My first app

Simply create a file named my-first-app.all components get added.

No change in css,spects

1. App.module.ts

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

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

import { AppComponent } from './app.component';

@NgModule({

declarations: [

AppComponent

],

imports: [

BrowserModule,

FormsModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

we just import formsModule and add it in imports

1. App.component.ts

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

myname = 'my-app is';

}

Here we give the variable name and its content or logic which is to be used in html

1. App.comp.html

<input type="text" [(ngModel)]="myname">

<p>{{myname}}</p>

o/p

my-app is----its in text box.whatever printed here gets printed below as well

my-app –this is below printed.

………………………………………………………………………………………

ADDING BOOTSTRAP

In previous my fisrt app,just delete html codes we used ,remove myname and its logic from app.component.ts.remove imports of form from app.module.ts.browser will look empty.

Now we go to terminal and type ng install –save bootstrap@3

Here we go to angular.json

And in styles we add a single line

"styles": [

"node\_modules/bootstrap/dist/css/bootstrap.min.css",

"src/styles.css"

],

now finaly bootstrap is added into our file which will be used later.

………………………………………………….

Components

Method 1

If we want to create components,we should go to app and create a folder named server.inside it we are to create files names server.component.html and server.component.ts

In app.module.ts we go and check if imports of server is there or not. If not we import and add the same in ngmodule

In server.component.ts,we write the following

import { Component } from "@angular/core";

@Component({

selector:'app-server',

templateUrl:'./server.component.html'

})

export class ServerComponent{

}

App.component.html

<h3>I am in the component app</h3>

Setver html

<h2>hi baby there</h2>

App.module.ts

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

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

import {HttpModule} from '@angular/Http';

import { AppComponent } from './app.component';

import {ServerComponent} from './server/server.component';

import { ServersComponent } from './servers/servers.component';

@NgModule({

declarations: [

AppComponent,

ServerComponent,

ServersComponent

],

imports: [

BrowserModule,

FormsModule,

HttpModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

o/p

I am in my component app

hi baby there

…………………..

Method 2

Use command

Ng g component servers

All files will be created

Add imports and its name in ngmodule at app.module.ts

App.comp.html

<h3>I am in my component app</h3>

<hr>

<app-servers></app-servers>

In servers.html

<app-server></app-server>

<app-server></app-server>

Servers.comp.ts

import { Component, OnInit } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: './servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

constructor() { }

ngOnInit() {

}

}

I am in my component app

hi baby there

hi baby there

………………………………

String interpolation

Just like before app.module.ts where we call and import both server and servers

Now we write the logic in server.ts

import { Component } from "@angular/core";

@Component({

selector:'app-server',

templateUrl:'./server.component.html'

})

export class ServerComponent{

serverId:number=10;

serverStatus:string='offLine';

}

Next we go to server.html and call them by their ids

<h2>{{'Server'}} with ID {{serverId}} is {{serverStatus}}</h2>

Now moving to servers.comp.ts where we use its html internally in template and call app server in it

import { Component, OnInit } from '@angular/core';

@Component({

selector: 'app-servers',

template: `

<app-server></app-server>

<app-server></app-server>`,

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

constructor() { }

ngOnInit() {

}

}

o/p

I am in my component app

Server with ID 10 is offLine

Server with ID 10 is offLine

…………………………………………………………………………………………………………………………..

Property binding

Ts servers

import { Component, OnInit } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: '/servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

allowNewServer=false;

constructor() {

setTimeout(() =>{

this.allowNewServer=true;

},2000);

}

ngOnInit() {

}

}

Servers html

<button class="btn btn primary" [disabled]="!allowNewServer">Add Server</button>

<app-server></app-server>

<app-server></app-server>

o/p

a button is created which remains unreactive or disabled for 2 secs and then abled

…………………………………………………………………………………….

Event binding

Servers.ts

import { Component, OnInit } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: '/servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

allowNewServer=false;

serverCreationStatus="no server was created";

constructor() {

setTimeout(() =>{

this.allowNewServer=true;

},2000);

}

ngOnInit() {}

onCreateServer()

{

this.serverCreationStatus="server was created";

}

}

Servers.html

<button class="btn btn primary"

[disabled]="!allowNewServer"

(click)="onCreateServer()">Add Server</button><br>

<!--{{allowNewServer}}property binding-->

<!--<p [innerText]="allowNewServer"></p> //string interpolation generaly used vt txt-->

<p>{{serverCreationStatus}}</p>

<app-server></app-server>

<app-server></app-server>

o/p

here after 2 secs when button becomes enabled onclick of it we get to see the line changed from server not created to server was created

……………………………………………………………..

Passing and using data with event binding

import { Component, OnInit, Input } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: '/servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

allowNewServer=false;

serverCreationStatus="no server was created";

serverName='';

constructor() {

setTimeout(() =>{

this.allowNewServer=true;

},2000);

}

ngOnInit() {}

onCreateServer()

{

this.serverCreationStatus="server was created";

}

onUpdateServerName(event:any){

//console.log(event);

this.serverName=(<HTMLInputElement>event.target).value;

}

}

Html

<label>Server Name:</label>

<input type="text" class="form-control"

(input)="onUpdateServerName($event)">

<P>{{serverName}}</P>

<button class="btn btn primary"

[disabled]="!allowNewServer"

(click)="onCreateServer()">Add Server</button><br>

<!--{{allowNewServer}}property binding-->

<!--<p [innerText]="allowNewServer"></p> //string interpolation generaly used vt txt-->

<p>{{serverCreationStatus}}</p>

<app-server></app-server>

<app-server></app-server>

………………………….

TS file:

import { Component, OnInit, Input } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: '/servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

allowNewServer=false;

serverCreationStatus="no server was created";

serverName='Testserver';

constructor() {

setTimeout(() =>{

this.allowNewServer=true;

},2000);

}

ngOnInit() {}

onCreateServer()

{

this.serverCreationStatus="server was created";

}

onUpdateServerName(event:any){

//console.log(event);

this.serverName=(<HTMLInputElement>event.target).value;

}

}

HTML File:

<input type="text" class="form-control"

(input)="onUpdateServerName($event)">

<input type="text" class="form-control"

[(ngModel)]="serverName">

<p>{{serverName}}</p>

<button class="btn btn primary"

[disabled]="!allowNewServer"

(click)="onCreateServer()">Add Server</button><br>

<!--{{allowNewServer}}property binding-->

<!--<p [innerText]="allowNewServer"></p> //string interpolation generaly used vt txt-->

<p>{{serverCreationStatus}}</p>

<app-server></app-server>

<app-server></app-server>

Output :

I am in my component app



ok swthrt

Add Server

no server was created

Server with ID 10 is offLine

Server with ID 10 is offLine

……………………

Combining 4 forms of binding

o/p

I am in my component app



Testserver 2

Add Server-🡪it’s a button

server was created..name is:Testserver 2

Server with ID 10 is offLine

Server with ID 10 is offLine

…………………………….

Ngif

Ts file

import { Component, OnInit, Input } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: '/servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

allowNewServer=false;

serverCreationStatus="no server was created";

serverName='Testserver';

serverCreated=false;

constructor() {

setTimeout(() =>{

this.allowNewServer=true;

},2000);

}

ngOnInit() {}

onCreateServer()

{

this.serverCreated=true;

this.serverCreationStatus="server was created..name is:" +this.serverName;

}

onUpdateServerName(event:any){

//console.log(event);

this.serverName=(<HTMLInputElement>event.target).value;

}

}

Html

<label>Server Name:</label>Server Name

<input type="text" class="form-control"

(input)="onUpdateServerName($event)">

<input type="text" class="form-control"

[(ngModel)]="serverName">

<!--<p>{{serverName}}</p>-->

<button class="btn btn primary"

[disabled]="!allowNewServer"

(click)="onCreateServer()">Add Server</button><br>

<!--{{allowNewServer}}property binding-->

<!--<p [innerText]="allowNewServer"></p> //string interpolation generaly used vt txt-->

<!--<p>{{serverCreationStatus}}</p>-->

<p \*ngIf="serverCreated">Server was created,name is{{serverName}}</p>

<app-server></app-server>

<app-server></app-server>

o/p

I am in my component app

Server Name:Server NameAdd Server

Server was created,name istrsysys 2

Server with ID 10 is offLine

Server with ID 10 is offLine

……………………………………………..

If else condition same as above just 1 line chnge in html

<label>Server Name:</label>Server Name

<input type="text" class="form-control"

(input)="onUpdateServerName($event)">

<input type="text" class="form-control"

[(ngModel)]="serverName">

<!--<p>{{serverName}}</p>-->

<button class="btn btn primary"

[disabled]="!allowNewServer"

(click)="onCreateServer()">Add Server</button><br>

<!--{{allowNewServer}}property binding-->

<!--<p [innerText]="allowNewServer"></p> //string interpolation generaly used vt txt-->

<!--<p>{{serverCreationStatus}}</p>-->

<p \*ngIf="serverCreated else noServer">Server was created,name is{{serverName}}</p>

<ng-template #noServer><p>no server created</p></ng-template>

<app-server></app-server>

<app-server></app-server>

…………………………………

Ng style

Server.ts

import { Component } from "@angular/core";

@Component({

selector:'app-server',

templateUrl:'./server.component.html'

})

export class ServerComponent{

serverId:number=10;

serverStatus:string='offLine';

constructor(){

this.serverStatus = Math.random() >0.6 ? 'online':'offline';

}

getColor(){

return this.serverStatus== 'online'?'green':'red';

}

}

Server.html

<h2 [ngStyle]="{backgroundColor:getColor()}">{{'Server'}} with ID {{serverId}} is {{serverStatus}}</h2>

I am in my component app

Server Name:Server NameAdd Server

no server created

Server with ID 10 is offline

Server with ID 10 is online

……………………………………..

Applying css class dynamically with ngstyle

Ts

import { Component } from "@angular/core";

@Component({

selector:'app-server',

templateUrl:'./server.component.html',

styles:[`

.online{

color: white;

}

`]

})

export class ServerComponent{

serverId:number=10;

serverStatus:string='offLine';

constructor(){

this.serverStatus = Math.random() >0.6 ? 'online':'offline';

}

getColor(){

return this.serverStatus== 'online'?'green':'red';

}

}

Html

<h2 [ngStyle]="{backgroundColor:getColor()}"

[ngClass] ="{online:serverStatus==='online'}">

{{'Server'}} with ID {{serverId}} is {{serverStatus}}</h2>

o/p

I am in my component app

Server Name:Server NameAdd Server

no server created

Server with ID 10 is online

Server with ID 10 is online

……………………….

List with ngFor

Ts servers

import { Component, OnInit, Input } from '@angular/core';

@Component({

selector: 'app-servers',

templateUrl: '/servers.component.html',

styleUrls: ['./servers.component.css']

})

export class ServersComponent implements OnInit {

allowNewServer=false;

serverCreationStatus="no server was created";

serverName='Testserver';

serverCreated=false;

servers=['Testserver','testserver2'];

constructor() {

setTimeout(() =>{

this.allowNewServer=true;

},2000);

}

ngOnInit() {}

onCreateServer()

{

this.serverCreated=true;

this.servers.push('serverName');

this.serverCreationStatus="server was created..name is:" + this.serverName;

}

onUpdateServerName(event:any){

//console.log(event);

this.serverName=(<HTMLInputElement>event.target).value;

}

}

Html servers

<label>Server Name:</label>Server Name

<input type="text" class="form-control"

(input)="onUpdateServerName($event)">

<input type="text" class="form-control"

[(ngModel)]="serverName">

<!--<p>{{serverName}}</p>-->

<button class="btn btn primary"

[disabled]="!allowNewServer"

(click)="onCreateServer()">Add Server</button><br>

<!--{{allowNewServer}}property binding-->

<!--<p [innerText]="allowNewServer"></p> //string interpolation generaly used vt txt-->

<!--<p>{{serverCreationStatus}}</p>-->

<p \*ngIf="serverCreated else noServer">Server was created,name is{{serverName}}</p>

<ng-template #noServer><p>no server created</p></ng-template>

<app-server \*ngFor="let server of servers"></app-server>

o/p

I am in my component app

Server Name:Server NameAdd Server

Server was created,name isTestserver

Server with ID 10 is online

Server with ID 10 is offline

Server with ID 10 is offline

Server with ID 10 is offline

Server with ID 10 is offline

Server with ID 10 is offline

Server with ID 10 is offline

Server with ID 10 is offline

………………………………….