.http.put<any>(`/hotjobs/${jobId}/${status}`, params)

.pipe(

map((response: any) => response)

);

}

```

+ Gọi webservice và nhận về kết quả bằng dùng method \*\*subscribe\*\*

```

loadInfoOfCurrentJob() {

this.jobService.updateHotJobStatus(this.currentJob.slug).subscribe(response => {

this.currentJob = response;

});

}

```

+ Subcribe

+Khi chúng ta làm việc với Observable thì sẽ thường xuyên sử dụng phương thức subscribe để theo dõi các thay đổi của dữ liệu hoặc nhận về kết quả của một công việc nào đó.

this.http.get(apiUrl).subscribe(val => console.log(val));

+ UnSubcribe

+Khi API trả về kết quả thì dữ liệu đó sẽ được log ra qua phương thức subscribe. Và để kết thúc quá trình lắng nghe kết quả từ API trả về thì chúng ta sẽ xử dụng đến phương thức unsubscribe

this.http.get(apiUrl).subscribe(val => console.log(val)).unsubscribe();

## Observable

+ Vì sao chọn Observable

https://www.learnrxjs.io/

https://techtalk.vn/chon-promise-hay-observable-khi-lam-viec-voi-angular.html

## ES6

https://www.lifewire.com/best-javascript-es6-features-4579821

# Deploy app lên firebase

https://medium.com/@longboardcreator/deploying-angular-6-applications-to-firebase-hosting-b5dacde9c772

# Tham khảo ví dụ tại trang

+ https://angular.io/start

+ https://viblo.asia/p/cung-tim-hieu-kien-truc-cua-angular-2-va-su-khac-nhau-ve-kien-truc-giua-angular-1-va-angular-2-gGJ592qpKX2

+ http://itprotraining.vn/vi/tips-cong-nghe/Kien-truc-tong-quat-cua-Angular%205 <br>

+ https://github.com/aviabird/angularspree/tree/develop/src/app

+ https://www.learnrxjs.io