Tìm hiểu Angular 2

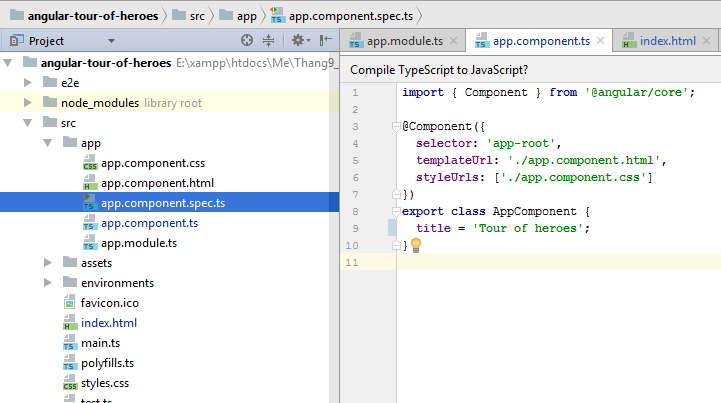
**I: Component**

1. Giới thiệu

- Component là thành phần thực hiện việc render ra các phần view cho end-user nhìn thấy và tương tác. Component có thể sử dụng lại nhiều lần trong 1 ứng dụng.

- Một Angular có ít nhất 1 component. Mặc định khi khởi tạo app Angular mới là AppComponent trong file src/app/app.component.ts

2. Cấu trúc



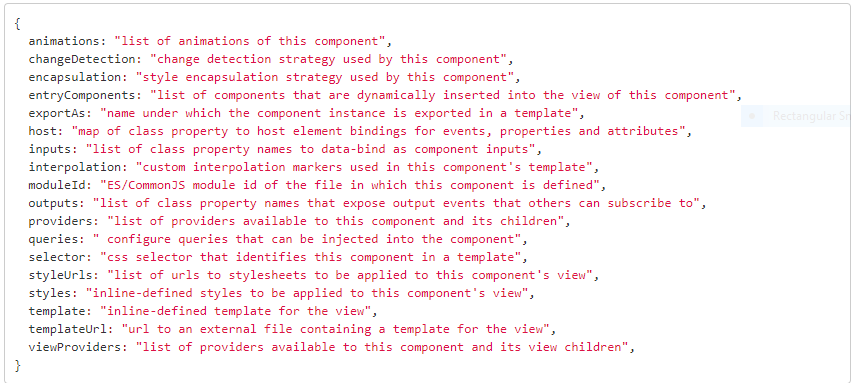
- Dòng 1: Là dòng để import module nào đó hoặc thành phần của một module nào đó để sử dụng.

- @Component({}):

+ Khai báo cho class sau đó, để chỉ ra rằng class đó là một Angular Component. Cung cấp các metadata để Angular biết tạo ra Component tương ứng.

+ @Component yêu cầu tối thiểu truyền vào một javascript object với ít nhất một thuộc tính là: selector và templateURL hoặc là template.

* Selector: chỉ ra rằng khi trong template nếu có 1 thẻ dạng HTML với tên <app-root> thì Angular sẽ hiển thị ra Component đó để thay thế khi render. Ví dụ: index.html
* TemplateURL: để link đến phần template tương ứng của component này
* Template: Dùng để viết template theo kiểu inline. Nếu có nhiều dòng thì mỗi dòng phân cách nhau bởi cặp dấu ‘’ . Không nên viết kiểu inline này.
* StyleURLs: Dùng để khai báo các CSS được sử dụng cho component này



-Dòng 8: Định nghĩa lớp AppComponent, property title, sau đó export cho module khác sử dụng.

3. Tạo mới một component

3.1: Tạo bằng tay

- Tạo file: xxx.component.ts

- Import {} from ‘@angular/core’;

- Sang module muốn sử dụng component này khai báo ở phần import và declarations:

3.2: Dùng angular CLI

- Bật cửa sổ cmd trong thư mục của angular app rồi nhập vào với cú pháp:

Ng g component <component-name>

**II. CSS và phạm vi ảnh hưởng**

- Có 2 cách style css trong angular 2

+ CSS inline style: bằng khai báo thuộc tính style: [‘’] trong @Component

+ CSS external: bằng khai báo thuộc tính stylesUrl: [‘xxx….css’] trong Component

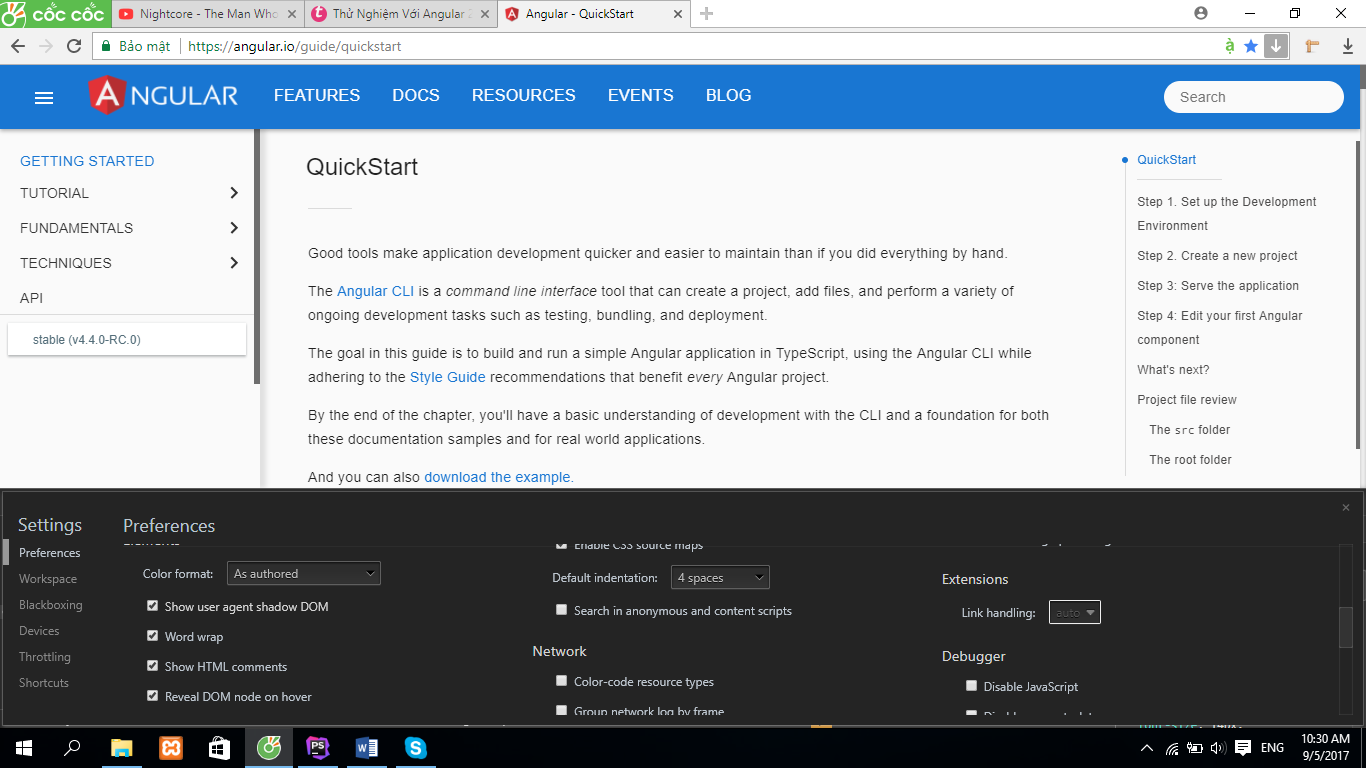
- Các css được viết trong file style.css có phạm vi áp dụng cho toàn bộ app.

**III. View Encapsulation**

3.1: Shadow Dom

- Là một tiêu chuẩn của web component

- Bật shadow Dom

Vào F12-> Setting

- Những phần nào trong shadow dom thì sẽ không bị ảnh hưởng từ bên ngoài. Ví dụ: select tag của html

3.2: View Encapsulation

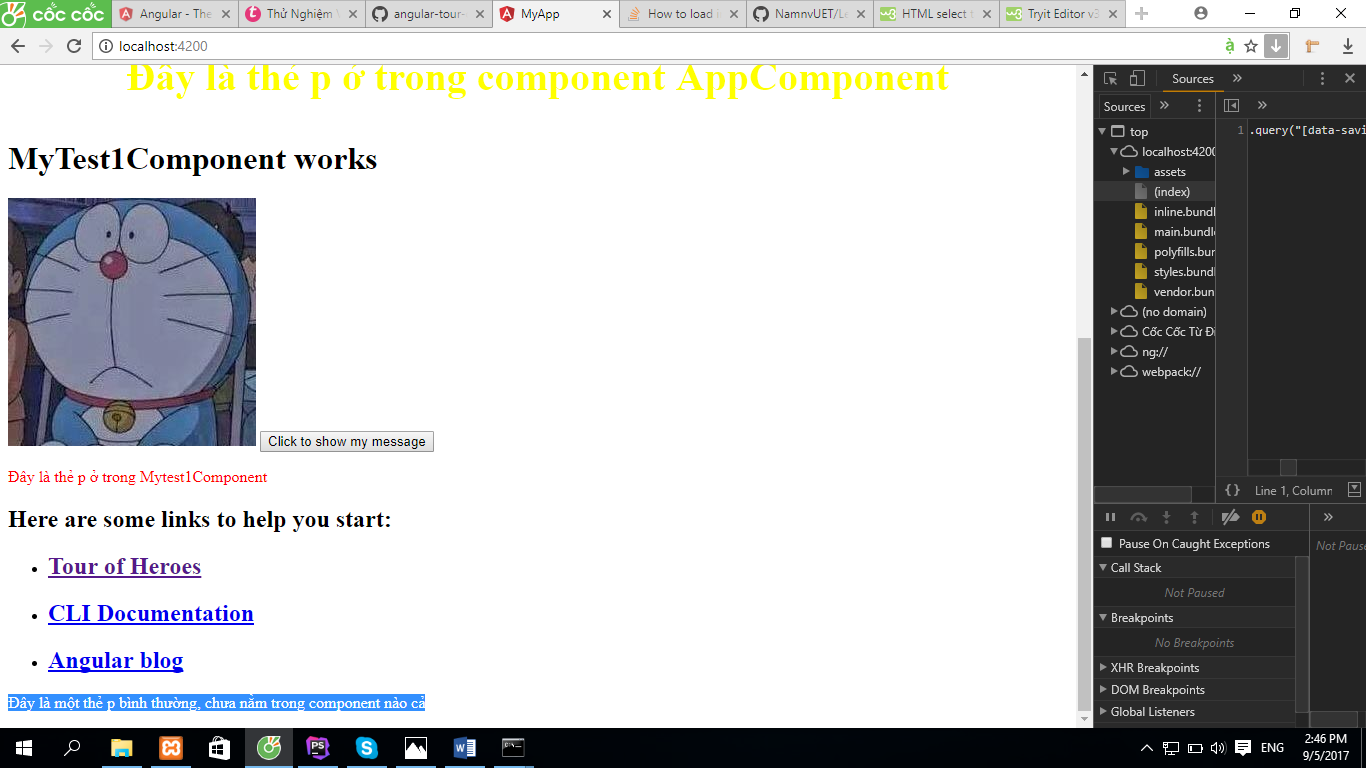
- Angular 2 cung cấp một số cách để có thể đóng gói các component lại. Việc style từ component này sẽ không ảnh hưởng đến component khác.

- Khi một component được tạo ra mà chưa được khai báo về view and capsulation thì sẽ là emulated. Angular sẽ sinh ra các properties riêng để áp dụng các css vào.

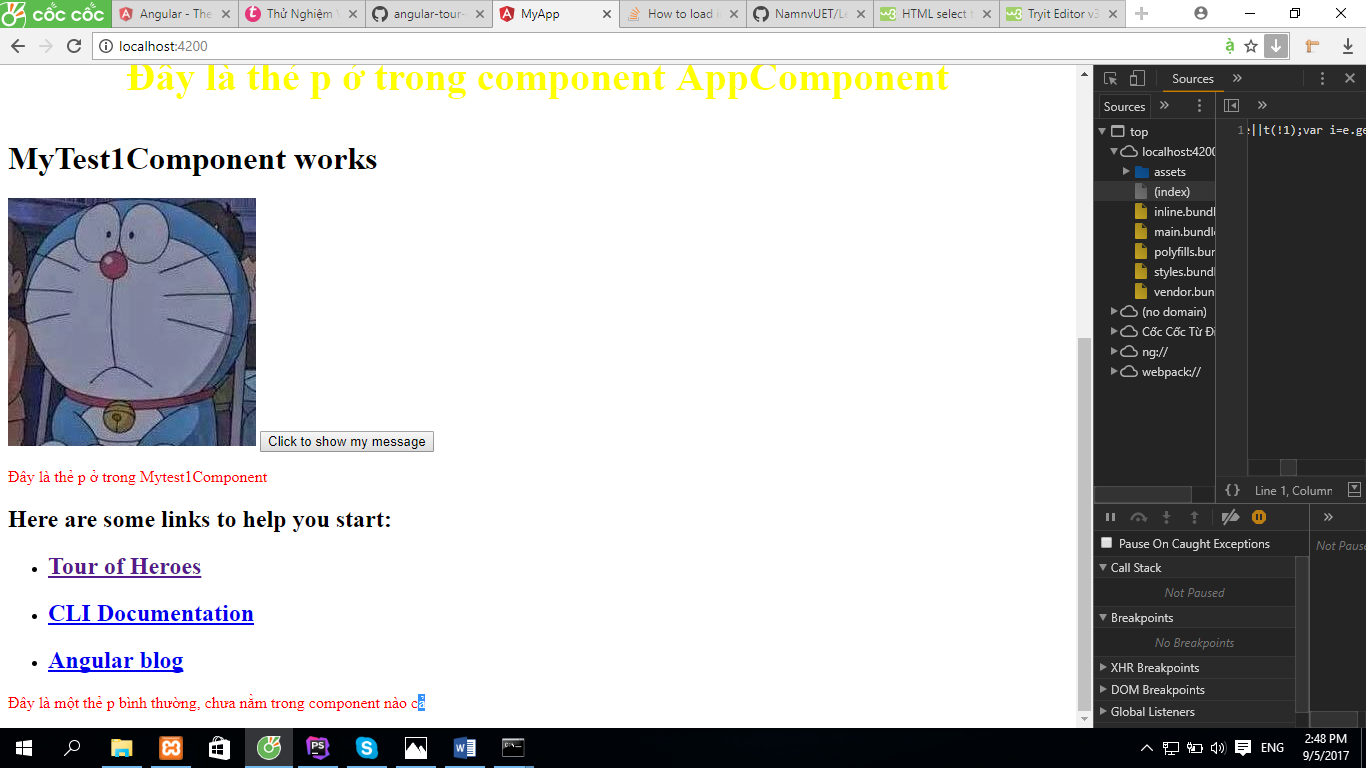
- Chúng ta có thể chỉnh sửa bằng cách thêm thuộc tính encapsulation vào @Component (và khai báo thêm ViewEncapsulation ở phần import)

Ví dụ:

Trước khi thêm encapsulation: ViewEncapsulation.None

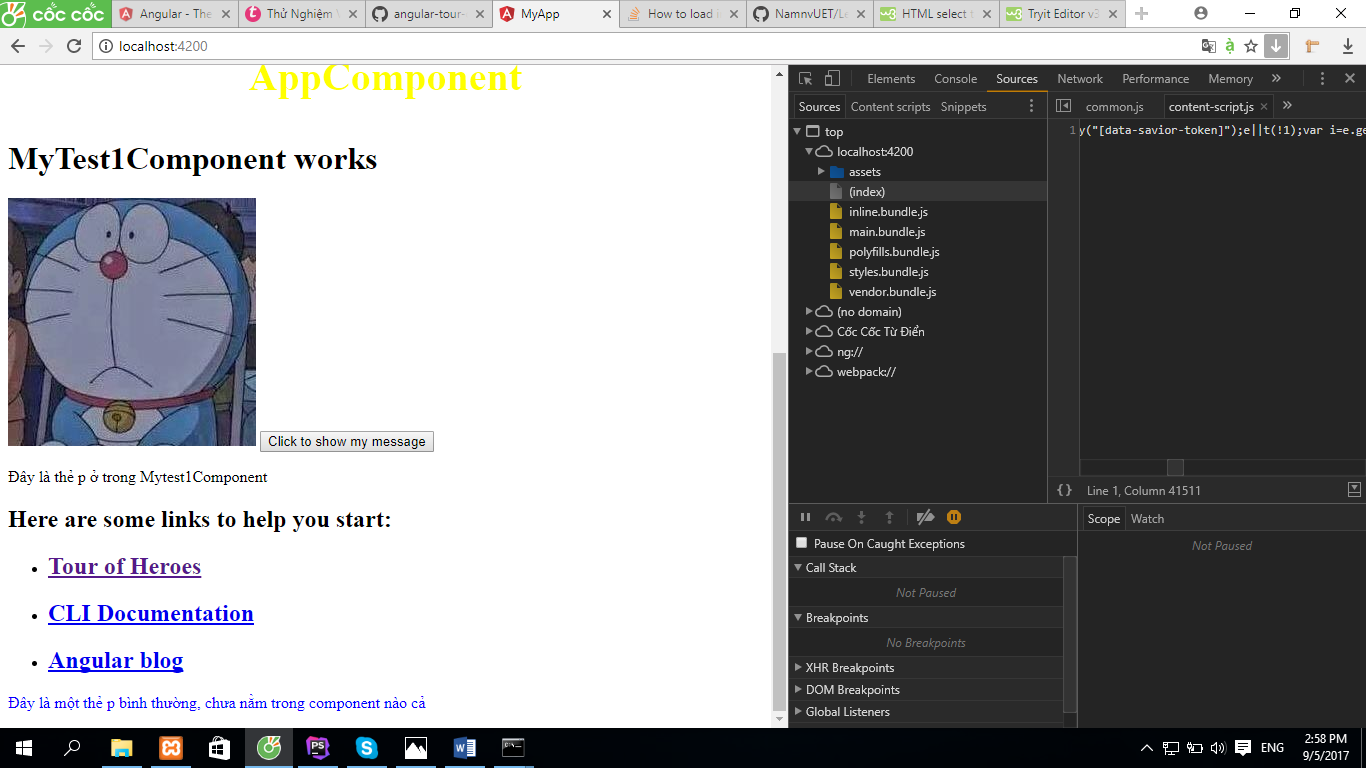


Sau khi thêm encapsulation vào component MyTest1Component



Vì khi thêm ViewEncapsulation.None vào thì phần css của component Mytest1Component sẽ không được đóng gói nữa và phần style color:red cho thẻ p trong mytest1component.component.css sẽ trở thành css global.

Ngoài ra, còn có ViewEncapsulation.Native. Khi sử dụng thì phần css của component đó sẽ đóng gói và sẽ không chịu tác động từ bên ngoài, ví dụ như các css khai báo global

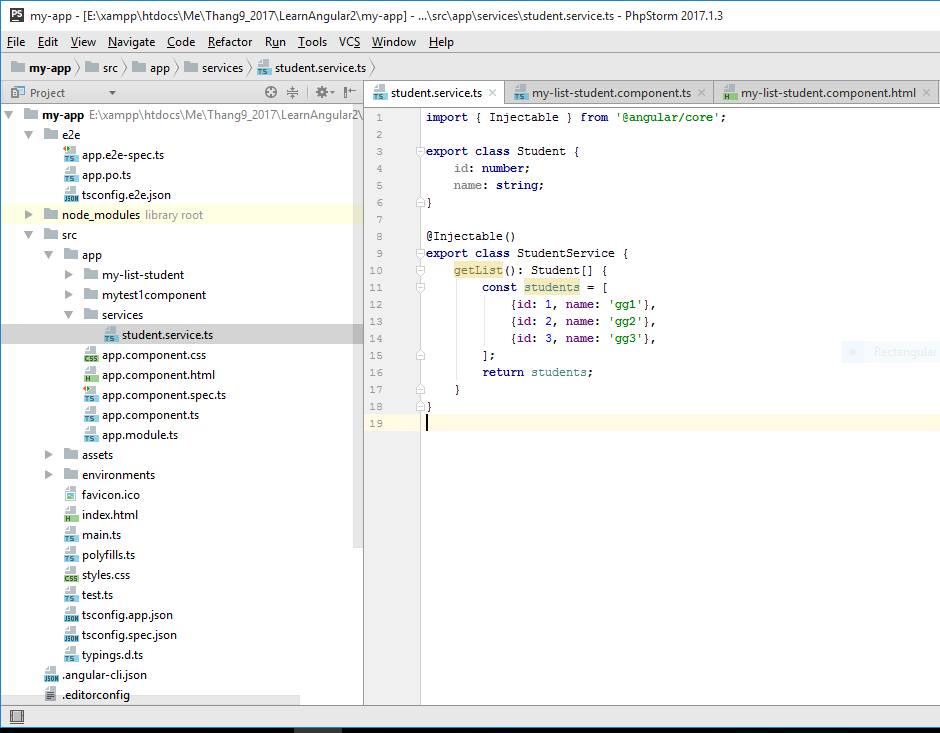


Trong hình, mặc dù có style global (p color: blue) nhưng do trong mytest1Component đã khai báo ViewEncapsulation.Native nên thẻ p trong component đó không chị tác động của style global

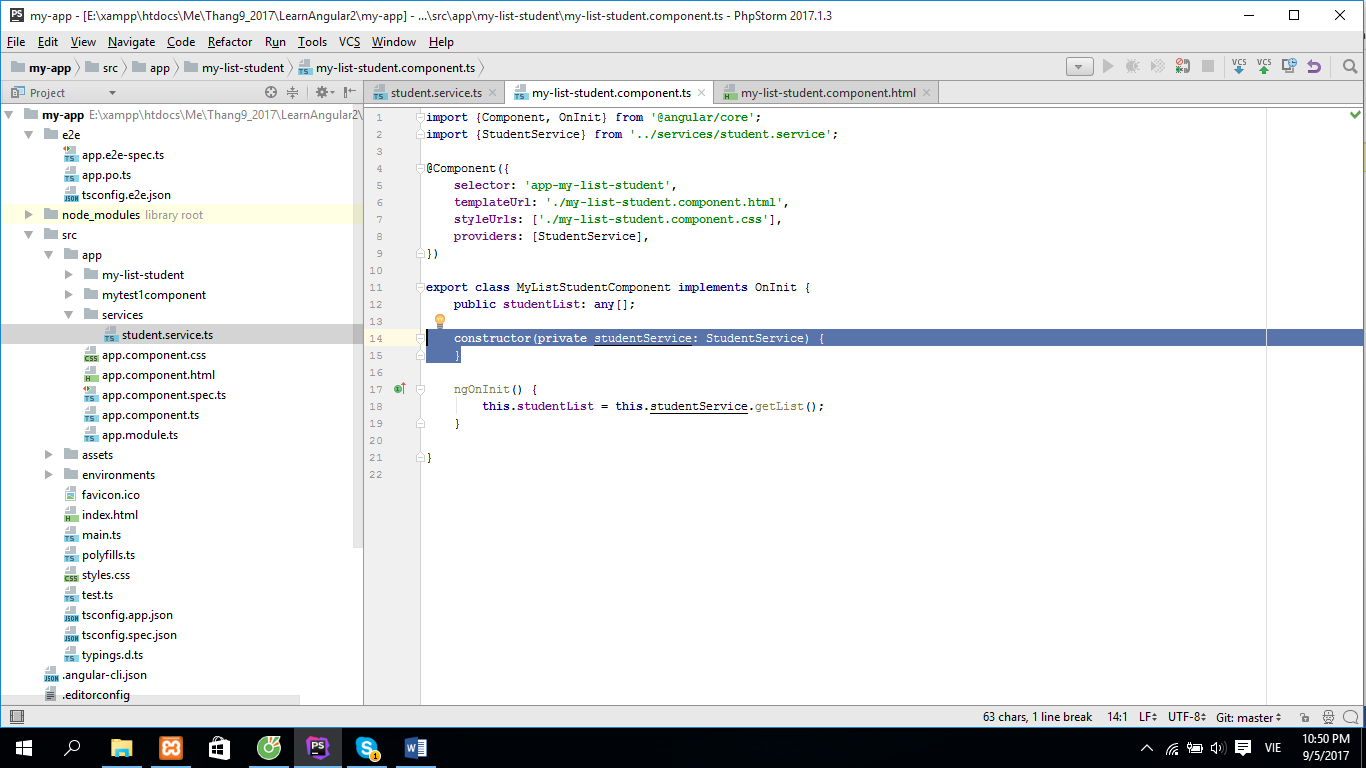
**IV. Service**

**-** Service trong Angular 2 là một lớp đóng gói một vài chức năng được dùng chung cho component hoặc cho các modules.

- Dependency injection: Các ứng dụng trong thực tế làm rất nhiều thứ. Để quản lý hệ thống phức tạp này người ta chia nó thành các hệ thống nhỏ hơn chỉ làm một hoặc một vài việc. Kết quả là chúng ta có một hệ thống đơn giản hơn được tạo thành từ các component nhỏ phụ thuộc lẫn nhau. Nhưng chúng ta muốn các phần phụ thuộc này càng ít phụ thuộc lẫn nhau càng tốt => SOLID, quy tắc dependency injection.



Trong hình là service student, chứa một hàm getList() với kiểu trả về Student[].



Tiêm StudentService vào component qua hàm khởi tạo và đăng kí sử dụng StudentService qua thuộc tính providers trong @component

Note: Có thể đăng kí sử dụng service ở mức module. Khi đó, ta thêm thuộc tính providers trong @Ngmodel

**V. Promises and observable**

- Promises trong ES6 sinh ra để xử lý kết quả của một hành động cụ thể, kết quả của mỗi hành động sẽ là thành công hay thất bại.