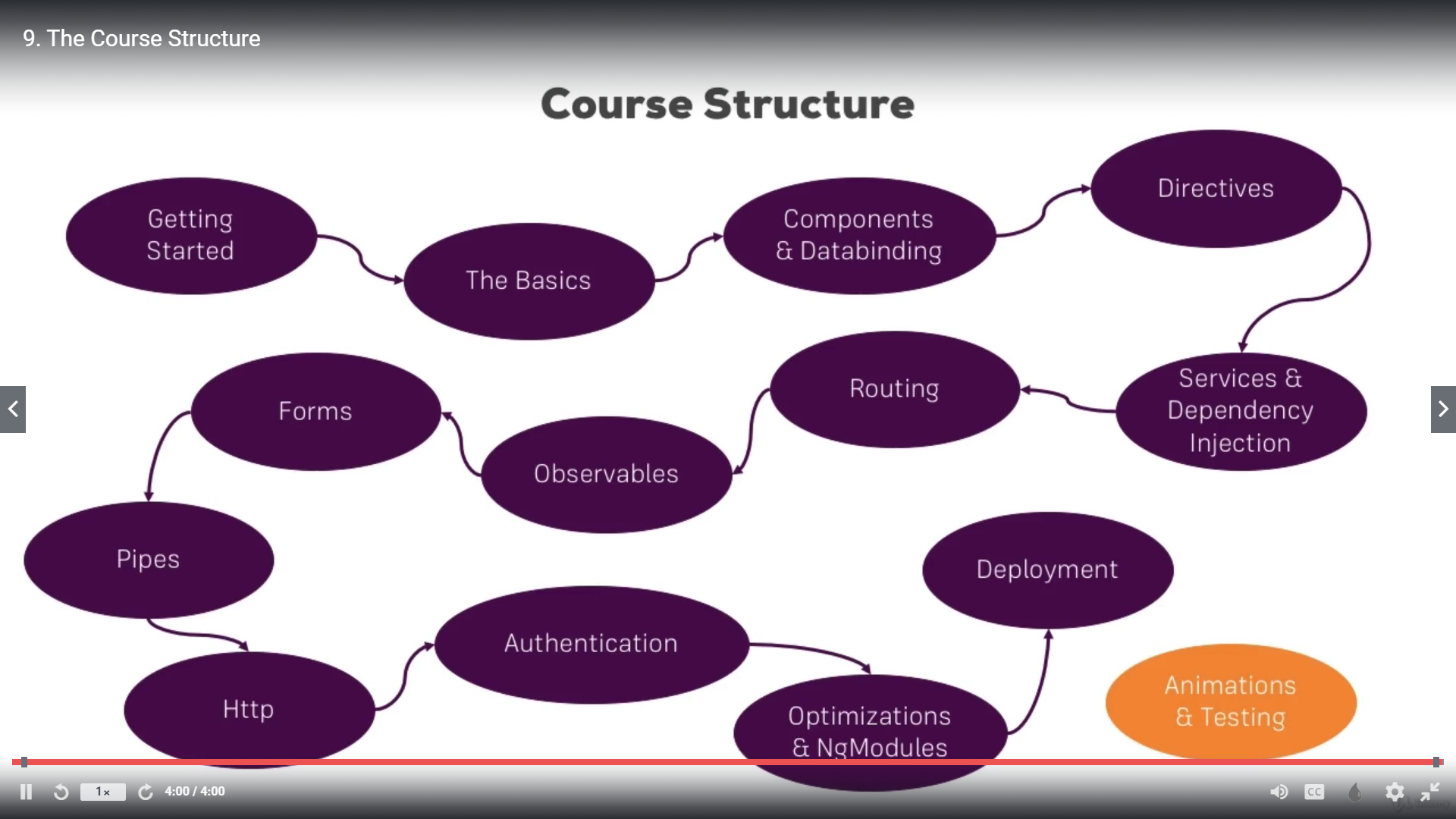
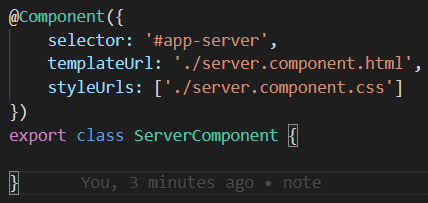
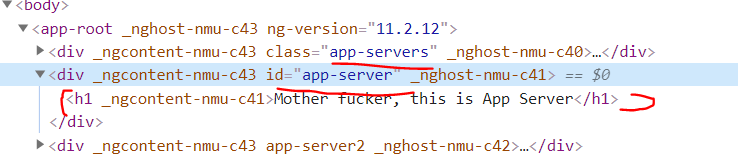
-ngModel là một thành phần của FormsModule, phải import FormsModule thì mới sử dụng được

- ngModel là một directive, dùng [(ngModel)] để gọi.

- ngModel sử dụng cơ chế two-way data binding, đồng bộ dữ liệu giữa component với dom và ngược lại

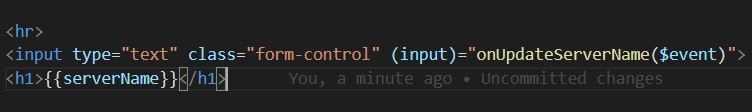
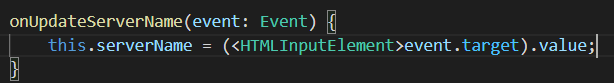
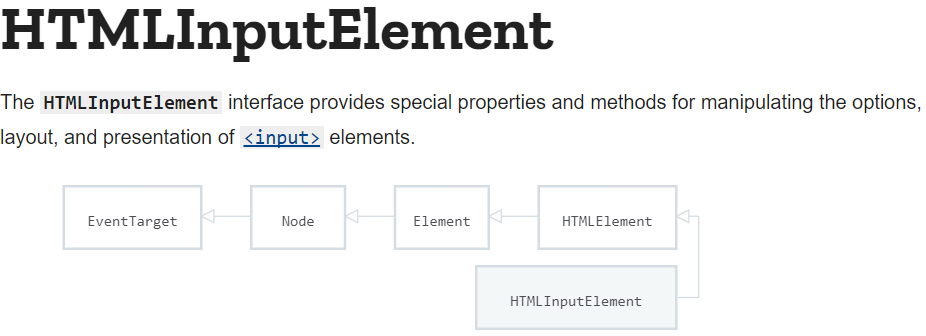
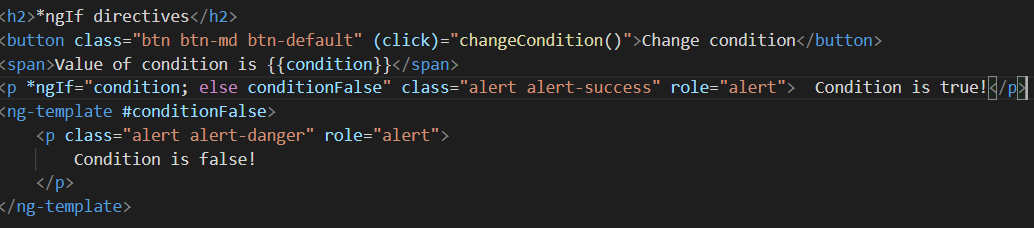
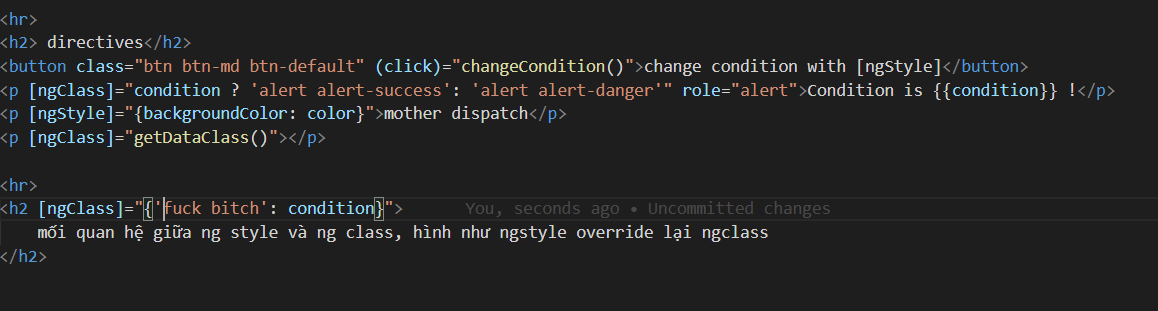
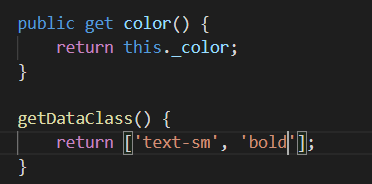
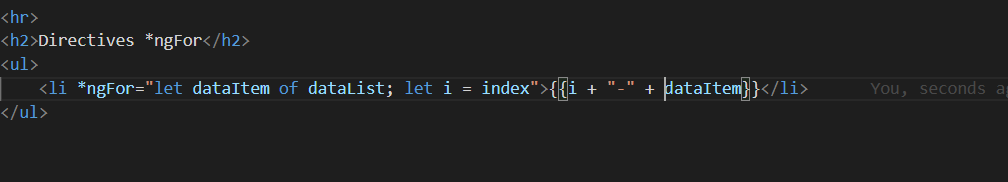
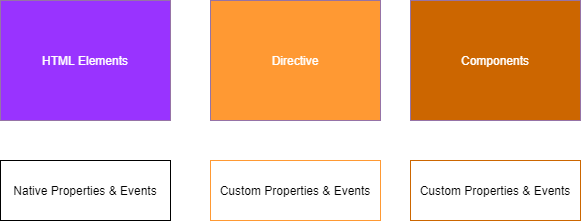
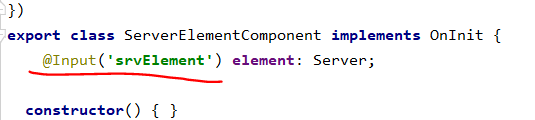
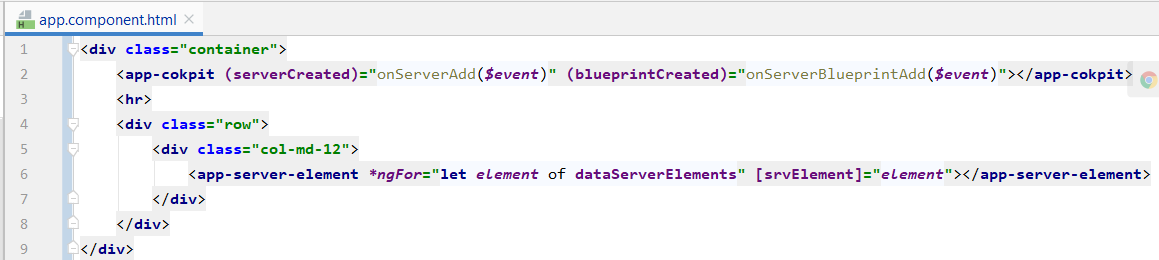
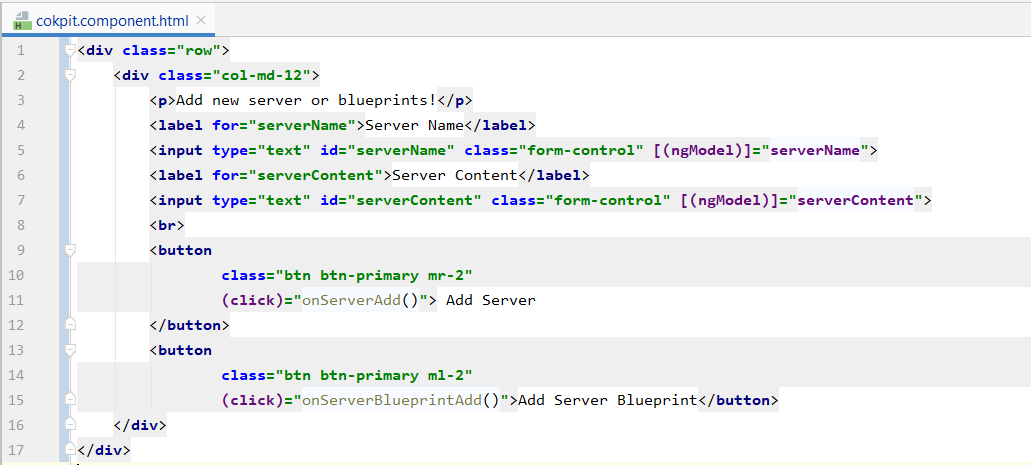
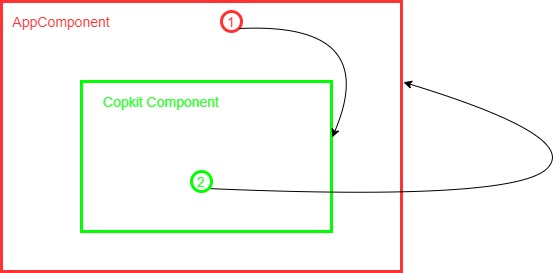
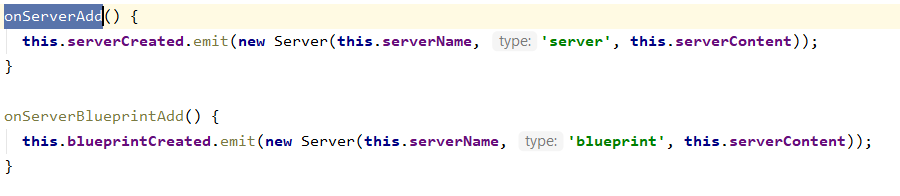
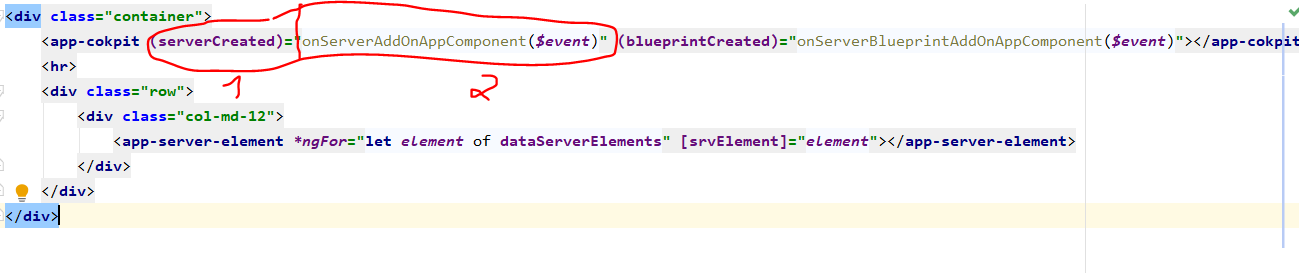
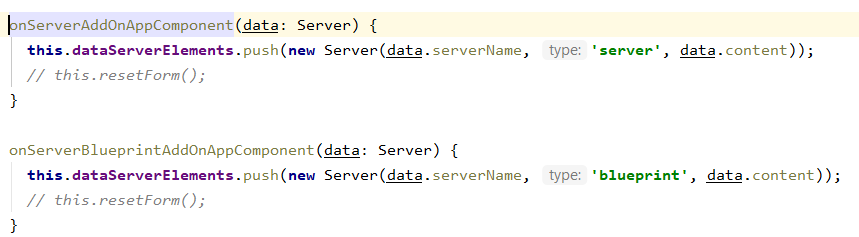
  
selector trong angular component, có 4 kiểu đặt selector:

1. Kiểu mặc định  
     
   Khi gọi component này, ta sử dụng cú pháp : <app-root></app-root>
2. Kiểu thứ 2 là sử dụng mảng:  
     
   Khi khai báo selector kiểu này, ta khai báo với cú pháp  
   <div app-server2></div>
3. Kiểu thứ 3 là dưới dạng #Selectors  
     
   Ta sử dụng cú pháp <div id=”app-server”></div>
4. Kiểu cuối cùng là dạng class như Jquery: .Selector  
     
   ta sử dụng cú pháp : <div class=”app-server”></div>

Đây là các cú pháp selector, tuỳ vào cách khai báo selector mà angular detect theo cách đó mà tìm chỗ khai báo component để rồi ném template của component vào (component sẽ là element con của các thẻ có id hoặc class hoặc attribute ở trên). Giả sử khai báo dạng id thì angular tự tìm element có id là app-server, dạng class thì tự tìm class,…  
  
Nhưng kiểu khuyên dùng vẫn là dạng normal là selector:‘app-root’  
+> Binding attribute , ta sử dụng ngoặc vuông : [disabled] =”allowDisable”

+> String interpolation: {{ dataOutput }} hiển thị data ra ngoài màn hình, khác với Property Binding, bind data ra attribute của HTML Element. Tuy nhiên ta cũng có thể binding data ra màn hình với property binding là [innerText]. Không được mix String interpolation và Data binding. String Interploation cuối cùng cũng sẽ convert về Property Binding. Ta có thể dùng String Interpolation ở property, tuy nhiên là không nên dùng như vậy:  
<button disabled={{allowDisable}}>Login </button> (chỉ support một số các property built in của Angular)

string interpolation thì convert data sang string, còn property binding thì không

1. Event binding ta sẽ dùng dấu ngoặc tròng (click)=”data()”  
     
   Khi mà click thì angular sẽ bind event vào trong hàm onUpdateServerName.  
     
   Mặc định hàm event.target không có chưa function value, để get value của input bằng event, ta phải ép kiểu target về HTMLInputElement.  
     
   \*\* FormModule cần phải có để two-way data binding hoạt động, cần phải enalbe ngModel
2. Directives: là một đối tượng giúp ta thay đổi một đối tượng khác (Directive is instruction in the DOM)  
   có các loại directives:  
   - Components directive: Có thể gọi Component là Directive cmnl, Directive có selector, và nhúng vào html với selector tag  
   - Structural directives: \*ngFor, \*ngIf  
   - Attribute directives: \*ngStyle  
   \*\* Đối với ngIf, template sẽ re-render lại cái đoạn nằm trong if chứ ko hidden đi  
   - Đối với Directives \*ngIf , ngoài cách else bằng (condition ? a : b), ta dùng **ng-template** và **local reference** để else data như sau :  
     
   trỏ tới ng-template bằng local reference  
   - Không giống như Structual Directives: attribute directives chỉ add hoặc remove cái element mà nó điều khiển, ko có add vs remove nhiều elements khác.  
     
     
   Ta có thể cấu hình style bằng object hoặc chuỗi string, cấu hình class bằng 1 object, 1 mảng hoặc một chuỗi string  
   - Directives: \*ngFor  
   
3. **Properties Binding  
   **Chuyên mục truyền dữ liệu qua lại giữa các Components
   1. Concept đầu tiên của Binding Properties là @Input()  
        
      Tạo một property binding mới của ng-component, cho phép gán giá trị cho sererElements từ component khác (appComponent gọi cái thằng AppServerElementComponent và nó truyền data cho thằng AppServerElementComponent). Việc bind data nó sẽ chạy trước khi AppServerElementComponent  
        
      Đôi khi ta muốn đổi tên của @Input, ta dùng cú pháp @Input(‘aliasName’)  
        
      Ngoài ra, ta có thể gán Input ở cái @Component chứ ko cần dùng annotation @Input, tuy nhiên là éo nên dùng  
      Ta có thể dùng @Input ở hàm setter
   2. Component Event với EventEmitter và @Output  
        
      Copkit Component được gọi trong AppComponent.  
        
        
        
      \*\* Chu trình gọi của EventEmitter  
      ****  
      Khi nhấn nút OnServerAdd, sự kiện nhấn sẽ gọi tới hàm onServerAdd() và onBlueprintAdd() của CopkitComponent. Tại đây, với sự khai báo của @Output và EventEmitter  
        
      Sẽ emit data ngược lại AppComponent dưới dạng $event  
        
      và khi đó, hàm onServerAddOnAppComponent() và onServerBlueprintAddOnAppConponent() khai báo ở AppComponent được trigger và gửi data vào  
        
      Tương tự như @Input thì @Output cũng có thể change Alias name để phù hợp với ngữ cảnh code