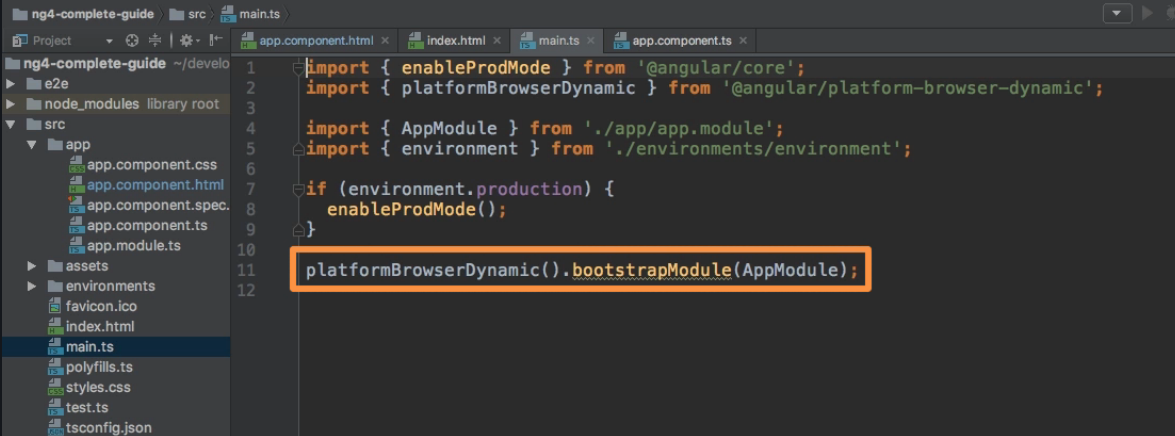
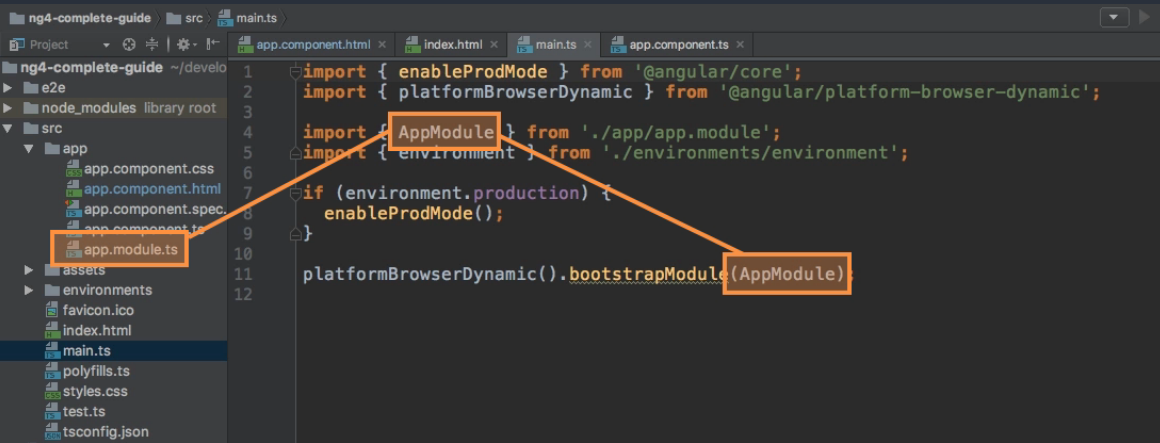
Section2 How Angular Works

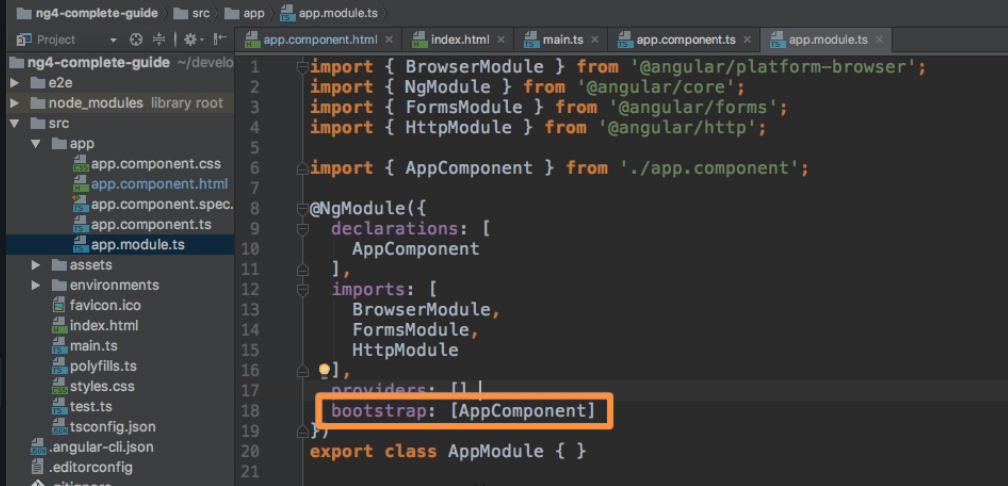
Starts with main.ts



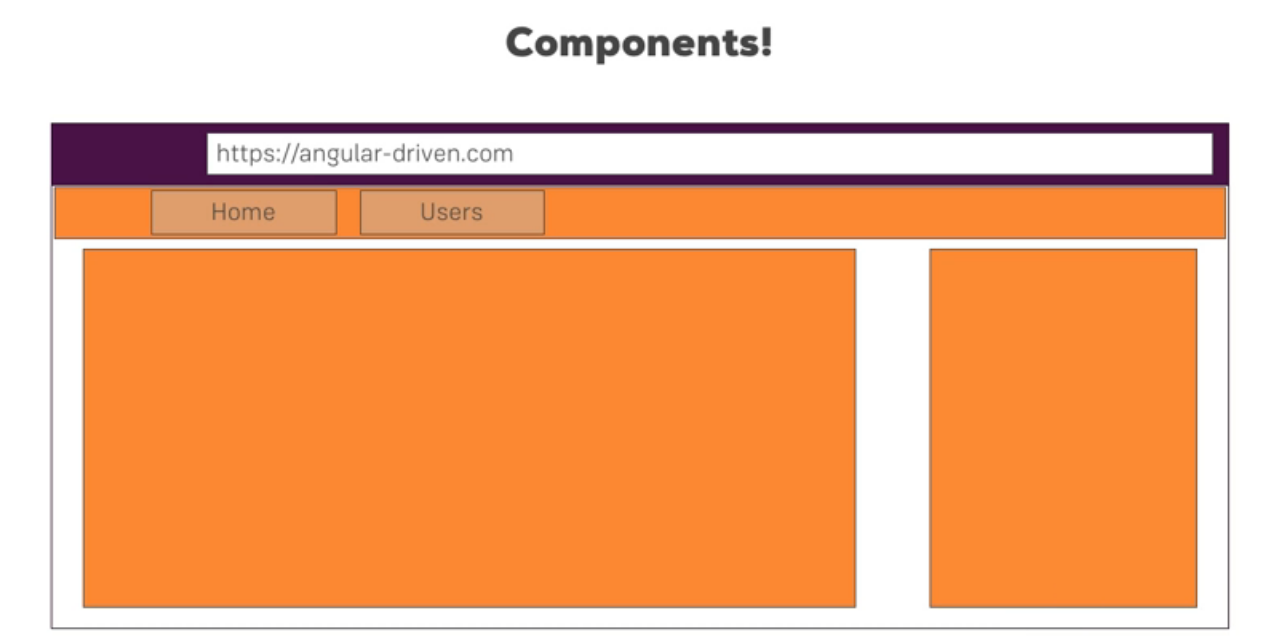
Then bootstrap AppModule and links with app.module.ts



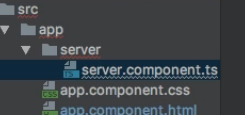
Then app.module.ts asks to bootstrap AppComponent and then resolves selector.

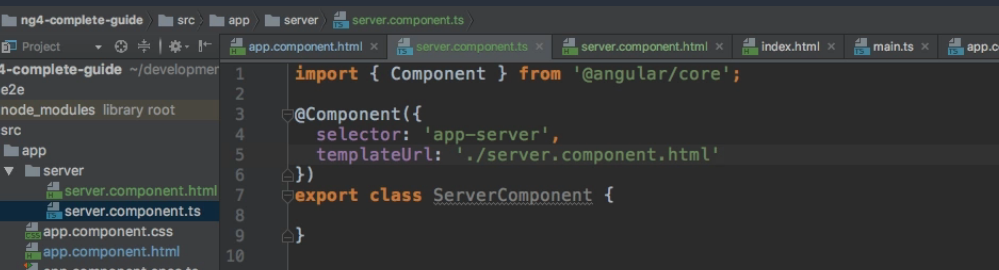


Break Applications into Components



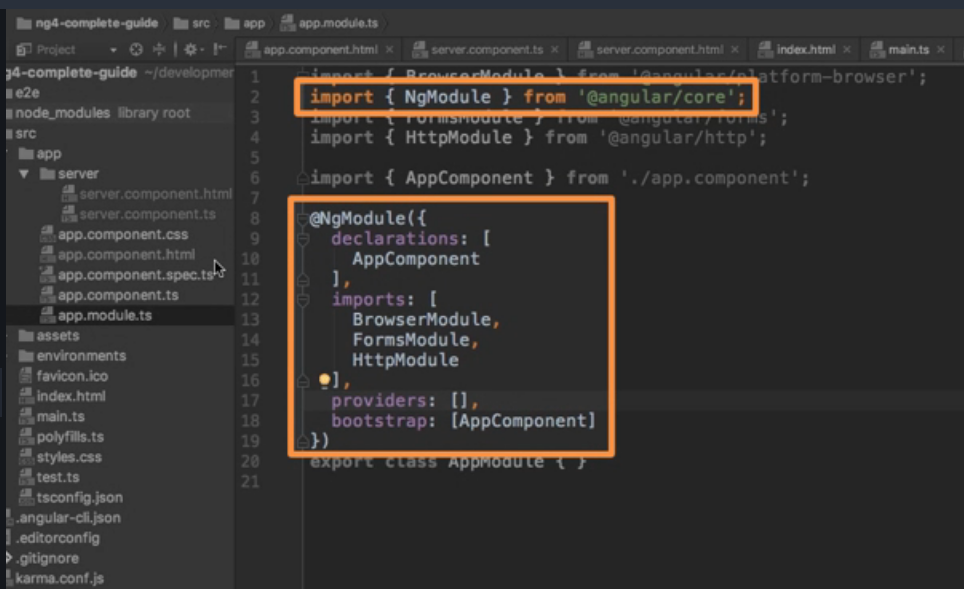
Create component manually.

* Create a folder with same name as name of the component and create a ts file
* i.e. component server would be 

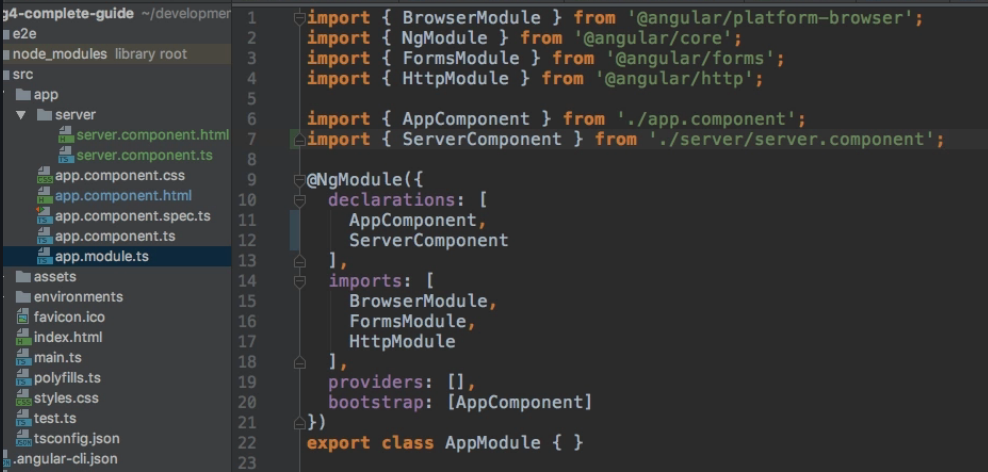


@component is a decorator which we import from angular core and we pass a json configuration object to it having selector and templateUrl which points to html content for that component.

@NgModule is a module decorator



Add your Server component in app.module.ts



Then add your selector i.e. <app-server> in html code of app component.

**Generate Component Using Angular CLI**

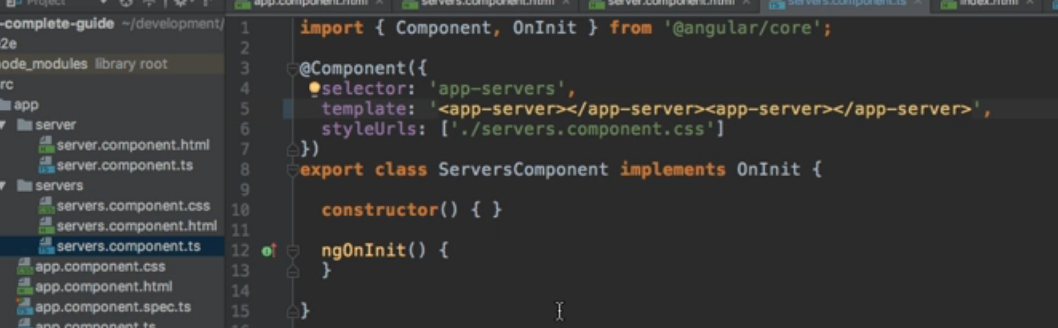
The command should be run from project root where angular.json file is present.

$ng generate component servers

$ng g c servers (shortform)

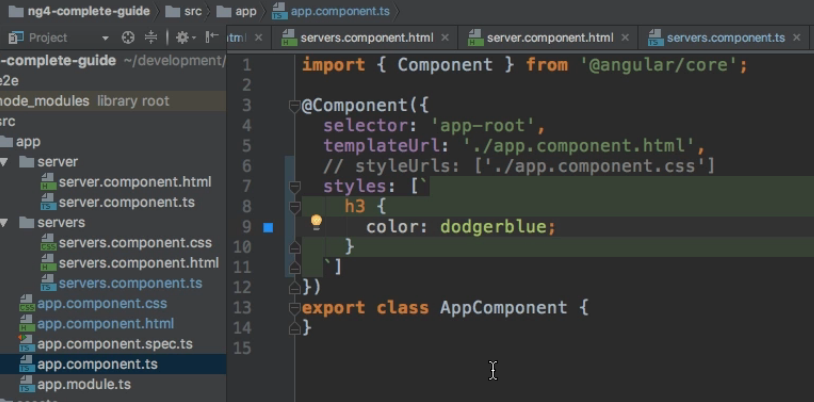
This will create a component with name servers in servers directory and all necessary files i.e. .ts and .html and also add them in app.module.ts

Instead of templateUrl one can use template for inline HTML



One can use backtick ` if you want to have multiline template HTML Code

Styles will go in respective css file referred in **styleUrls** and inline styles can be specified using **styles** array and use backticks to specify multiline styles.



Types of selectors in angular

1. Element Selector:

In TypeScript file selector : ‘app-server’

In HTML file < app-server></ app-server>

1. Attribute Selector:

In TypeScript file selector : ‘[app-server]’

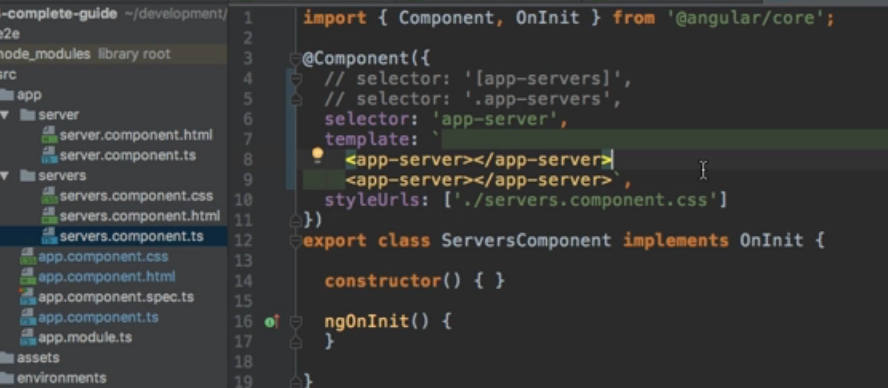
In HTML file <div app-server><div>

1. Style Selector:

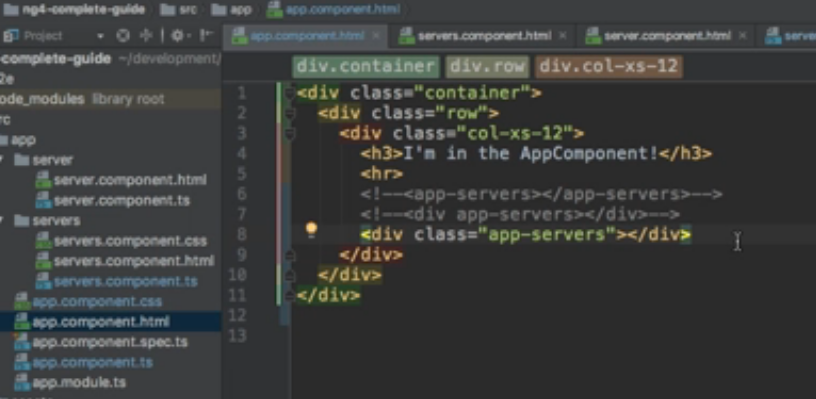
In TypeScript file selector : ‘.app-server’

In HTML file <div class=”app-server”><div>

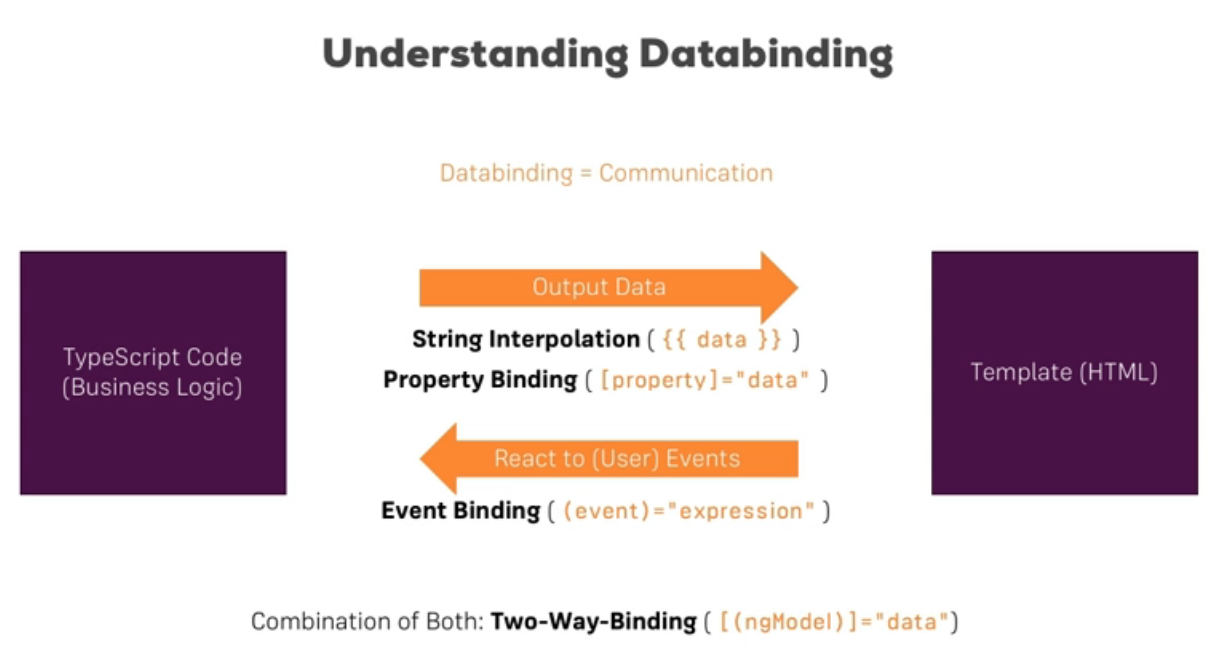
TypeScript



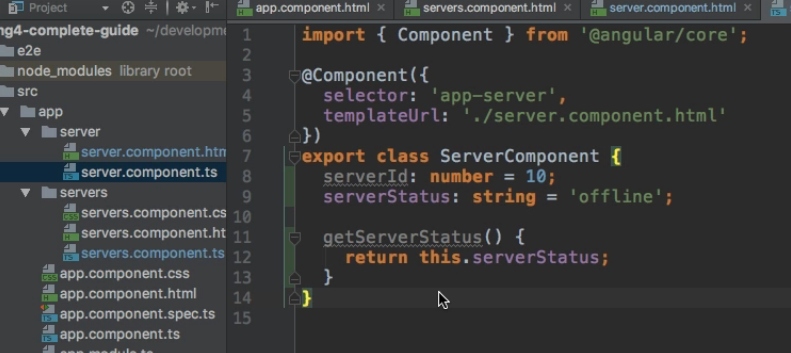
HTML

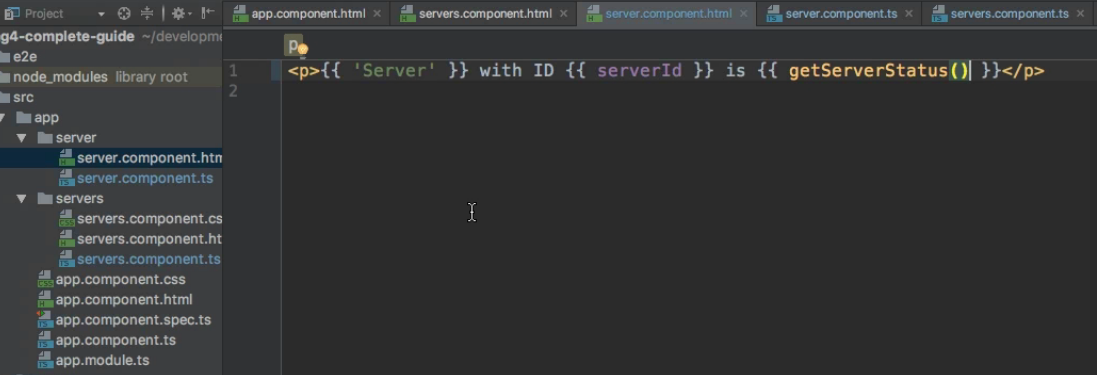


DataBinding in angular 4 types **Look at syntax**



String interpolation: any expression that can be resolved to string can be used in {{ }}

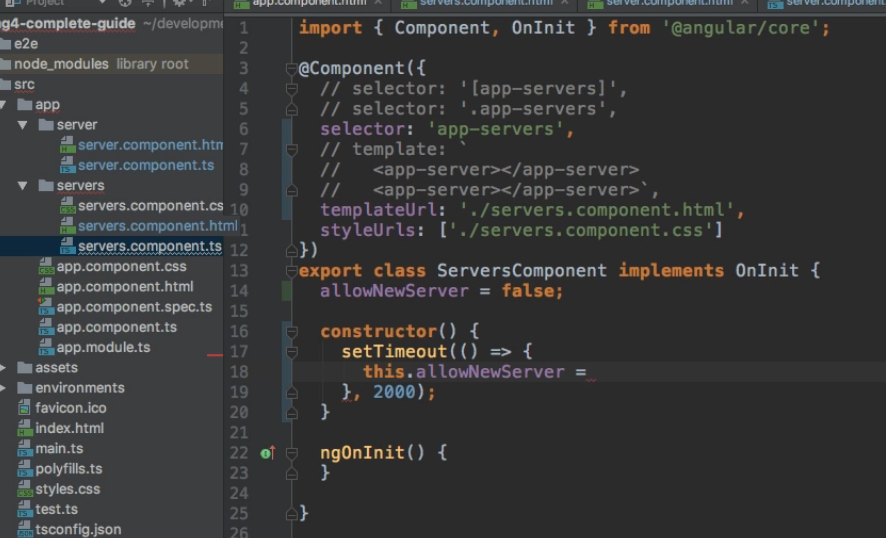


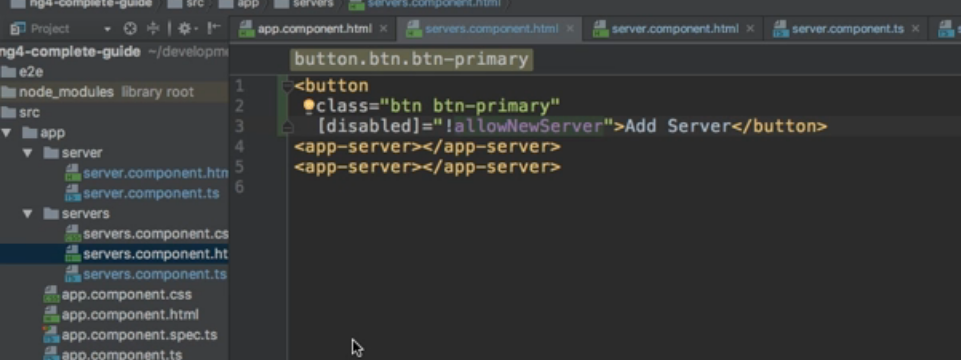


Property Binding:

Square braket indicates that it’s a property binding.

[disabled] = “isDisabled”



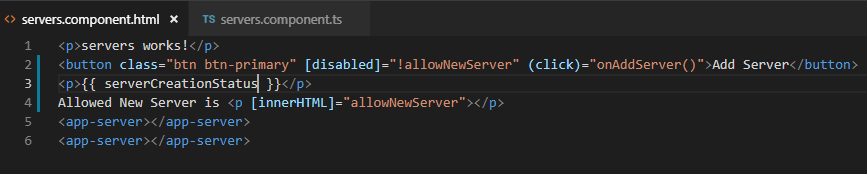


Event Binding

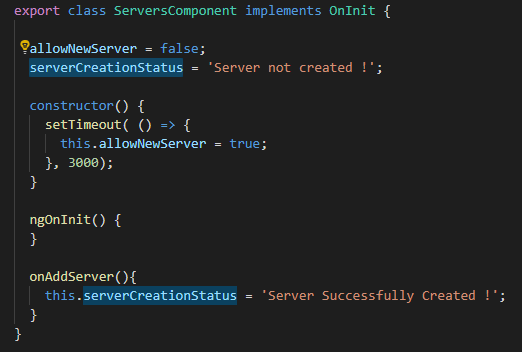
For events, you don't bind to onclick but only to click (=> (click)).

The MDN (Mozilla Developer Network) offers nice lists of all properties and events of the element you're interested in. Googling for YOUR\_ELEMENT properties or YOUR\_ELEMENT events should yield nice results.

HTML bind click event



Add Typescript method



$event will pass event data input by event.

**For Two-Way-Binding to work**, you need to enable the ngModel directive.

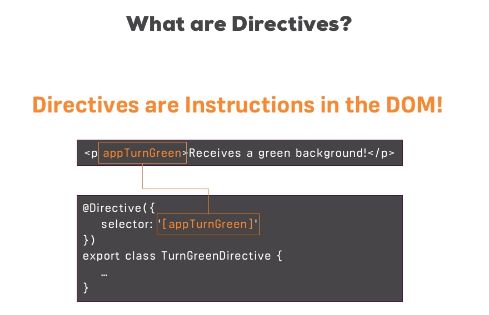
This is done by adding the FormsModule to the imports[] array in the AppModule.

You then also need to add the import from @angular/forms in the app.module.ts file:

import { FormsModule } from '@angular/forms';

<input class="form-control" [(ngModel)]="serverName">

Property serverName is bound to input.



<p \*ngIf="serverName != ''">{{ serverCreationStatus }}</p>

Structural directive ngIf needs \* before it as it changes dom. Element is not hidden here but gets introduced newly in DOM.

Ng If Else Syntax

<p \*ngIf="serverName != ''; else noServer">{{ serverCreationStatus }}</p>

<ng-template #noServer>

<p> No Server Created !</p>

</ng-template>