



3416-003 Web Development Using Angular 8 and TypeScript Module 4: Angular Components



UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 1: Section 1

What Is This Course About

Installed Bootstrap 4.2.1

- Created directory C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version1
- In dos-prompt: cd C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version1
- Step 1: Type “ng new bootstrap4”
- Click “Yes” for “Would you like to add Angular routing”.
- Click “enter” for “Which stylesheet format would you like to use?”.
- Cd into directory: “cd bootstrap4”

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1>ng new bootstrap4
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? CSS
```

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1>cd bootstrap4
```

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1\bootstrap4>
```

Installed Bootstrap 4.1.3

- Type following command:
- Step 2: npm install popper.js@1.14.3

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1\bootstrap4>npm install popper.js@1.14.3
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"}
} (current: {"os":"win32","arch":"x64"})

+ popper.js@1.14.3
added 1 package from 2 contributors and audited 43277 packages in 12.474s
found 0 vulnerabilities
```

Installed Bootstrap 4.1.3

- Type following command:
- Step 3: npm install [jquery@3.4.1](#)

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1\bootstrap4>npm install jquery@3.4.1
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":any}
"} (current: {"os":"win32","arch":"x64"})

+ jquery@3.4.1
added 1 package from 1 contributor and audited 43278 packages in 11.555s
Found 0 vulnerabilities

C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1\bootstrap4>
```

Installed Bootstrap 4.1.3

- Type following command:
- Step 4: npm install bootstrap@4.1.3

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1\bootstrap4>npm install bootstrap@4.1.3
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ bootstrap@4.1.3
added 1 package from 2 contributors and audited 43279 packages in 11.523s
found 1 moderate severity vulnerability
  run `npm audit fix` to fix them, or `npm audit` for details

C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version1\bootstrap4>
```

Installed Bootstrap 4.2.1

- Type following command:
- Step 5: Open C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version1\bootstrap4\angular.json and add the following:
 - "node_modules/bootstrap/dist/css/bootstrap.min.css"
- "styles": [
 - "src/styles.css",
"node_modules/bootstrap/dist/css/bootstrap.min.css"
•]

Installed Bootstrap 4.2.1

```
angular.json
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "bootstrap4": {
      "root": "",
      "sourceRoot": "src",
      "projectType": "application",
      "prefix": "app",
      "schematics": {},
      "architect": {
        "build": {
          "builder": "@angular-devkit/build-angular:browser",
          "options": {
            "outputPath": "dist/bootstrap4",
            "index": "src/index.html",
            "main": "src/main.ts",
            "polyfills": "src/polyfills.ts",
            "tsConfig": "src/tsconfig.app.json",
            "assets": [
              "src/favicon.ico",
              "src/assets"
            ],
            "styles": [
              "src/styles.css",
              "node_modules/bootstrap/dist/css/bootstrap.min.css"
            ],
            "scripts": []
          },
          "es5BrowserSupport": true
        },
        "configurations": {
          "production": {
            "fileReplacements": [
              {
                "replace": "src/environments/environment.ts",
                "with": "src/environments/environment.prod.ts"
              }
            ],
            "optimization": true,
            "outputHashing": "all",
            "sourceMap": false,
            "extractCss": true,
            "namedChunks": false,
            "aot": true,
            "extractLicenses": true,
            "vendorChunk": false,
            "buildOptimizer": true,
            "budgets": [
              {
                "type": "initial",
                "maximumWarning": "2mb",
                "maximumError": "5mb"
              }
            ]
          }
        }
      }
    }
  }
}
```

Add this line



Installed Bootstrap 4.2.1

- Type following command:
- Step 6: Open C:\workspace-UT-Angular8-SEPT-2020
 \session4\example1-bootstrap4-version1
 \bootstrap4\angular.json and add the following:

```
"scripts": [  
    "node_modules/jquery/dist/jquery.slim.min.js",  
    "node_modules/bootstrap/dist/js/bootstrap.min.js",  
    "node_modules/popper.js/dist/umd/popper.min.js"  
]
```

Installed Bootstrap 4.2.1

- Type following command:
- Step 6: Open C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version1\bootstrap4\angular.json and add the following:

```
"scripts": [  
    "node_modules/jquery/dist/jquery.slim.min.js",  
    "node_modules/bootstrap/dist/js/bootstrap.min.js",  
    "node_modules/popper.js/dist/umd/popper.min.js"  
]
```

Installed Bootstrap 4.2.1

- Type following command:

```
"projects": {  
  "bootstrap4": {  
    "root": "",  
    "sourceRoot": "src",  
    "projectType": "application",  
    "prefix": "app",  
    "schematics": {},  
    "architect": {  
      "build": {  
        "builder": "@angular-devkit/build-angular:browser",  
        "options": {  
          "outputPath": "dist/bootstrap4",  
          "index": "src/index.html",  
          "main": "src/main.ts",  
          "polyfills": "src/polyfills.ts",  
          "tsConfig": "src/tsconfig.app.json",  
          "assets": [  
            "src/favicon.ico",  
            "src/assets"  
          ],  
          "styles": [  
            "src/styles.css",  
            "node_modules/bootstrap/dist/css/bootstrap.min.css"  
          ],  
          "scripts": [  
            "node_modules/jquery/dist/jquery.slim.min.js",  
            "node_modules/bootstrap/dist/js/bootstrap.min.js",  
            "node_modules/popper.js/dist/umd/popper.min.js"  
          ],  
          "environmentSource": "environments/environment.ts"  
        }  
      }  
    }  
  }  
}
```

Installed Bootstrap 4.2.1

- Type following command:
- Step 7: Open C:\workspace-UT-Angular7-May-2019\session7\example1-bootstrap4-version1\bootstrap4\ng serve -o

Installed Bootstrap 4.2.1

- Right click inspect and choose “Elements” tab.

The screenshot shows the browser's developer tools with the "Elements" tab selected. The code displayed is the source map and part of the Bootstrap 4.1.3 CSS file. The code includes a sourceMappingURL directive, a large base64 encoded sourceMappingURL, and the main CSS content which includes Bootstrap's license information and color variables.

```
/*#
sourceMappingURL=data:application/json;base64,eyJ2ZXJzaW9uIjozLCJzb3VyY2VzIjpBInNyYy9zdHlsZXMuY3NzIl0sIm5hbWVzIjpBXSwibWFwcGluZ3MiOiJBQUFBLDhFQUE4RSIsImZpbGUiOiJzcmMvc3R5bGVzLmNzcyIsInNvdXJjZXNDb250ZW50IjpBIi8qIFlvdSBjYW4gYWRkIGdsb2JhbCBzdHlsZXMgdG8gdGhpccyBmaWx1LCBhbmQgYWxzbyBpbXBvcnQgb3RoZXIgc3R5bGUgZmlsZXMuKi9cbiJdfQ== */
</style>
▼<style type="text/css">
  /*!
   * Bootstrap v4.1.3 (https://getbootstrap.com/)
   * Copyright 2011-2018 The Bootstrap Authors
   * Copyright 2011-2018 Twitter, Inc.
   * Licensed under MIT
   (https://github.com/twbs/bootstrap/blob/master/LICENSE)
   *:root{--blue:#007bff;--indigo:#6610f2;--purple:#6f42c1;--pink:#e83e8c;--red:#dc3545;--orange:#fd7e14;--yellow:#ffc107;--green:#28a745;--teal:#20c997;--cyan:#17a2b8;--white:#fff;--gray:#6c757d;--gray-dark:#33334d;--primary:#007bff;--secondary:#6c757d;--
```

html head

How an Angular App gets Loaded and Started

- C:\workspace-UT-Angular8-SEPT-2020\\session4\example1-bootstrap4-version1\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version2
- Run “ng serve –o”.
- In C:\workspace-UT-Angular8-SEPT-2020\\session4\example1-bootstrap4-version2\bootstrap4\src\app\app.component.html delete the content and replaced with “I’m in the AppComponent!”

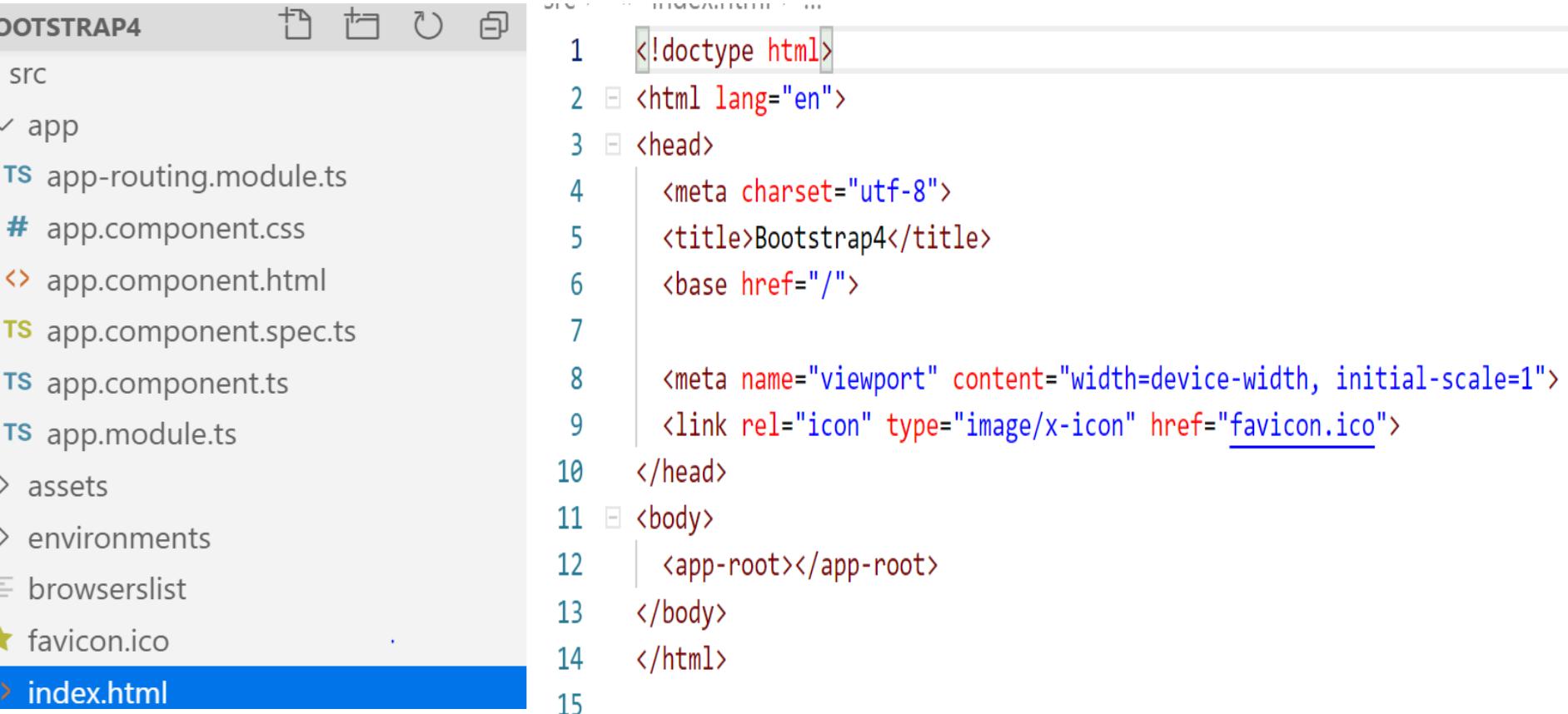
```
src ▶ app ▶ <▶ app.component.html ▶ ...  
1   I'm in the AppComponent! |  
2  
3   <router-outlet></router-outlet>  
4
```

How an Angular App gets Loaded and Started

- How does our browser or how does a server hosting our app know that it should render the content of app component html. You could argue, it's the only component we have right now and we will take a closer look at component's soon. But that is not the reason. And actually this is not the file served by the server. Instead the index HTML file is served by the server and remembered that angular is a framework which allows you to create single page application. This is the single page which has served the index HTML file.

How an Angular App gets Loaded and Started

- Now if you have a look at this index.html file we see that this is a normal HTML file.(C:\workspace-UT-Angular8-SEPT-2020 \session4\example1-bootstrap4-version2\bootstrap4\src\index.html)



The screenshot shows a code editor interface with a sidebar on the left displaying the project structure of a Bootstrap 4 Angular application named "DOTSTRAP4". The "index.html" file is selected and highlighted in blue at the bottom of the sidebar.

The main editor area displays the content of "index.html", which is a standard HTML document structure:

```
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>Bootstrap4</title>
6     <base href="/">
7
8     <meta name="viewport" content="width=device-width, initial-scale=1">
9     <link rel="icon" type="image/x-icon" href="favicon.ico">
10    </head>
11   <body>
12     <app-root></app-root>
13   </body>
14 </html>
15
```

The code editor has syntax highlighting, with tags like <!doctype>, <html>, <head>, <meta>, <title>, <base>, <viewport>, <link>, <body>, and <app-root> colored in various shades of red, blue, and green. Line numbers are visible on the left side of the code.

How an Angular App gets Loaded and Started

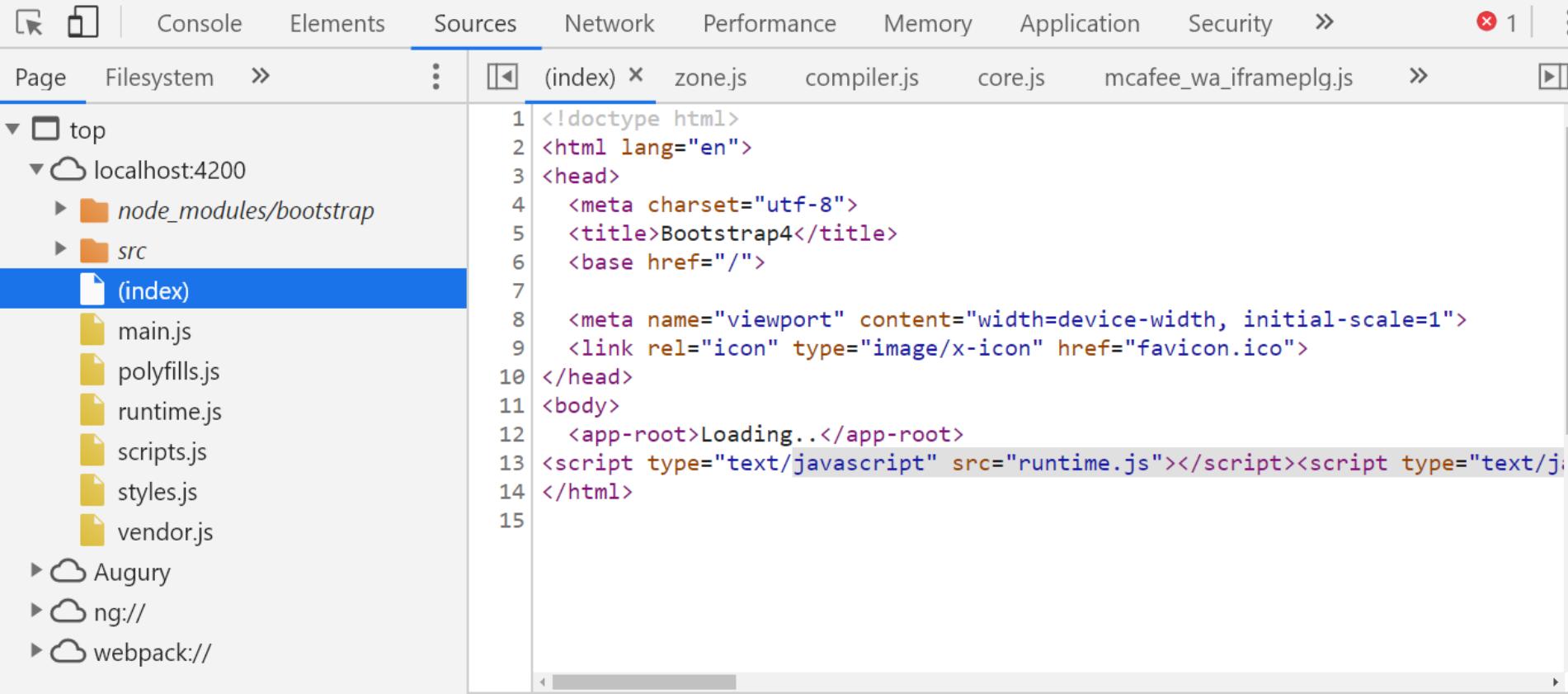
- The body of the index.html file is very interesting. We get this ‘<app-root></app-root>’ . We added:
- <app-root>Loading..</app-root>

```
<body>
  <app-root>Loading..</app-root>
</body>
```

How an Angular App gets Loaded and Started

- Don't see "loading.." in the output in the browser:

I'm in the AppComponent!

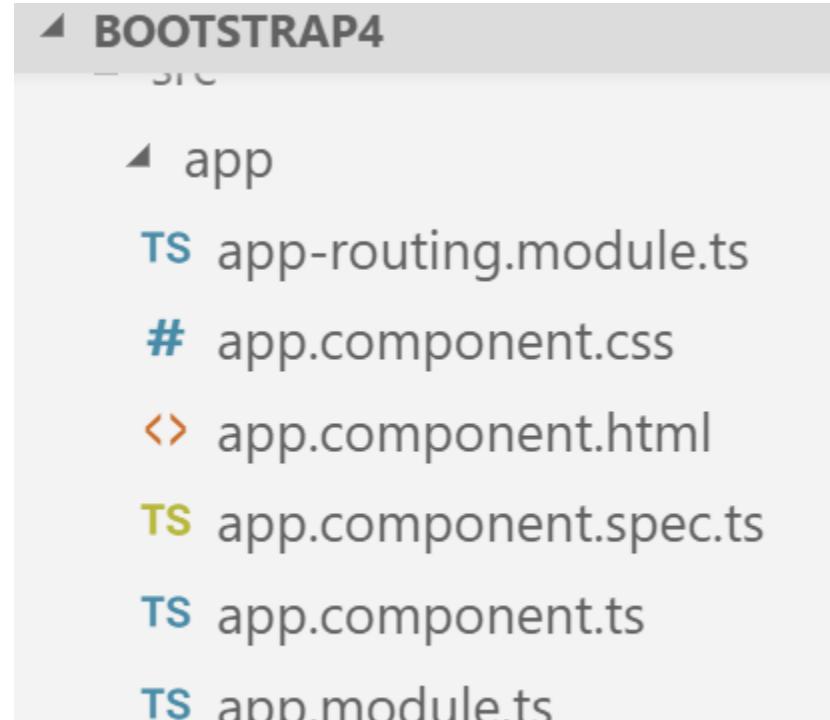


The screenshot shows the Chrome DevTools interface with the 'Sources' tab selected. On the left, the file tree shows the project structure with 'localhost:4200/index.html' selected. The right pane displays the source code of the index.html file.

```
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>Bootstrap4</title>
6     <base href="/">
7
8     <meta name="viewport" content="width=device-width, initial-scale=1">
9     <link rel="icon" type="image/x-icon" href="favicon.ico">
10    </head>
11    <body>
12      <app-root>Loading..</app-root>
13      <script type="text/javascript" src="runtime.js"></script><script type="text/j...</script>
14    </body>
15  </html>
```

How an Angular App gets Loaded and Started

- “<app-root>” here is of course not our default html element. Instead as is one of our own components we will soon dive into how we create our own components but the CLI created one for us. The root component of our application, the component which will tie together our whole application in the end and all the files in the app folder here which have component in their name.



How an Angular App gets Loaded and Started

- The “app.component.ts” file

The screenshot shows a code editor interface with the following details:

- EXPLORER** panel on the left:

 - OPEN EDITORS**: app.component.html, index.html, **app.component.ts** (selected).
 - BOOTSTRAP4** folder:
 - app
 - app-routing.module.ts**
 - # **app.component.css** (highlighted)
 - app.component.html
 - app.component.spec.ts**
 - app.component.ts** (highlighted)
 - app.module.ts
 - OUTLINE**: html

- app.component.html** tab in the top bar.
- index.html** tab in the top bar.
- app.component.ts** tab in the top bar.
- File Structure:** src > index.html > html > body > app-root
- Code Content (app.component.ts):**

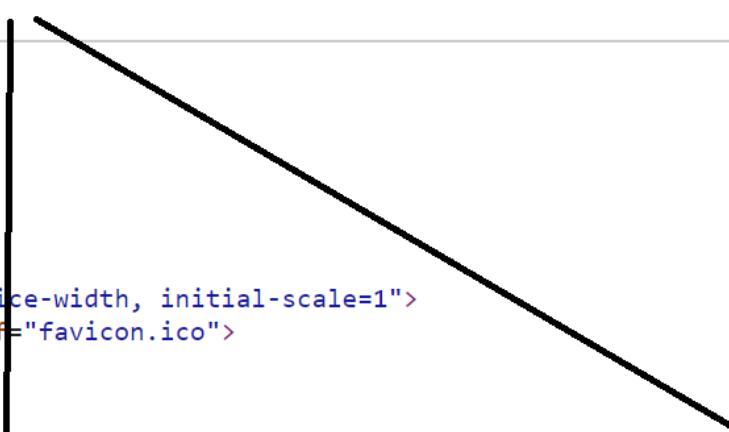
```
src > index.html > html > body > app-root
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf-8">
5  <title>Bootstrap4</title>
6  <base href="/">
7
8  <meta name="viewport" content="width=device-width, initial-scale=1">
9  <link rel="icon" type="image/x-icon" href="favicon.ico">
10 </head>
11 <body>
12   <app-root>Loading..</app-root>
13 </body>
14 </html>
15
```

- Annotations:**
 - A black rectangle highlights the entire content of the `app.component.ts` code block.
 - A black rectangle highlights the `<app-root>Loading..</app-root>` line.
 - A black arrow points from the bottom of the highlighted `app.component.ts` code to the annotation text at the bottom right.
- Annotation Text:** '<app-root>' replaced by app.component.html, app.component.ts and app.component.css

How an Angular App gets Loaded and Started

- How is Angular triggered, how is it kicked off to actually run over our 'app.component.html' file? The answer is in the final index page of the file getting served in the browser and a bunch of javascript files are injected by the CLI.

These javascript files are injected by the CLI automatically.



```
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>Bootstrap4</title>
6   <base href="/">
7
8   <meta name="viewport" content="width=device-width, initial-scale=1">
9   <link rel="icon" type="image/x-icon" href="favicon.ico">
10 </head>
11 <body>
12   <app-root>Loading..</app-root>
13   <script type="text/javascript" src="runtime.js"></script><script type="text/javascript" src="es2015-polyfills.js">
14 </body>
15 </html>
```

How an Angular App gets Loaded and Started

- So in the final file, script imports here are present and these javascript script files are there for executed and they're actually the first code to be executed.

Main.ts

- Main.ts is the first code get executed.

```
src ▶ TS main.ts ▶ ...
1  import { enableProdMode } from '@angular/core';
2  import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
3
4  import { AppModule } from './app/app.module';
5  import { environment } from './environments/environment';
6
7  if (environment.production) {
8    enableProdMode();
9  }
10
11 platformBrowserDynamic().bootstrapModule(AppModule)
12   .catch(err => console.error(err));
13
```

Pass "AppModule". "AppModule" refer to
"app.module.ts" file.

Check if production mode or not.

App.module.ts

- Copy C:\workspace-UT-Angular8-SEPT-2020 \session4\example1-bootstrap4-version2\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020 \session4\example1-bootstrap4-version3
- The app.module.ts(C:\workspace-UT-Angular8-SEPT-2020 \session4\example1-bootstrap4-version3\bootstrap4\src\app\app.module.ts) file

App.module.ts

```
src ▶ app ▶ TS app.module.ts ▶ ...
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6
7  @NgModule({
8    declarations: [
9      AppComponent
10   ],
11    imports: [
12      BrowserModule,
13      AppRoutingModule
14   ],
15    providers: [],
16    bootstrap: [AppComponent]
17 })
18  export class AppModule { }
```

Reference our 'AppComponent' in 'app.component.ts'.
Angular analyze 'AppComponent' and know there is a
selector called 'app-root' and now Angular able to
handle 'app-root' in the index.html file. Angular knows
to insert the html in 'app.component.html'



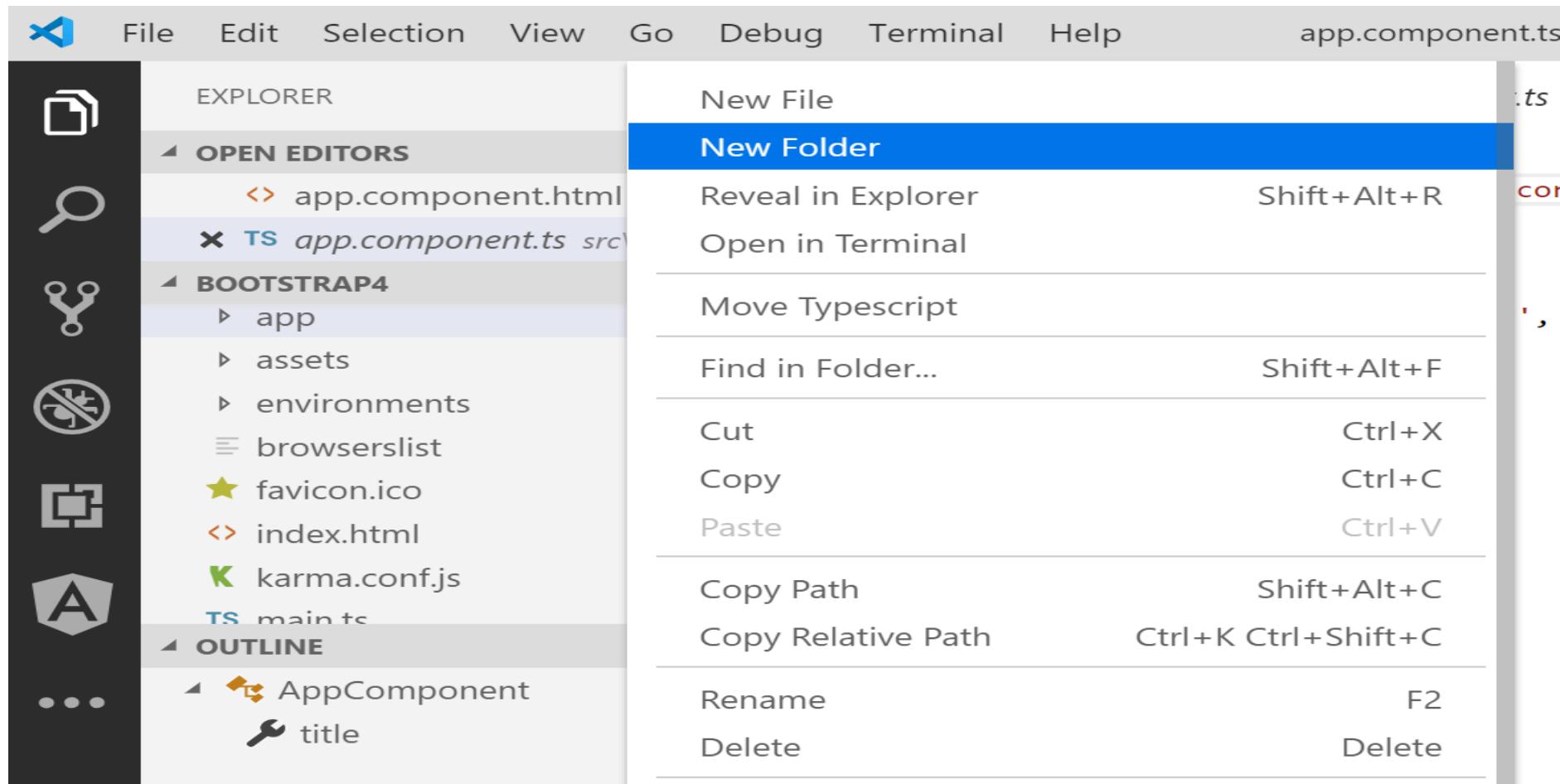
Components are important

- Components are a key feature in Angular. You build a whole application from a couple of components, which you created on your own. We started from “appComponent”, the root component, which holds our entire application in the end. So the root component will be the template where we will add our other components.

Creating a new component

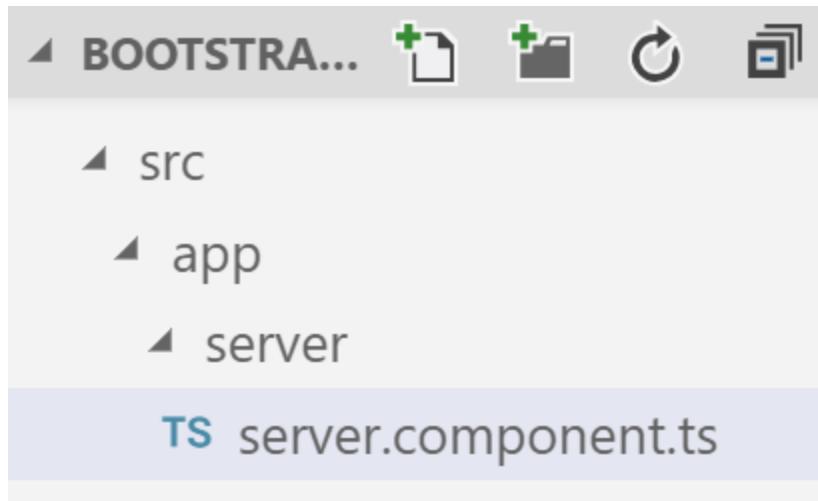
- All new components selectors will be added to the “app.component.html” file.
- Let’s say we want to output some information about a server. We’re building a back-end for our server management application and we want to output some server information.
- Create a new folder inside “app” folder called “server”.
- Right click “app” folder and choose “new folder”.

Creating a new component



Creating a new component

- Right click on “server” folder and choose new file. Name it “server.component.ts”.



Creating a new component

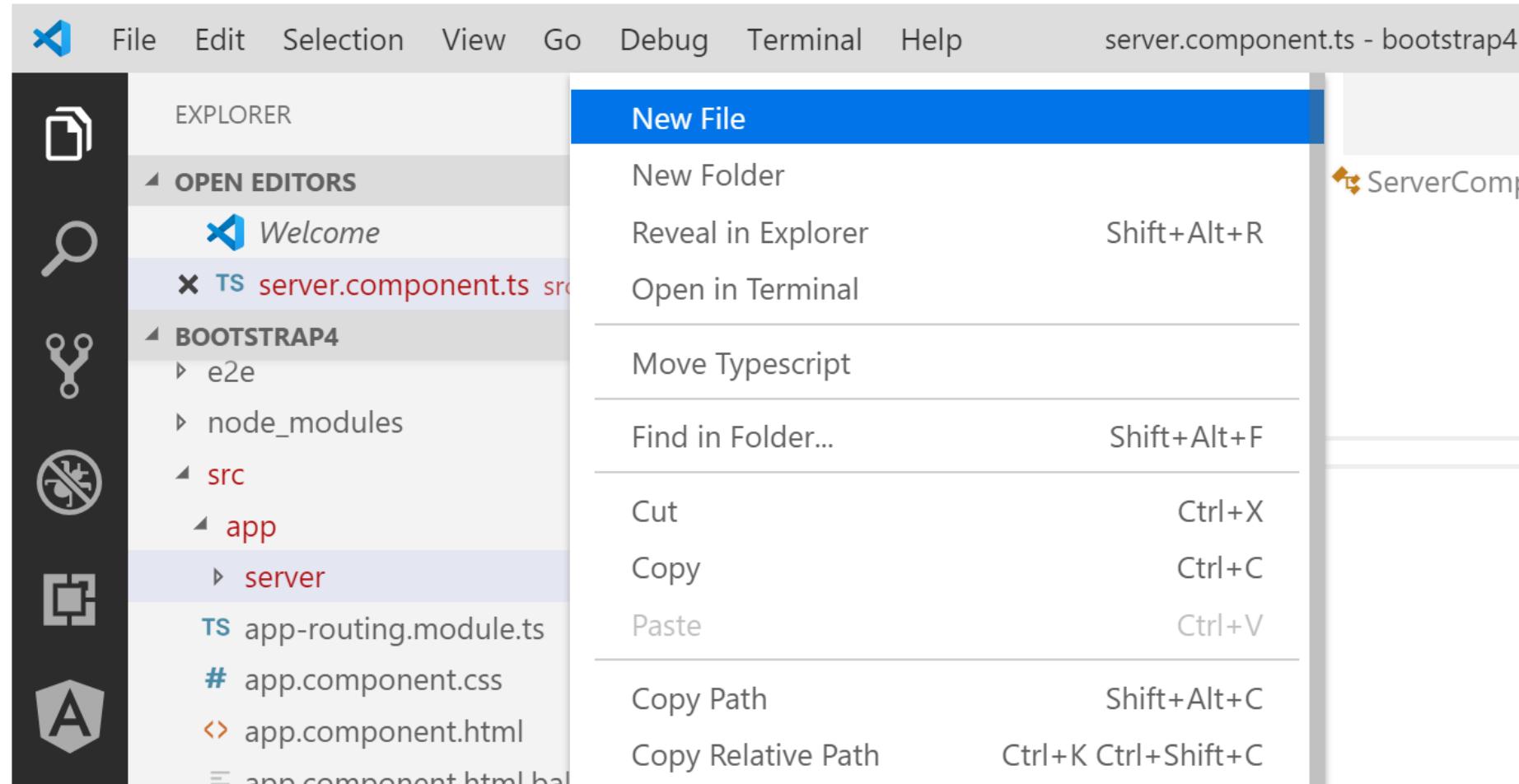
- A component is a typescript class.

Creating a new component

```
src ▶ app ▶ server ▶ TS server.component.ts ▶ ...
1   import { Component } from '@angular/core';
2
3   @Component({
4     selector: 'app-server',
5     templateUrl: './server.component.html'
6   })
7   export class ServerComponent {
8
9 }
10
11
12
```

Creating a new component

- Right click on server folder and choose new file.



Creating a new component

- Type “server.component.html” file.

TS app.module.ts

TS server.component.ts ●

↔ server.component.html

src > app > server > TS server.component.ts > ServerComponent

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-server',
5   templateUrl: './server.component.html'
6 })
7 )
8 export class ServerComponent {
9
10 }
11
```

Creating a new component

- Type the following in “server.component.html”.

The screenshot shows a code editor interface with three tabs:

- app.component.html
- server.component.ts
- server.component.html

The active tab is "server.component.html". The file structure is shown as:

```
src ▶ app ▶ server ▶ server.component.html ▶ p
```

The content of the file is:

```
1 <p>The Server Component</p>
```

Line 1 contains the opening tag `<p>` and the text `The Server Component`. Line 2 contains the closing tag `</p>`.

App.module.ts

- What is “app.module.ts”?
- Angular uses components to build web pages and uses modules to bungle different pieces. For example, components of your apps to packages. A module is basically a bungle of functionality of your app. It basically gives Angular information which features does my app have?
- Just creating the server.component.ts is not enough. Angular doesn’t know it exists. In order to let Angular know the server.component.ts exist, we have to register it in the app.module.ts, specifically in the “@NgModule”.

App.module.ts

- Original code:

```
src ▶ app ▶ TS app.module.ts ▶ ...
```

```
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6
7  @NgModule({
8    declarations: [
9      AppComponent
10   ],
11    imports: [
12      BrowserModule,
13      AppRoutingModule
14   ],
15    providers: [],
16    bootstrap: [AppComponent]
17 })
18 export class AppModule { }
```

App.module.ts

- new code:

```
src ▶ app ▶ TS app.module.ts ▶ AppModule
```

```
1 import { BrowserModule } from '@angular/platform-browser';
2 import { NgModule } from '@angular/core';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { ServerComponent } from './server/server.component';
7
8 @NgModule({
9   declarations: [
10     AppComponent,
11     ServerComponent
12   ],
13   imports: [
14     BrowserModule,
15     AppRoutingModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
20 export class AppModule { }
```

Add two new lines

Simply add other modules into this module.
Importing some modules into Angular

App.module.ts

- Run following:
- C:\workspace-UT-Angular8-SEPT-2020
 \session4\example1-bootstrap4-version3\bootstrap4>npm
 install @angular/http@latest or npm install
 @angular/http@7.2.15

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1- bootstrap4-version3\bootstrap4>npm install @angular/http@7.2.15
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ @angular/http@7.2.15
added 1 package from 1 contributor and audited 43281 packages in 44.04s
Found 1 moderate severity vulnerability
  run `npm audit fix` to fix them, or `npm audit` for details

C:\workspace-UT-Angular7-Sept-2019\Session4\example1- bootstrap4-version3\bootstrap4>
```

Creating a new component

src ▶ app ▶ **TS** app.module.ts ▶ ...

```
1 import { BrowserModule } from '@angular/platform-browser';
2 import { NgModule } from '@angular/core';
3
4 import { FormsModule } from '@angular/forms';
5 import { HttpClientModule } from '@angular/http';
6
7 import { AppRoutingModule } from './app-routing.module';
8 import { ServerComponent } from './server/server.component';
9 import { AppComponent } from './app.component';

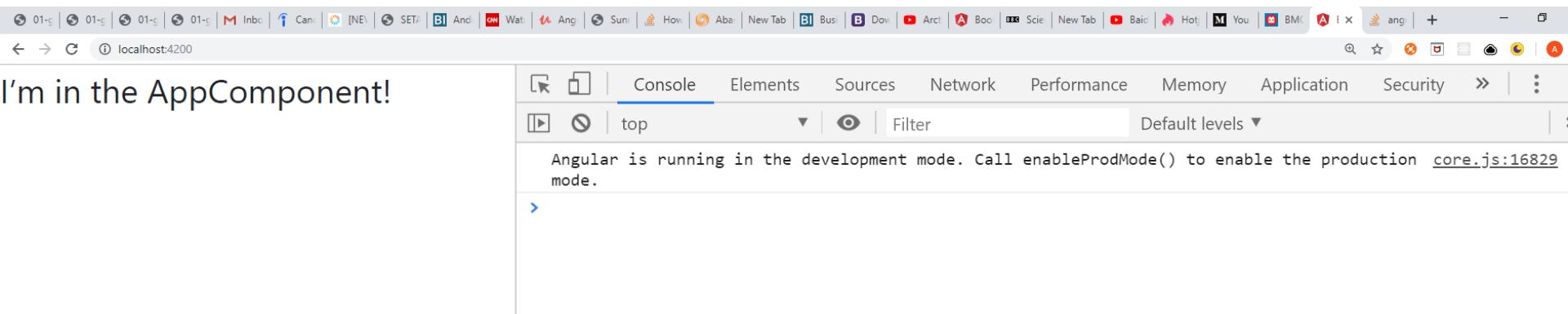
10
11
12
13 @NgModule({
14   declarations: [
15     AppComponent,
16     ServerComponent
17   ],
18   imports: [
19     BrowserModule,
20     FormsModule,
21     HttpClientModule, // This line is crossed out
22     AppRoutingModule
23   ],
24   providers: [],
25   bootstrap: [AppComponent]
26 })
27 export class AppModule { }
```

Add new lines of code



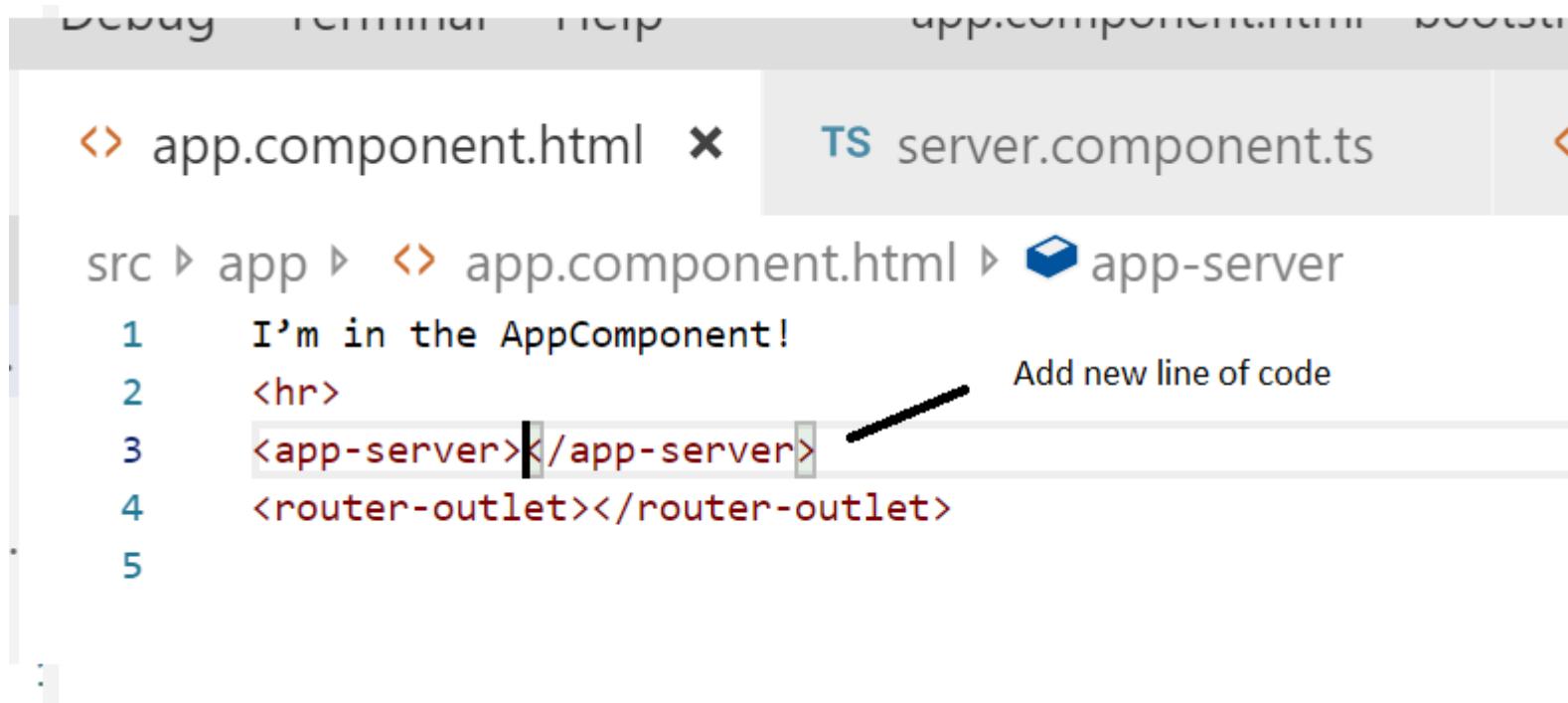
Creating a new component

- Make sure still working:



Using Custom Components

- Add new line of code in app.component.html:



The screenshot shows a code editor interface with a tab bar at the top. The active tab is "app.component.html". Below the tabs, there's a breadcrumb navigation bar showing the file structure: "src" > "app" > "app.component.html" > "app-server". The main content area contains the following code:

```
1 I'm in the AppComponent!
2 <hr>
3 <app-server></app-server>
4 <router-outlet></router-outlet>
5
```

A red arrow points from the text "Add new line of code" to the closing tag of the `<app-server>` element.

Using Custom Components

- Look at output in google chrome.

← → ⌂ ⓘ localhost:4200

I'm in the AppComponent!

The Server Component

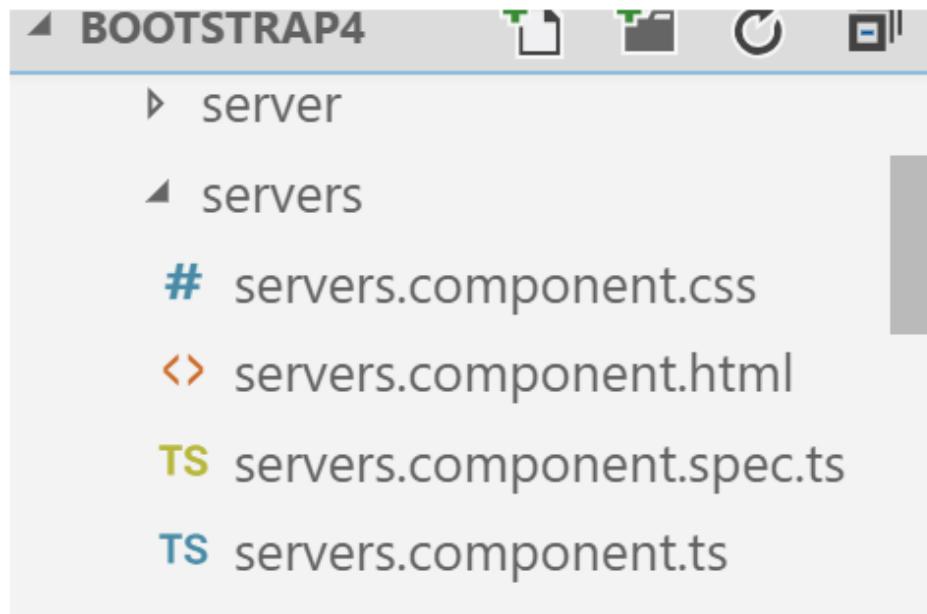
Generate Custom Components

- Copy C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version3\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version4
- Stop current CLI and change to directory C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version4\bootstrap4
- In dos-prompt type:ng generate component servers or ng g c servers

```
C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version4\bootstrap4>ng g c servers
CREATE src/app/servers/servers.component.html (26 bytes)
CREATE src/app/servers/servers.component.spec.ts (635 bytes)
CREATE src/app/servers/servers.component.ts (273 bytes)
CREATE src/app/servers/servers.component.css (0 bytes)
UPDATE src/app/app.module.ts (685 bytes)

C:\workspace-UT-Angular7-Sept-2019\Session4\example1-bootstrap4-version4\bootstrap4>
```

Generate Custom Components



Delete 'server.component.spec.ts' because it is
a testing file

Generate Custom Components

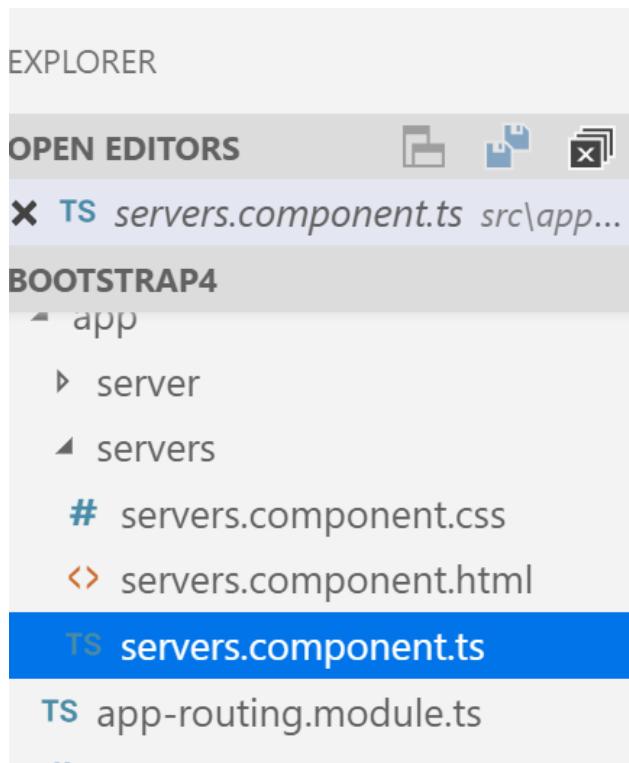
The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER** pane:
 - OPEN EDITORS**: app.module.ts (src\app) [2]
 - BOOTSTRAP4**:
 - app-routing.module.ts
 - app.component.css
 - app.component.html
 - app.component.spec.ts
 - app.component.ts
 - app.module.ts** (highlighted in blue) [2]
 - assets
 - environments
 - browserslist- OUTLINE** pane:
 - AppModule [2]
- Editor pane (app.module.ts)**:

```
src > app > TS app.module.ts > ...
4   import { FormsModule } from '@angular/forms';
5   import { HttpClientModule } from '@angular/http';
6
7   import { AppRoutingModule } from './app-routing.module';
8   import { AppComponent } from './app.component';
9   import { ServerComponent } from './server/server.component';
10  import { ServersComponent } from './servers/servers.component'
11
12
13
14 @NgModule({
15   declarations: [
16     AppComponent,
17     ServerComponent,
18     ServersComponent
19   ],
20   imports: [
21     BrowserModule,
22     FormsModule,
23     HttpClientModule,
24     AppRoutingModule
25   ],
26   providers: [],
27   bootstrap: [AppComponent]
```

A callout arrow points from the text "Add the above two lines automatically" to the line "import { HttpClientModule } from '@angular/http';".

Generate Custom Components



Use 'app-servers' in html file

TS servers.component.ts

```
src ▶ app ▶ servers ▶ TS servers.component.ts ▶ ...
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-servers',
5   templateUrl: './servers.component.html',
6   styleUrls: ['./servers.component.css']
7 })
8 export class ServersComponent implements OnInit {
9
10   constructor() { }
11
12   ngOnInit() {
13 }
```

Generate Custom Components

- In C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version4\bootstrap4\src\app\app.component.html use “app-servers” instead of “app-server”.

Delete "app-server" with 'app-servers'

```
<!-- app.component.html -->
<div>
  I'm in the AppComponent!
  <hr>
  <app-server></app-server>
</div>
```

```
<!-- servers.component.html -->
<div>
  I'm in the ServersComponent!
  <hr>
  <app-server></app-server>
</div>
```

app.component.html

```
src > app > app.component.html > app-server
```

```
1 I'm in the AppComponent!
2 <hr>
3 <app-server></app-server>
4 <router-outlet></router-outlet>
5
```

Generate Custom Components

- In servers.component.html:

```
< app.component.html > servers.component.html ×
```

```
src > app > servers > < servers.component.html > ...
```

```
1 <app-server></app-server>
2 <app-server></app-server>
3 |
```

Generate Custom Components

The screenshot shows a browser window at `localhost:4200`. The page content includes the text "I'm in the AppComponent!" and "Two 'app-server'". In the developer tools, the "Elements" tab is selected, displaying the DOM structure:

```
<!doctype html>
...<html lang="en"> == $0
  <head>...</head>
  <body>
    <app-root _ngcontent-xjj-c0 ng-version="7.2.15">
      "I'm in the AppComponent!
      "
      <hr _ngcontent-xjj-c0>
      <app-servers _ngcontent-xjj-c0 _ngcontent-xjj-c1>
        <app-server _ngcontent-xjj-c1>...</app-server>
        <app-server _ngcontent-xjj-c1>...</app-server>
      </app-servers>
      <router-outlet _ngcontent-xjj-c0></router-outlet>
    </app-root>
    <script type="text/javascript" src="runtime.js"></script>
    <script type="text/javascript" src="es2015-polyfills.js" nomodule>
      ...
    </script>
  </body>
</html>
```

Two arrows point from the text "Two 'app-server'" to the two nested `<app-server>` elements in the DOM tree.

Working with Component Templates

- Instead of using external template file, you can use inline template file.
- Go to servers.component.ts and change templateUrl to “template”.
- Copy C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version4\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version5\bootstrap4
- Change C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version5\bootstrap4\src\app\servers\servers.component.ts.

Working with Component Templates

Delete templateUrl to template

app.component.html servers.component.html TS servers.component.ts x

src ▶ app ▶ servers ▶ TS servers.component.ts ▶ ServersComponent

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-servers',
5   template: '<app-server></app-server><app-server></app-server>',
6   styleUrls: ['./servers.component.css']
7 })
8 export class ServersComponent implements OnInit {
9
10   constructor() { }
11
12   ngOnInit() {
13   }
14 }
```

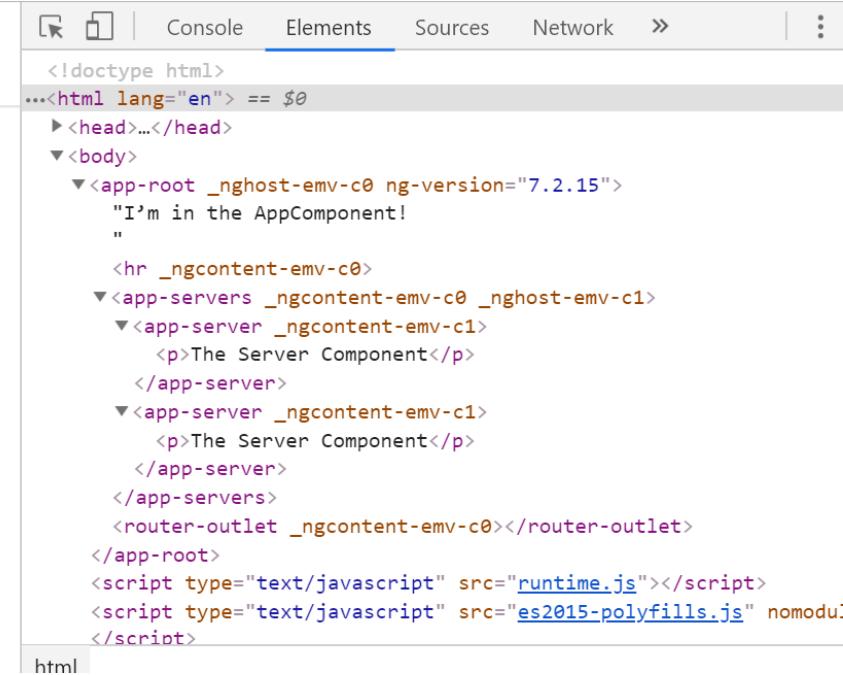
Working with Component Templates

- Check in browser still working:

I'm in the AppComponent!

The Server Component

The Server Component



The screenshot shows the browser's developer tools with the "Elements" tab selected. The DOM tree is displayed, starting with the <html> element. Inside the <body> element, there is an <app-root> element with attributes _ngcontent-emv-c0 and ng-version="7.2.15". This root component contains two <app-server> components, each with an _ngcontent-emv-c1 attribute. Each <app-server> component contains a <p> element with the text "The Server Component". Below the <app-root> is a <router-outlet> element with an _ngcontent-emv-c0 attribute. At the bottom of the <html> element, there are two <script> tags: one with src="runtime.js" and another with src="es2015-polyfills.js" and the nomodule attribute.

```
<!doctype html>
...<html lang="en"> == $0
  > <head>...</head>
  > <body>
    > <app-root _ngcontent-emv-c0 ng-version="7.2.15">
      "I'm in the AppComponent!
      "
      <hr _ngcontent-emv-c0>
      > <app-servers _ngcontent-emv-c0 _ngcontent-emv-c1>
        > <app-server _ngcontent-emv-c1>
          <p>The Server Component</p>
        </app-server>
        > <app-server _ngcontent-emv-c1>
          <p>The Server Component</p>
        </app-server>
      </app-servers>
      <router-outlet _ngcontent-emv-c0></router-outlet>
    </app-root>
    <script type="text/javascript" src="runtime.js"></script>
    <script type="text/javascript" src="es2015-polyfills.js" nomodule></script>
  </html>
```

Working with Component Styles

- Copy C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version4\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6
- Edit the following:
- C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6\bootstrap6\src\app\app.component.html

Working with Component Styles

app.component.html

src ▶ app ▶ app.component.html ▶ router-outlet

```
1   <div class="container">
2     <div class="row">
3       <div class="col-xs-12">
4         <h3>I'm in the AppComponent!</h3>
5         <hr>
6         <app-servers></app-servers>
7       </div>
8     </div>
9   </div>
10  <router-outlet></router-outlet>
11
```

Working with Component Styles

- Edit the following C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6\bootstrap4\src\app\app.component.css

```
app.component.html # app.component.css x
src ▶ app ▶ # app.component.css ▶ h3
1   h3 {
2     color: darkblue;
3   }
4
```

Working with Component Styles

- Edit the following C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6\bootstrap4\src\app\app.component.css

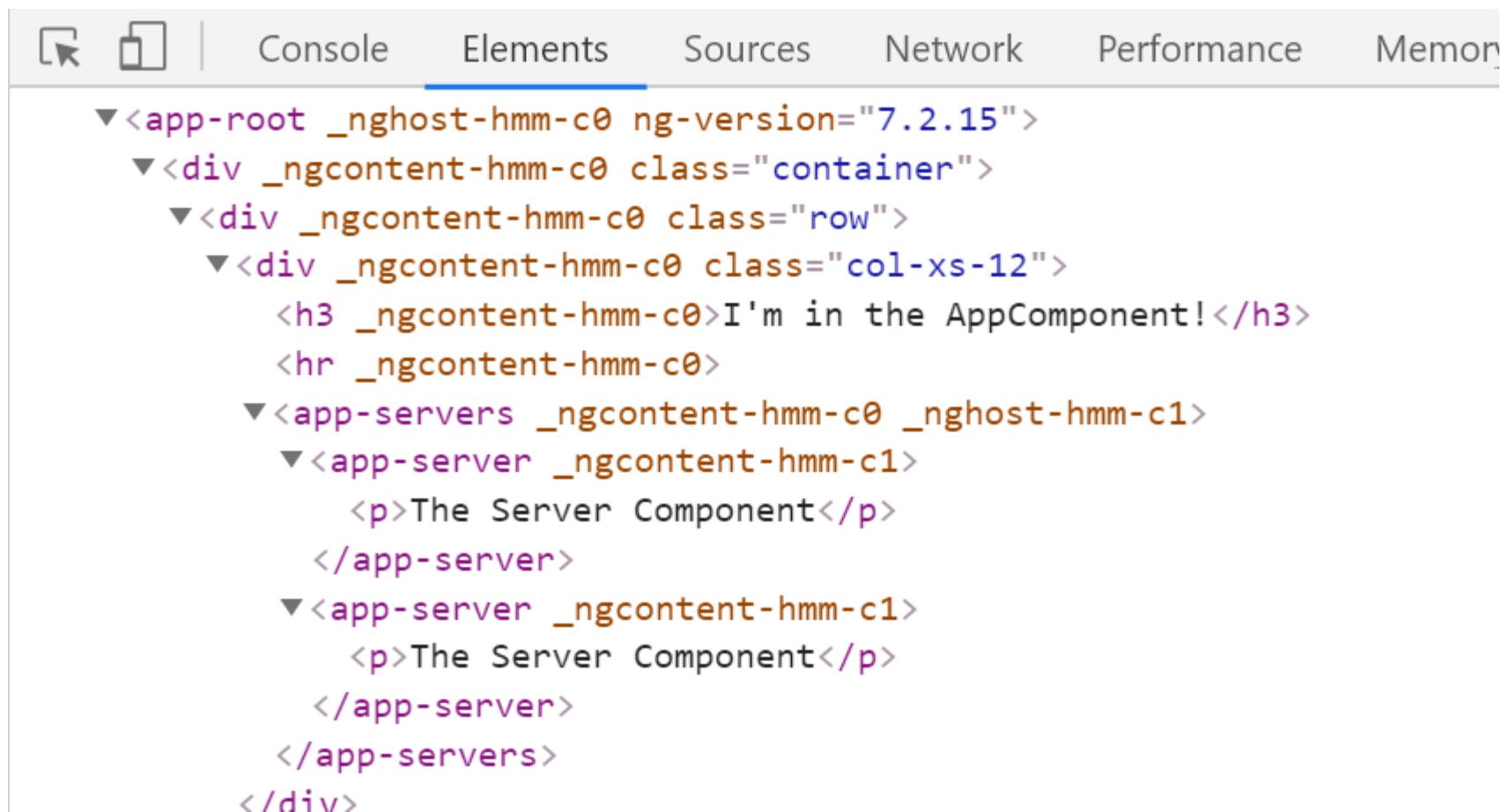
I'm in the AppComponent!

The Server Component

The Server Component

Working with Component Styles

- Edit the following C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6\bootstrap4\src\app\app.component.css

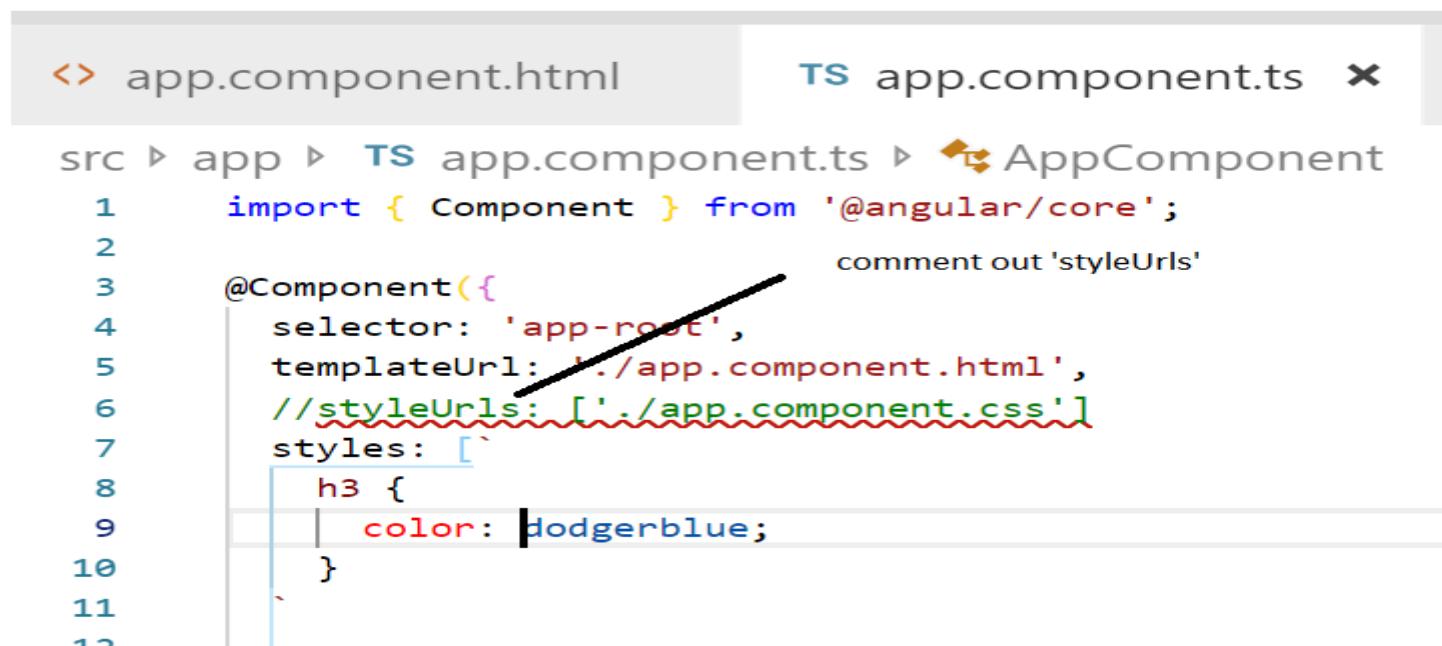


The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. The DOM tree displays the structure of the AppComponent. The root node is `<app-root _ngcontent-hmm-c0 ng-version="7.2.15">`. It contains a `<div _ngcontent-hmm-c0 class="container">`, which has a `<div _ngcontent-hmm-c0 class="row">` child. This row contains a `<div _ngcontent-hmm-c0 class="col-xs-12">` which includes an `<h3 _ngcontent-hmm-c0>I'm in the AppComponent!</h3>` and an `<hr _ngcontent-hmm-c0>`. Below this is another `<app-servers _ngcontent-hmm-c0 _ngcontent-hmm-c1>` node, which contains two `<app-server _ngcontent-hmm-c1>` nodes. Each server node contains a `<p>The Server Component</p>`.

```
<app-root _ngcontent-hmm-c0 ng-version="7.2.15">
  <div _ngcontent-hmm-c0 class="container">
    <div _ngcontent-hmm-c0 class="row">
      <div _ngcontent-hmm-c0 class="col-xs-12">
        <h3 _ngcontent-hmm-c0>I'm in the AppComponent!</h3>
        <hr _ngcontent-hmm-c0>
      </div>
    </div>
    <app-servers _ngcontent-hmm-c0 _ngcontent-hmm-c1>
      <app-server _ngcontent-hmm-c1>
        <p>The Server Component</p>
      </app-server>
      <app-server _ngcontent-hmm-c1>
        <p>The Server Component</p>
      </app-server>
    </app-servers>
  </div>
</app-root>
```

Working with Component Styles

- Edit C:\workspace-UT-Angular7-May-2019\session7\example1-bootstrap4-version6\bootstrap4\src\app\app.component.ts



```
<> app.component.html      TS app.component.ts ×  
src ▷ app ▷ TS app.component.ts ▷ AppComponent  
1 import { Component } from '@angular/core';  
2                                         comment out 'styleUrls'  
3 @Component({  
4   selector: 'app-root',  
5   templateUrl: './app.component.html',  
6   //styleUrls: ['./app.component.css']  
7   styles: [  
8     h3 {  
9       color: dodgerblue;  
10    }  
11  '  
12  ]  
13 ]  
14 })  
15 export class AppComponent {  
16   title = 'bootstrap4';
```



Working with Component Styles

- Edit C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6\bootstrap4\src\app\app.component.ts

The screenshot shows a browser window with multiple tabs. The active tab is 'localhost:4200' which displays the text 'I'm in the AppComponent!'. Below this, there are two sections: 'The Server Component' and 'The Server Component' under 'router-outlet'. The 'router-outlet' section has a size of '0 x 0'. To the right of the browser is the Chrome DevTools Elements panel. The 'Elements' tab is selected, showing the DOM structure of the page. A blue highlight is applied to the 'router-outlet' element in the DOM tree. The DOM content includes the following code:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="icon" type="image/x-icon" href="favicon.ico">
<style type="text/css">...</style>
...
<style type="text/css">...</style> == $0
<style>h3[_ngcontent-mab-c0] {
  color: dodgerblue;
}</style>
<style>...</style>
</head>
<body>
  <app-root _ngcontent-mab-c0 ng-version="7.2.15">
    <div _ngcontent-mab-c0 class="container">
      <div _ngcontent-mab-c0 class="row">...</div>
    </div>
    <router-outlet _ngcontent-mab-c0></router-outlet>
  </app-root>
  <script type="text/javascript" src="runtime.js"></script>

```

Fully Understanding The Component Selector

- Copy C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version6\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version7\bootstrap4
- Edit C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version7\bootstrap4\src\app\servers\servers.component.ts

Fully Understanding The Component Selector

TS servers.component.ts X <> app.component.html

src > app > servers > TS servers.component.ts > ServersComponent

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-servers',
5   selector: '[app-servers]',
6   templateUrl: './servers.component.html',
7   styleUrls: ['./servers.component.css']
8 })
9 export class ServersComponent implements OnInit {
10
11   constructor() { }
12
13   ngOnInit() {
14   }
15
16 }
17
```

Comment this out and add this line '[app-servers]'



Fully Understanding The Component Selector

- Edit C:\workspace-UT-Angular8-SEPT-2020\\session4\example1-bootstrap4-version7\bootstrap4\src\app\app.component.html

TS servers.component.ts <> app.component.html X

src > app > <> app.component.html > div.container > div

```
1  <div class="container">
2    <div class="row">
3      <div class="col-xs-12">
4        <h3>I'm in the AppComponent!</h3>
5        <hr>
6        <!--<app-servers></app-servers>-->
7        <div app-servers></div> add this line
8      </div>
9    </div>
10   </div>
11   <router-outlet></router-outlet>
```

Comment out this line

add this line

Fully Understanding The Component Selector

- Check browser:

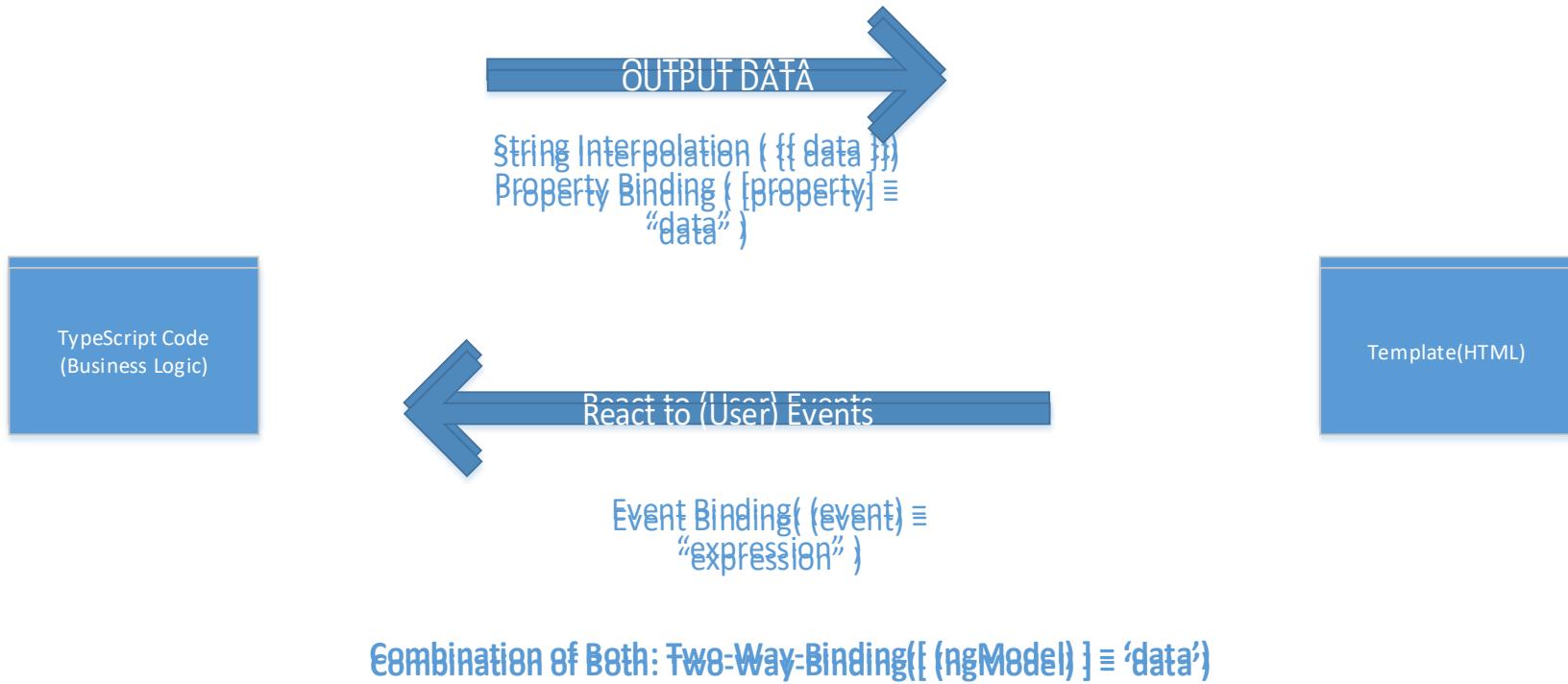
The screenshot shows a browser window with the URL `localhost:4200`. The main content area displays the text "I'm in the AppComponent!". Below this, there are two instances of the text "The Server Component". On the right side, the browser's developer tools are open, specifically the "Elements" tab. The DOM tree is visible, starting with the `<!doctype html>` tag. It includes an `<app-root>` component with attributes `_ngcontent-ajc-c0` and `ng-version="7.2.15"`. Inside this root component is a `<div>` element with class `"container"` and attribute `_ngcontent-ajc-c0`. This container holds a `<div>` element with class `"row"` and attribute `_ngcontent-ajc-c0`. Inside the row is a `<div>` element with class `"col-xs-12"` and attribute `_ngcontent-ajc-c0`, which contains an `<h3>` element with text "I'm in the AppComponent!". Below this is an `<hr>` element. Further down the tree, there are two `<app-server>` components. The first `<app-server>` has an attribute `_ngcontent-ajc-c1 == $0` and contains a `<p>` element with text "The Server Component". The second `<app-server>` has an attribute `_ngcontent-ajc-c1` and contains a `<p>` element with text "The Server Component". Both of these server components have a corresponding `</app-server>` tag below them.

What is databinding

- Copy C:\workspace-UT-Angular8-SEPT-2020 \session4\example1-bootstrap4-version4\bootstrap4 to C:\workspace-UT-Angular8-SEPT-2020 \session4\example1-bootstrap4-version8
- What is databinding?

What is databinding

Databinding ≡ Communication



What is databinding

- C:\ workspace-UT-Angular8-SEPT-2020
 \session4\example1-bootstrap4-
 version8\bootstrap4\src\app\server\server.component.ts

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

@Component({
  selector: 'app-server',
  templateUrl: './server.component.html'
})
export class ServerComponent {
  serverId: number = 10;
  serverStatus: string = 'offline';

  getServerStatus(): string {
    return this.serverStatus;
  }
}
```

What is databinding

- C:\workspace-UT-Angular8-SEPT-2020
 \session4\example1-bootstrap4-
 version8\bootstrap4\src\app\server\server.component.html

The screenshot shows a code editor interface with two tabs: 'server.component.html' and 'server.component.ts'. The 'server.component.html' tab is active, displaying the following code:

```
src ▶ app ▶ server ▶ <> server.component.html ▶ p
1  <p>{{'Server'}} with ID {{ serverId }} is {{ getServerStatus() }} </p>
2
```

The code uses Angular's double curly brace syntax for interpolation. The 'getServerStatus()' method call is highlighted with a light blue background.

What is databinding

- C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version8\bootstrap4\src\app\server\server.component.html

The screenshot shows a browser window with multiple tabs open at the top. The main content area displays the text "I'm in the AppComponent!" followed by two identical messages: "Server with ID 10 is offline". To the right, the browser's developer tools are open, specifically the Elements tab. The DOM tree shows the structure of the Angular application, including the AppComponent root element, nested app-servers, and app-server components.

```
<!doctype html>
<html lang="en">
  <head>...</head>
  ...<body> == $0
    <app-root _ngcontent-mwi-c0 ng-version="7.2.15">
      "I'm in the AppComponent!
      "
      <hr _ngcontent-mwi-c0>
      <app-servers _ngcontent-mwi-c0 _ngcontent-mwi-c1>
        <app-server _ngcontent-mwi-c1>
          <p>Server with ID 10 is offline </p>
        </app-server>
        <app-server _ngcontent-mwi-c1>
          <p>Server with ID 10 is offline </p>
        </app-server>
      </app-servers>
      <router-outlet _ngcontent-mwi-c0></router-outlet>
    </app-root>
    <script type="text/javascript" src="runtime.js"></script>
    <script type="text/javascript" src="es2015_polyfills.js" nomodule></script>
    <script type="text/javascript" src="polyfills.js"></script>
```

What is databinding

- C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version8\bootstrap4\src\app\server\server.component.html

The screenshot shows a browser window with multiple tabs open at the top. The main content area displays the Angular application's output. The browser's developer tools are open, specifically the Elements tab, which shows the DOM structure of the page.

The browser tabs include:

- A Boots! x
- Mortg... x
- Type 0 x
- Here's x
- Inbox x
- PLAN x
- Day in x
- Ivanka x
- South x
- Sweden x
- at 2011 a x
- B 2067 E x
- 2069 x
- 2013 E x
- new cc x
- top ch x
- T Tiffany x
- +

The main content area shows the following text:

```
I'm in the AppComponent!
Server with ID 10 is offline
Server with ID 10 is offline
```

The developer tools Elements tab shows the DOM structure:

```
<!doctype html>
<html lang="en">
  <head>...</head>
  ...<body> == $0
    <app-root _ngcontent-mwi-c0 ng-version="7.2.15">
      "I'm in the AppComponent!
      "
      <hr _ngcontent-mwi-c0>
      <app-servers _ngcontent-mwi-c0 _ngcontent-mwi-c1>
        <app-server _ngcontent-mwi-c1>
          <p>Server with ID 10 is offline </p>
        </app-server>
        <app-server _ngcontent-mwi-c1>
          <p>Server with ID 10 is offline </p>
        </app-server>
      </app-servers>
      <router-outlet _ngcontent-mwi-c0></router-outlet>
    </app-root>
    <script type="text/javascript" src="runtime.js"></script>
    <script type="text/javascript" src="es2015_polyfills.js" nomodule></script>
    <script type="text/javascript" src="polyfills.js"></script>
```

Property Binding

- Copy C:\workspace-UT-Angular8-SEPT-2020\session4\example1-bootstrap4-version8\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session4\example2-propertybinding-version1
- Edit C:\workspace-UT-Angular8-SEPT-2020\session4\example2-propertybinding-version1\bootstrap4\src\app\servers\servers.component.ts

Property Binding

src > app > servers > **TS** servers.component.ts > ServersComponent >

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-servers',
5   //template: '<app-server></app-server><app-server></app-server>',
6   templateUrl: './servers.component.html',
7   styleUrls: ['./servers.component.css']
8 })
9 export class ServersComponent implements OnInit {
10   allowNewServer = false;
11
12   constructor() {
13     setTimeout(() => {
14       this.allowNewServer = true;
15     }, 2000);
16   }
17
18   ngOnInit() {
19   }
20
21 }
```

servers.co

Add this new line of code and constructor

Property Binding

- C:\workspace-UT-Angular8-JAN-2020 \session4\example2-propertybinding-version1\bootstrap4\src\app\servers\servers.component.html

"[]" indicate to Angular we're using property binding.

We want to dynamically bind some property.



The screenshot shows a code editor with two tabs: 'servers.component.html' and 'servers.component.ts'. The 'servers.component.html' tab is active, displaying the following code:

```
src ▶ app ▶ servers ▶ <▶ servers.component.html ▶ ⚒ button>
1   <button class="btn btn-primary" [disabled]="!allowNewServer">Add Server</button>
2   <app-server></app-server>
3   <app-server></app-server>
4
```

A red arrow points from the explanatory text above to the '[disabled]' attribute in the first line of the HTML code.

Property Binding

- The button is enabled after 2 seconds:

```
▼<body>
  ▼<app-root _nghost-hqk-c0 ng-version="7.2.15">
    "I'm in the AppComponent!
    "
    <hr _ngcontent-hqk-c0>
    ▼<app-servers _ngcontent-hqk-c0 _nghost-hqk-c1>
      <button _ngcontent-hqk-c1 class="btn btn-primary" disabled>Add Server</button> ==
      ▶<app-server _ngcontent-hqk-c1>...</app-server>
      ▶<app-server _ngcontent-hqk-c1>...</app-server>
    </app-servers>
    <router-outlet _ngcontent-hqk-c0></router-outlet>
  </app-root>
```

word "disabled" added at the beginning of load and then disappear



Property Binding vs String Interpolation

- Copy C:\workspace-UT-Angular8-JAN-2020 \session4\example2-propertybinding-version1\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020 \session4\example3-propertybinding-vs-stringInterpolation
- Edit C:\workspace-UT-Angular8-JAN-2020 \session4\example3-propertybinding-vs-stringInterpolation\bootstrap4\src\app\servers\servers.component.html

Property Binding vs String Interpolation

- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example3-propertybinding-vs-stringInterpolation\bootstrap4\src\app\servers\servers.component.html

Add string interpolation {{ allowNewServer }}

src/app/servers/servers.component.html

src/app/servers/servers.component.html

```
1 <button class="btn btn-primary" [disabled]="!allowNewServer">Add Server</button>
2 <p>{{ allowNewServer }}</p>
3 <app-server></app-server>
4 <app-server></app-server>
5
6
```

Property Binding vs String Interpolation

- Edit C:\ workspace-UT-Angular8-JAN-2020\session4\example3-propertybinding-vs-stringInterpolation\bootstrap4\src\app\servers\servers.component.html

You could simply bind to a property of this element the innerText property and set this equal to allow a new server.

The innerText property of an element is just what's between the opening and closing tag. So in this case we were able to easily replace string interpolation with property binding.

server.component.html ✘

src ▶ app ▶ servers ▶ < server.component.html ▶ ...

```
1  <button class="btn btn-primary" [disabled]="!allowNewServer">Add Server</button>
2  <p>{{ allowNewServer }}</p>
3  <p [innerText]="allowNewServer"></p>
4  <app-server></app-server>
5  <app-server></app-server>
6
7
```

Property Binding vs String Interpolation

- Edit C:\workspace-UT-Angular8-JAN-2020\session7\example3-propertybinding-vs-stringInterpolation\bootstrap4\src\app\servers\servers.component.html

The screenshot shows a browser window with multiple tabs open. The active tab is 'Bootstrap4' at 'localhost:4200'. The page content displays the message 'I'm in the AppComponent!' and two 'false' values. A hand-drawn mark with a checkmark is placed over the first 'false' value, with the annotation 'False and False at the beginning and then turn to true' written next to it.

The developer tools are open, specifically the 'Elements' tab. The DOM tree shows the structure of the application. The 'app-root' element has the attribute 'ng-version="7.2.15"'. Inside it, there is an 'app-servers' component and a 'router-outlet' component. The 'app-servers' component contains several 'app-server' components and a 'button' element with the class 'btn btn-primary' set to 'disabled'. The 'button' also has a 'false' value assigned to its 'disabled' property via property binding.

```
<!doctype html>
<html lang="en">
  <head>...</head>
  <body>
    <app-root _ngcontent-cga-c0 ng-version="7.2.15">
      "I'm in the AppComponent!
      "
      <hr _ngcontent-cga-c0>
      <app-servers _ngcontent-cga-c0 _ngcontent-cga-c1>
        ...
          <button _ngcontent-cga-c1 class="btn btn-primary" disabled>Ad
            <p _ngcontent-cga-c1>false</p>
            <p _ngcontent-cga-c1>false</p>
          <app-server _ngcontent-cga-c1>...</app-server>
          <app-server _ngcontent-cga-c1>...</app-server>
        </app-servers>
        <router-outlet _ngcontent-cga-c0></router-outlet>
    </app-root>
```

Property Binding vs String Interpolation

- Edit C:\workspace-UT-Angular8-JAN-2020\session7\example3-propertybinding-vs-stringInterpolation\bootstrap4\src\app\servers\servers.component.html



true

true

Server with ID 10 is offline

Property Binding vs String Interpolation

- So when should you use which of the two?
- Well basically if you want to output something in your template print some text to it using string interpolation.
- If you want to change some property be that of an html element you will typically use property binding.

<> servers.component.html ✘

src ▶ app ▶ servers ▶ <> servers.component.html ▶ ...

```
1   <button class="btn btn-primary" [disabled]="!allowNewServer">Add Server</button>
2   <!--<p>{{ allowNewServer }}</p>-->
3   <p [innerText] ="allowNewServer"></p>
4   <app-server></app-server>
5   <app-server></app-server>
6
7
```

Property Binding vs String Interpolation

- So when should you use which of the two?
- Well basically if you want to output something in your template print some text to it using string interpolation.
- If you want to change some property be that of an html element you will typically use property binding.

The screenshot shows a browser window with the URL `localhost:4200`. The page content includes:

- A button labeled "Add Server".
- The text "true".
- The message "Server with ID 10 is offline".
- The message "Server with ID 10 is offline".

In the browser's developer tools, the "Elements" tab is selected. The DOM tree is displayed, showing the following structure:

```
<!doctype html>
<html lang="en">
  <head>...</head>
  <body>
    <app-root _ngcontent-lsl-c0 ng-version="7.2.15">
      "I'm in the AppComponent!
      "
      <hr _ngcontent-lsl-c0>
      <app-servers _ngcontent-lsl-c0 _ngcontent-lsl-c1>
        ...
        <button _ngcontent-lsl-c1 class="btn btn-primary">A
          <p _ngcontent-lsl-c1>true</p>
        <app-server _ngcontent-lsl-c1>...</app-server>
        <app-server _ngcontent-lsl-c1>...</app-server>
      </app-servers>
      <router-outlet _ngcontent-lsl-c0></router-outlet>
    </app-root>
```

Eventbinding

- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example3-propertybinding-vs-stringInterpolation\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session4\example4-eventbinding\bootstrap4\src\app\servers\servers.component.ts

Eventbinding

15 servers.component.ts ✘

src ▶ app ▶ servers ▶ **TS** servers.component.ts ▶ ServersComponent ▶

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-servers',
5   templateUrl: './servers.component.html',
6   styleUrls: ['./servers.component.css']
7 })
8 export class ServersComponent implements OnInit {
9   allowNewServer = false;
10  serverCreationStatus = 'No server was created!';
11  constructor() {
12    setTimeout(() => {
13      this.allowNewServer = true;
14    }, 2000);
15  }
16
17  ngOnInit() {
18  }
19
20  onCreateServer() {
21    this.serverCreationStatus = 'Server was created!';
22  }
23}
```

Add these line of code



Eventbinding

- Edit C:\workspace-UT-Angular8-JAN-2020\session7\example4-eventbinding\bootstrap4\src\app\servers\servers.component.html

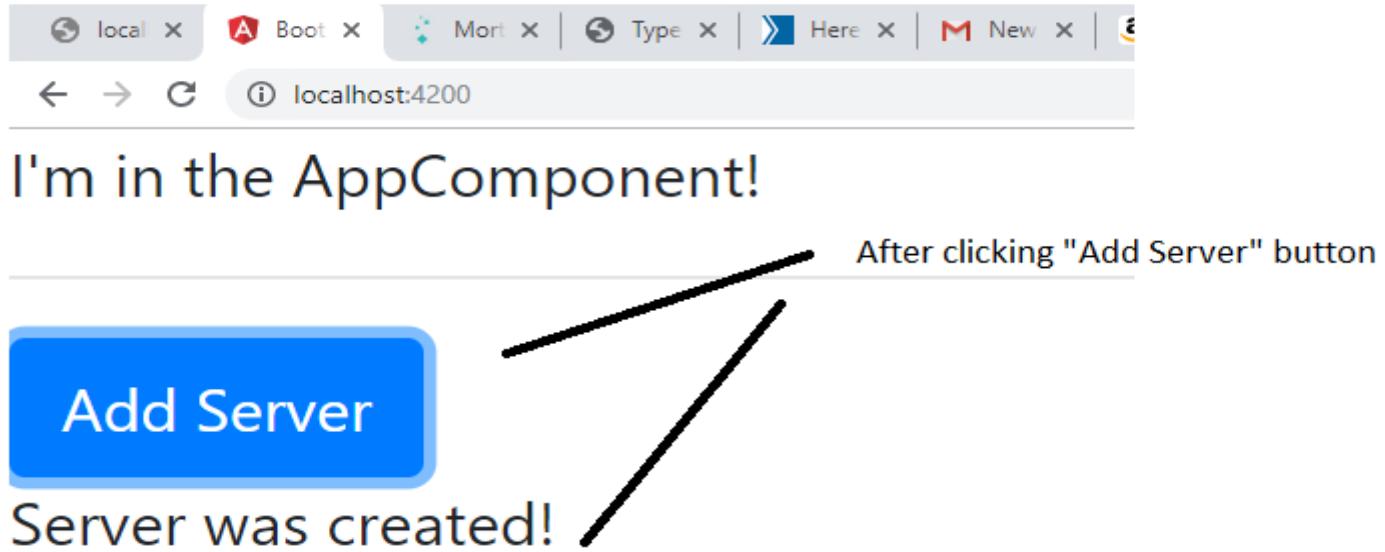
Event binding uses parentheses eg ().

Display value of variable
serverCreationStatus

TS servers.component.ts servers.component.html

```
src ▶ app ▶ servers ▶ servers.component.html ▶ p
1   <button class="btn btn-primary"
2     [disabled]="!allowNewServer"
3     (click)="onCreateServer()">Add Server</button>
4   <!--<p [innerText]="allowNewServer"></p>-->
5   <p>{{ serverCreationStatus }}</p>
6   <app-server></app-server>
7   <app-server></app-server>
8
9
```

Eventbinding



Server with ID 10 is offline

Server with ID 10 is offline

Passing and Using Data with Event Binding(example5-PassingAndUsingDataWithEventBinding)

- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example4-eventbinding\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session4\example5-PassingAndUsingDataWithEventBinding
- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example5-PassingAndUsingDataWithEventBinding\bootstrap4\src\app\servers\servers.component.html

Passing and Using Data with Event Binding(example5-PassingAndUsingDataWithEventBinding)

input and click are default events provided by the dom. Input event gives us some data about the event. We can capture this data if \$event passed as an argument to the method we're calling.

src app servers <> servers.component.html x

```
1  <label>Server Name</label>
2  <input type="text" class="form-control" (input)="onUpdateServerName($event)">
3  <button class="btn btn-primary"
4  [disabled]="!allowNewServer"
5  (click)="onCreateServer()">Add Server</button>
6  <!--<p [innerText]="allowNewServer"></p>-->
7  <p>{{ serverCreationStatus }}</p>
8  <app-server></app-server>
9  <app-server></app-server>
10
```

Passing and Using Data with Event

Binding(example5-

PassingAndUsingDataWithEventBinding)

- Edit C:\workspace-UT-Angular8-JAN-2020
 \session7\example5-
 PassingAndUsingDataWithEventBinding\bootstrap4\src\app
 \servers\servers.component.ts

```
onUpdateServerName(event: any) {  
  console.log(event);  
}
```

Passing and Using Data with Event Binding

The screenshot shows a web application interface and its corresponding developer tools.

Left Panel (Application View):

- I'm in the AppComponent!
- Server Name: (A red box highlights the input field)
- Add Server
- No server was created!
- Server with ID 10 is offline
- Server with ID 10 is offline

A black arrow points from the text "event.target.value is t" to the input field.

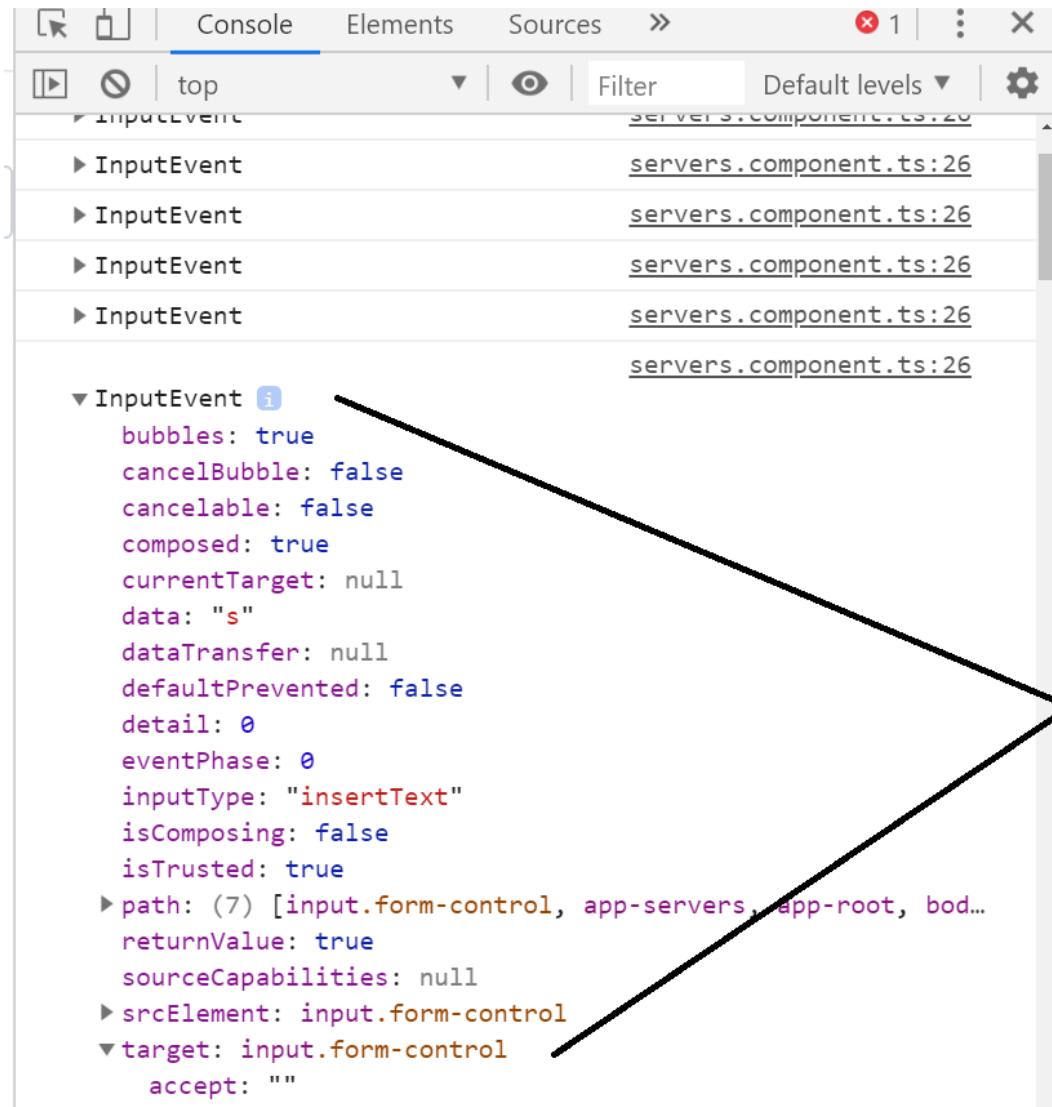
Right Panel (Developer Tools - Console Tab):

```
Object {onvolumechange: (...), onwaiting: (...), onwebkitfullscreenchange: (...), onwebkitfullscreenerror: (...), onwheel: (...), outerHTML: "<input _ngcontent-cgi-c1="" class="form-control" type="text" value="t">", outerText: "", ownerDocument: document, parentElement: app-servers, parentNode: app-servers, part: DOMTokenList [value:""], pattern: "", placeholder: "", prefix: null, previousElementSibling: label, previousSibling: label, readOnly: false, required: false, scrollHeight: 36, scrollLeft: 0, scrollTop: 0, scrollWidth: 372, selectionDirection: "forward", selectionEnd: 1, selectionStart: 1, shadowRoot: null, size: 20, slot: "", spellcheck: true, src: "", step: "", style: CSSStyleDeclaration {alignContent: "", alignItems: "", tabIndex: 0}, tagName: "INPUT", textContent: "", title: "", translate: true, type: "text", useMap: "", validationMessage: "", validity: ValidityState {valueMissing: false, typeMismatch: ""}, value: "t", valueAsDate: null, valueAsNumber: NaN}
```

Passing and Using Data with Event Binding

```
src:  
step: ""  
type: "text"  
defaultValue: ""  
value: "1"  
valueAsDate: null  
valueAsNumber: NaN  
width: 0  
willValidate: true  
► validity: ValidityState {valueMissing: false, typeMismatch: false, patternMismatch: false, timezoneMismatch: false, length: 1, lengthError: true, rangeError: false, stepMismatch: false, tooLong: false, tooShort: false, invalidCharacter: false, invalidEmail: false, invalidURL: false, invalidTel: false, invalidNumber: false, invalidRange: false}  
validationMessage: ""  
► labels: NodeList []  
selectionStart: 1  
selectionEnd: 1  
selectionDirection: "forward"  
align: ""  
useMap: ""  
webkitdirectory: false  
incremental: false  
► webkitEntries: []  
title: ""  
lang: ""  
translate: true  
...  
value|
```

Passing and Using Data with Event Binding



The screenshot shows the Chrome DevTools Console tab with the following output:

```
▶ InputEvent
▶ InputEvent
▶ InputEvent
▶ InputEvent
▼ InputEvent ⓘ
  bubbles: true
  cancelBubble: false
  cancelable: false
  composed: true
  currentTarget: null
  data: "s"
  dataTransfer: null
  defaultPrevented: false
  detail: 0
  eventPhase: 0
  inputType: "insertText"
  isComposing: false
  isTrusted: true
  ▶ path: (7) [input.form-control, app-servers, app-root, bod...
  returnValue: true
  sourceCapabilities: null
  ▶ srcElement: input.form-control
  ▶ target: input.form-control
    accept: ""
```

A callout arrow points from the text "event.target.value" to the "target" property in the expanded event object.

Passing and Using Data with Event Binding

The screenshot shows a web browser window with the URL `localhost:4200`. On the left, the application displays a form with a text input containing "Albert Lam", a button labeled "Add Server", and two success messages: "Server was created!" and "Server with ID 10 is offline". A tooltip below the input field says "Enter 'Albert Lam' for input and value 'Albert Lam'". On the right, the browser's developer tools are open, specifically the Console tab. The console output shows 11 messages, all of which are info-level logs. One log entry is expanded, showing properties like `tagName: "INPUT"`, `textContent: ""`, and `value: "Albert Lam"`. Other properties listed include `tabIndex: 0`, `translate: true`, `type: "text"`, and `width: 0`.

```
tabIndex: 0
tagName: "INPUT"
textContent: ""
title: ""
translate: true
type: "text"
useMap: ""
validationMessage: ""
validity: ValidityState {valueMissing: false, typeMismatch: false...}
value: "Albert Lam"
valueAsDate: null
valueAsNumber: NaN
webkitEntries: []
webkitdirectory: false
width: 0
willValidate: true
__zone_symbol__inputfalse: [ZoneTask]
```

example6-PassingAndUsingDataWithEventBinding

- Copy C:\workspace-UT-Angular8-JAN-2020 \session4\example5-PassingAndUsingDataWithEventBinding\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020 \session4\example6-PassingAndUsingDataWithEventBinding
- Edit C:\ workspace-UT-Angular8-JAN-2020 \session4\example6-PassingAndUsingDataWithEventBinding\bootstrap4\src\app\servers\servers.component.ts

example6-PassingAndUsingDataWithEventBinding

TS servers.component.ts x servers.component.html

src ▷ app ▷ servers ▷ TS servers.component.ts ▷ ServersComponent ▷

```
7  })
8  export class ServersComponent implements OnInit {
9    allowNewServer = false;
10   serverCreationStatus = 'No server was created!';
11   serverName = '';
12   constructor() {
13     setTimeout(() => {
14       this.allowNewServer = true;
15     }, 2000);
16   }
17
18   ngOnInit() {
19   }
20
21   onCreateServer() {
22     this.serverCreationStatus = 'Server was created!';
23   }
24   /*
25   onUpdateServerName(event: any) {
26     console.log(event);
27   }
28   */
29   onUpdateServerName(event: Event) {
30     this.serverName = (<HTMLInputElement>event.target).value;
31   }
32 }
```

Add new variable "serverName" and method "onUpdateServerName".



example6-PassingAndUsingDataWithEventBinding

- Edit C:\ workspace-UT-Angular8-JAN-2020\session4\example6-PassingAndUsingDataWithEventBinding\bootstrap4\src\app\servers\servers.component.html

add "servername" below input



```
TS servers.component.ts                                servers.component.html ×
```

src ▷ app ▷ servers ▷ servers.component.html ▷ p

```
1  <label>Server Name</label>
2  <input type="text" class="form-control" (input)="onUpdateServer"
3  <p>{{ serverName }}</p>
4  <button class="btn btn-primary"
5  [disabled]={!allowNewServer"
6  (click)="onCreateServer()">Add Server</button>
7  <!--<p [innerText]=""allowNewServer""></p>-->
8  <p>{{ serverCreationStatus }}</p>
9  <app-server></app-server>
10 <app-server></app-server>
11
12
```

example6-PassingAndUsingDataWithEventBinding

The screenshot shows a web browser window with the URL `localhost:4200`. The page content is as follows:

I'm in the AppComponent!

Server Name Type "Albert Lam" at the input box and Albert Lam shows up below the input box

Albert Lam

Albert Lam

Add Server

Server was created!

Server with ID 10 is offline

Server with ID 10 is offline

In the developer tools, the Console tab is selected. The console output is:

- 2 messages
- 1 user mess...
- 1 error
- No warnings
- 1 info
- No verbose

The error message is:

Uncaught SyntaxError: Unexpected token 'export'
Angular is running in the development mode. Call enableProdMode() to enable the production mode.

A hand-drawn arrow points from the text "Type 'Albert Lam' at the input box and Albert Lam shows up below the input box" to the input field.

example7-TwoWayDatabinding

- Copy C:\workspace-UT-Angular8-JAN-2020 \session7\example6-PassingAndUsingDataWithEventBinding\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020 \session4\example7-TwoWayDatabinding
- With two way databinding we combine property and event binding.
- Edit C:\workspace-UT-Angular8-JAN-2020 \session4\example7-TwoWayDatabinding\bootstrap4\src\app\servers\servers.component.html

example7-TwoWayDatabinding

src > app > servers > servers.component.html ...

```
1  <label>Server Name</label>          Add this line of code
2
3  <!--<input type="text" class="form-control" (input)="onUpdateServerName($event)">-->
4  <input type="text" class="form-control" [(ngModel)]="serverName">
5  <p>{{ serverName }} </p>
6  <button class="btn btn-primary"
7    [disabled]="!allowNewServer"
8    (click)="onCreateServer()">Add Server</button>
9  <!--<p [innerText]="allowNewServer"></p>-->
10 <p>{{ serverCreationStatus }}</p>
11 <app-server></app-server>
12 <app-server></app-server>
13
14
```

example7-TwoWayDatabinding

src ▶ app ▶ servers ▶ **TS** servers.component.ts ▶ ServersComponent

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-servers',
5   templateUrl: './servers.component.html',
6   styleUrls: ['./servers.component.css']
7 })
8 export class ServersComponent implements OnInit {
9   allowNewServer = false;
10  serverCreationStatus = 'No server was created!';
11  serverName = 'Testserver'; add word "Testserver"
12  constructor() {
13    setTimeout(() => {
14      this.allowNewServer = true;
15    }, 2000);
16 }
```

example7-TwoWayDatabinding

I'm in the AppComponent!

Server Name

Testserver

Testserver

Add Server

No server was created!

Default value "Testserver"

Console

Elements Sources Network Performance Me...

top

2 messages

1 user mess...

1 error

No warnings

1 info

No verbose

✖ Uncaught SyntaxError: Unexpected token 'export'
Angular is running in the development mode. Call enableProdMode() to enable the production mode.

example7-TwoWayDatabinding

I'm in the AppComponent!

Type te word "Albert Lam"

Server Name

Albert Lam

Albert Lam

Add Server

No server was created!

Server with ID 10 is offline

Server with ID 10 is offline

localhost:4200

Console

Elements

Sources

Network

Performance

Memory

top

2 messages

1 user mess...

1 error

No warnings

1 info

No verbose

Uncaught SyntaxError: Unexpected token 'export'
Angular is running in the development mode. Call enableProdMode() to enable the production mode.

example8-CombiningAllFormsOfDataBinding

- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example7-TwoWayDatabinding\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session7\example8-CombiningAllFormsOfDataBinding
- Edit servers.component.html

Comment out "serverName"

.servers.component.html x

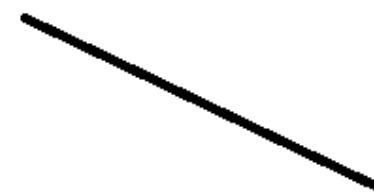
src ▶ app ▶ servers ▶ servers.component.html ▶ ...

```
1  <label>Server Name</label>
2
3  <!--<input type="text" class="form-control" (input)="onUpdateServerName($event)">-->
4  <input type="text" class="form-control" [(ngModel)]="serverName">
5  <!--<p>{{ serverName }} </p>-->
6  <button class="btn btn-primary"
7  [disabled]="!allowNewServer"
8  (click)="onCreateServer()">Add Server</button>
```

example8-CombiningAllFormsOfDataBinding

- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example8-CombiningAllFormsOfDataBinding\bootstrap4\src\app\servers\servers.component.ts

Add "this.serverName;"



```
20
21  onCreateServer() {
22      this.serverCreationStatus = 'Server was created! Name is ' + this.serverName;
23  }
```

example8-CombiningAllFormsOfDataBinding

I'm in the AppComponent!

type "Testserver 2 and click "Add Server"

Server Name

Testserver 2

Add Server

Server was created! Name is Testserver 2

Server with ID 10 is offline

Server with ID 10 is offline

4



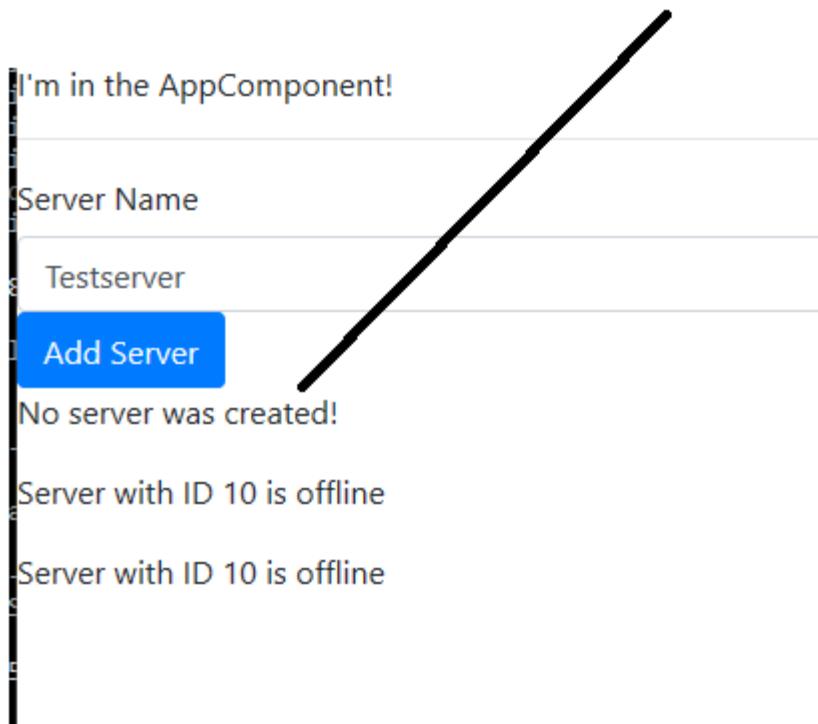
UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Directives(example9-UsingNglfToOutputData)

- What are Directives?
- Directives are Instructions in the DOM!
- Components are directives but directives with a template.
- *ngIf is a structural directives which add or remove elements.
- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example8-CombiningAllFormsOfDataBinding\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session4\example9-UsingNglfToOutputData

Directives(example9-UsingNgIfToOutputData)

Currently "No server was created!" display at the beginning. We need to conditionally show this message "No server was created!".



Directives(example9-UsingNglfToOutputData)

- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example9-UsingNglfToOutputData\bootstrap4\src\app\servers\servers.component.ts

```
new variable  
"serverCreated=false"  
  
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;  
  }  
}
```

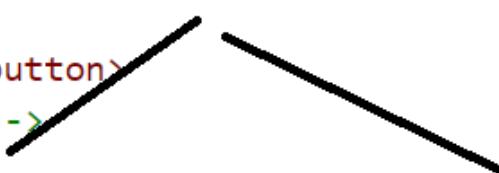
Add this line of code.

Directives(example9-UsingNgIfToOutputData)

- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example9-UsingNgIfToOutputData\bootstrap4\src\app\servers\servers.component.html

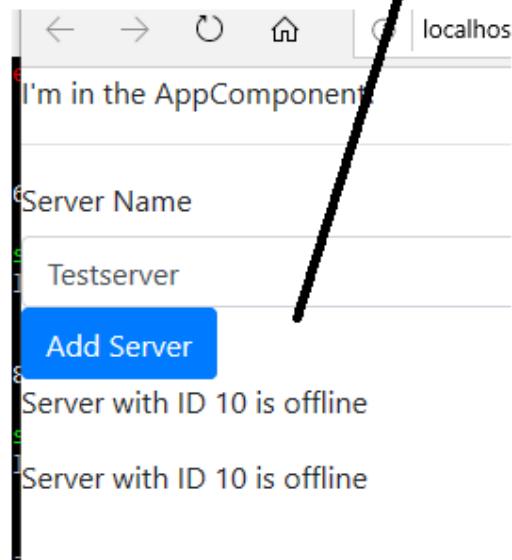
src ▶ app ▶ servers ▶ servers.component.html ▶ p

```
1  <label>Server Name</label>
2
3  <!--<input type="text" class="form-control" (input)="onUpdateServerName($event)">-->
4  <input type="text" class="form-control" [(ngModel)]="serverName">
5  <!--<p>{{ serverName }} </p>-->
6  <button class="btn btn-primary"          Add this new line
7  | [disabled]="!allowNewServer"
8  | (click)="onCreateServer()">Add Server</button>
9  <!--<p>[innerText]={{ allowNewServer }}</p>-->
10 <!--<p>{{ serverCreationStatus }}</p>-->
11 <p *ngIf="serverCreated">Server was created, server name is {{ serverName }}</p>
12 <app-server></app-server>
13 <app-server></app-server>
14
```



Directives(example9-UsingNgIfToOutputData)

line "Server was created, server name is Testserver"
doesn't show.



I'm in the AppComponent!

The line "Server was created.." show only after clicking "Add Server" button.

Server Name

Testserver

Add Server

Server was created, server name is Testserver

Server with ID 10 is offline

Server with ID 10 is offline



example10-styligElementsDynamicallyWithNgStyle

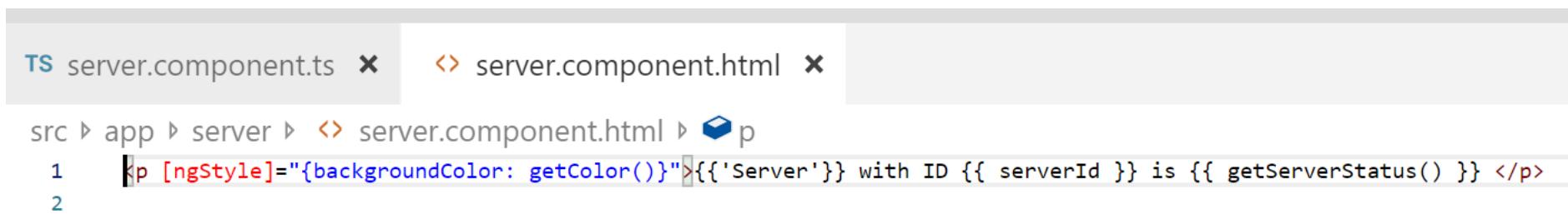
- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example9-UsingNgIfToOutputData\bootstrap4 to C:\workspace-UT-Angular7-Sept-2019\session4\example10-styligElementsDynamicallyWithNgStyle
- Unlike structural directives, attribute directives don't add or remove elements. They only change the element they were placed on.
- NgStyle allows us to dynamically assign a style. NgStyle allows us to change the css style itself.
- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example10-styligElementsDynamicallyWithNgStyle\bootstrap4\src\app\server\server.component.ts

example10-stylingElementsDynamicallyWithNgStyle

```
server.component.ts  server.component.html
src ▶ app ▶ server ▶ TS server.component.ts ▶ ServerComponent ▶ getServerStatus()
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-server',
5   templateUrl: './server.component.html'
6 })
7 export class ServerComponent {
8   serverId: number = 10;
9   serverStatus: string = 'offline';
10
11 constructor() {
12   this.serverStatus = Math.random() > 0.5 ? 'online' : 'offline';
13 }
14 getServerStatus(): string {
15   return this.serverStatus;
16 }
17
18 getColor() {
19   return this.serverStatus === 'online' ? 'green' : 'red';
20 }
21 }
```

example10-stylingElementsDynamicallyWithNgStyle

- Edit C:\workspace-UT-Angular8-JAN-2020\Session4\example10-stylingElementsDynamicallyWithNgStyle\bootstrap4\src\app\server\server.component.html



```
TS server.component.ts ✘  < > server.component.html ✘  
src ▶ app ▶ server ▶ < > server.component.html ▶ p  
1   <p [ngStyle]="{backgroundColor: getColor()}">{{'Server'}} with ID {{ serverId }} is {{ getServerStatus() }} </p>  
2
```

example10-stylingElementsDynamicallyWithNgStyle

I'm in the AppComponent!

Server Name

Testserver

Add Server

Server with ID 10 is offline

Server with ID 10 is online

example11-NgClass

- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example10-styligElementsDynamicallyWithNgStyle\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session4\example11-NgClass
- NgClass allows us to dynamically add or remove CSS classes.
- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example11-NgClass\bootstrap4\src\app\server\server.component.ts

example11-NgClass

Add this line

TS server.component.ts 

src ▶ app ▶ server ▶ TS server.component.ts ▶  ServerComponent

```
1 import { Component } From '@angular/core';
2
3 @Component({
4   selector: 'app-server',
5   templateUrl: './server.component.html',
6   styles: [
7     .online {
8       color: white;
9     }
10    ]
11 })
```

example11-NgClass

- C:\workspace-UT-Angular7-Sept-2019\session4\example11-NgClass\bootstrap4\src\app\server\server.component.html

The diagram illustrates a code flow from the server component's TypeScript file to its corresponding HTML template. A large downward-pointing arrow originates from the top right and points to the 'server.component.ts' tab. Another arrow points from the 'server.component.ts' tab to the 'server.component.html' tab. The 'server.component.html' tab is currently active, showing the following code:

```
src ▶ app ▶ server ▶ <server.component.html> p
1  <p [ngStyle]="{backgroundColor: getColor()}" [ngClass]="{online: serverStatus === 'online'}">
2    {{'Server'}} with ID {{ serverId }} is {{ getServerStatus() }} </p>
3
```

The code uses Angular's binding syntax to set the background color and apply a class based on the server status.

example11-NgClass

The screenshot shows a browser window with two tabs: "Bootstrap4" and "Deutsche Bank is being slammed". The active tab displays a simple Angular application. The page contains the following elements:

- A header with the text "'m in the AppComponent!'".
- An input field labeled "Server Name" containing the value "Testserver".
- A blue button labeled "Add Server".
- A red row containing the text "Server with ID 10 is offline".
- A green row containing the text "Server with ID 10 is online".

Below the green row, there is a note: "class='online' being added for online row. The characters are white."

The browser's developer tools are open, specifically the "Elements" tab. It shows the HTML structure of the application. The red row corresponds to the element `<p _ngcontent-qdp-c2 class="online" ...> Server with ID 10 is offline </p>`. The green row corresponds to the element `<p _ngcontent-qdp-c2 class="online" ng-reflect-ng-class="[object Object]" ng-reflect-ng-style="[object Object]" style="background-color: green;"> Server with ID 10 is online </p>`. The developer tools also show other parts of the application, including a button "Add Server" and some nested components.

example12-NgFor

- Copy C:\workspace-UT-Angular8-JAN-2020\session4\example11-NgClass\bootstrap4 to C:\workspace-UT-Angular8-JAN-2020\session4\example12-NgFor
- Edit C:\workspace-UT-Angular8-JAN-2020\session4\example12-NgFor\bootstrap4\src\app\servers\servers.component.ts

example12-NgFor

```
export class ServersComponent implements OnInit {
  allowNewServer = false;
  serverCreationStatus = 'No server was created!';
  serverName = 'Testserver';
  serverCreated = false;
  servers = ['Testserver', 'Testserver 2'];

  constructor() {
    setTimeout(() => {
      this.allowNewServer = true;
    }, 2000);
  }

  ngOnInit() {}

  onCreateServer() {
    this.serverCreated = true;
    this.servers.push(this.serverName);
    this.serverCreationStatus = 'Server was created! Name: ' + this.serverName;
  }
}
```

Add new array "servers" and push new object to new array 'servers' when onCreateServer event is trigger.



example12-NgFor

Updated this line of code with "*ngFor".

TS servers.component.ts ×

✓ servers.component.html ×

src ▶ app ▶ servers ▶ servers.component.html ▶ ...

```
1  <label>Server Name</label>
2  <!--<input type="text" class="form-control" (input)="onUpdateServerName($event)">-->
3  <input type="text" class="form-control" [(ngModel)]="serverName">
4  <button class="btn btn-primary" [disabled]="!allowNewServer" (click)="onCreateServer()">Add Server</button>
5  <!--<p>{{ serverName }}</p>-->
6  <!--<p>{{ allowNewServer }}</p>-->
7  <!--<p> [innerText]={{ allowNewServer }}</p>-->
8  <!--<p> {{ serverCreationStatus }}</p>-->
9  <p *ngIf="serverCreated">Server was created, server name is {{ serverName }}</p>
10 <app-server *ngFor="let server of servers"></app-server>
11 <!--<app-server></app-server>--> Comment out this line of code
12
```



example12-NgFor

localhost:4200

I'm in the AppComponent!

Server Name

Testserver3

Add Server

Server was created, server name is Testserver3

Server with ID 10 is offline

Server with ID 10 is online

Server with ID 10 is offline

Server with ID 10 is offline

Server with ID 10 is offline

