One framework.
Mobile & desktop.

# 3강. Angular Component

## DEVELOP ACROSS ALL PLATFORMS

Learn one way to build applications with Angular and reuse your code and abilities to build apps for any deployment target. For web, mobile web, native mobile and native desktop.

# 03 강.

## Angular Component

# ❑ 1) Angular의 Component 개념

컴포넌트는 다음과 같이 3 영역으로 구분된다.

가. import 영역
나. @Component
다. 컴포넌트 클래스 영역

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

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
```

```
export class AppComponent {
  title = 'my-app';
}
```

화면 처리
및
동적 처리 담당

# ❑ 1) Angular의 Component 개념

## 가. import 영역

컴포넌트에서 필요한 외부 모듈(모듈 패키지)의 특정 클래스를 사용하기 위해 import 필요.

외부 모듈 종류 2가지

- **라이브러리 모듈** : Angular에 내장된 모듈을 의미, **@ 사용**

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
```

- **사용자 정의 모듈** : 사용자가 필요에 의해서 만든 모듈 의미. **상대 경로 ./ 이용**.

```
import { AppComponent } from './app.component';
import { PersonComponent } from './person/person.component';
import { ItemEditComponent } from './item-edit/item-edit.component';
```

# ❑ 1) Angular의 Component 개념

## 나. @Component 영역(component.ts)

컴포넌트와 관련된 설정 정보를 지정하는 영역이다.

### - selector 속성

Component의 인스턴스를 생성하고 삽입하도록 지시하는 CSS 선택자.

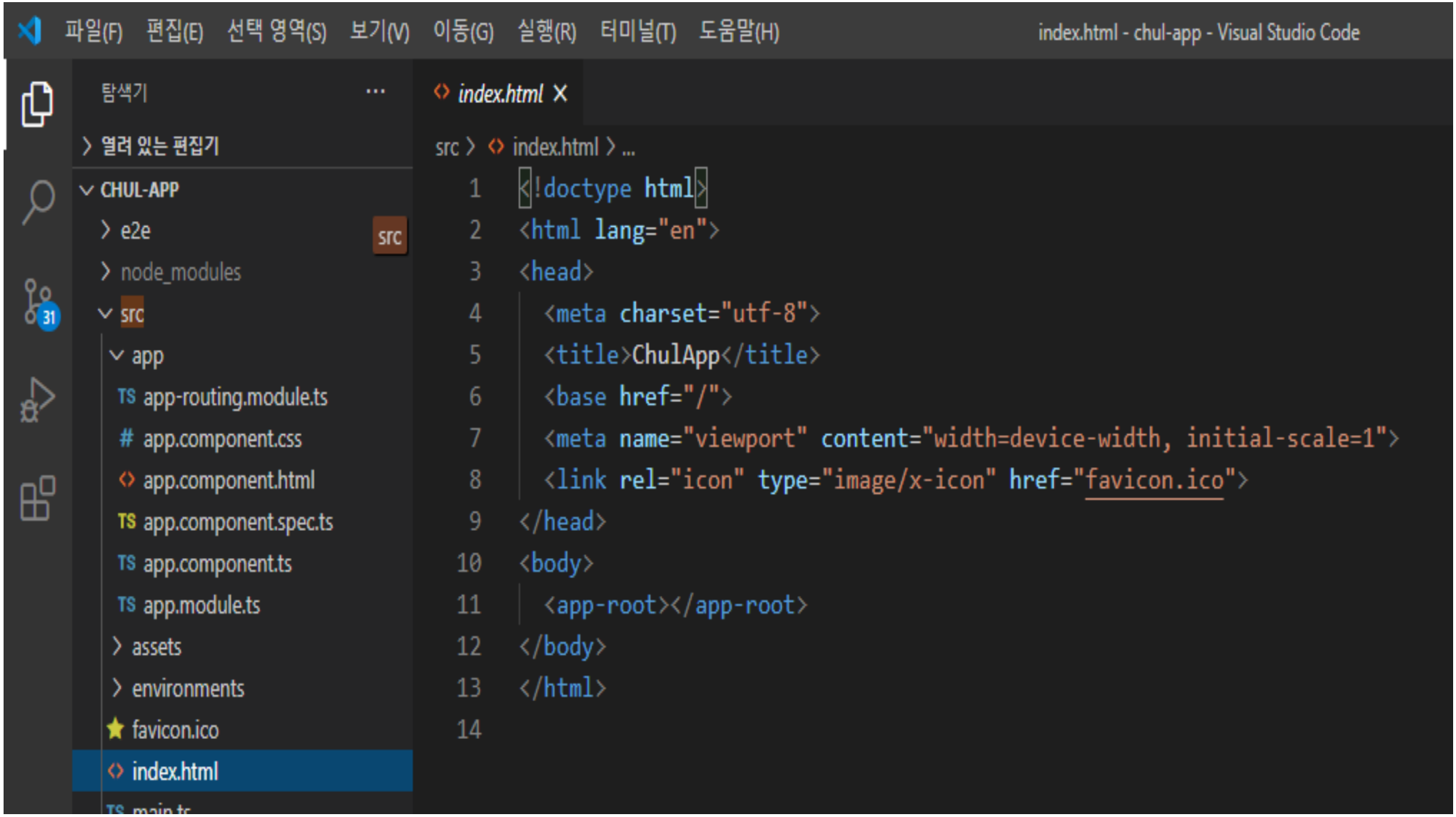
```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
```

index.html

my-app > src > index.html > ...

```
1   <!doctype html>
2   <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>MyApp</title>
6     <base href="/">
7     <meta name="viewport" content="wi
8     <link rel="icon" type="image/x-ic
9   </head>
10  <body>
11    <app-root></app-root>
12  </body>
13  </html>
```

# ❑ App-root의 index.html에서 로딩



## ❑ 1) Angular의 Component 개념

- templateUrl, template 속성

HTML 템플릿 파일의 상대 경로 또는 HTML 코드

- templateUrl을 template로 변경, `` back-tickt사용

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
```

```
@Component({
  selector: 'app-root',
  template:`
            <table>
             <tr>
               <td>홍길동</td>
             </tr>
            </table>
           `,
  styleUrls: ['./app.component.css']
})
```

# ❑ 1) Angular의 Component 개념

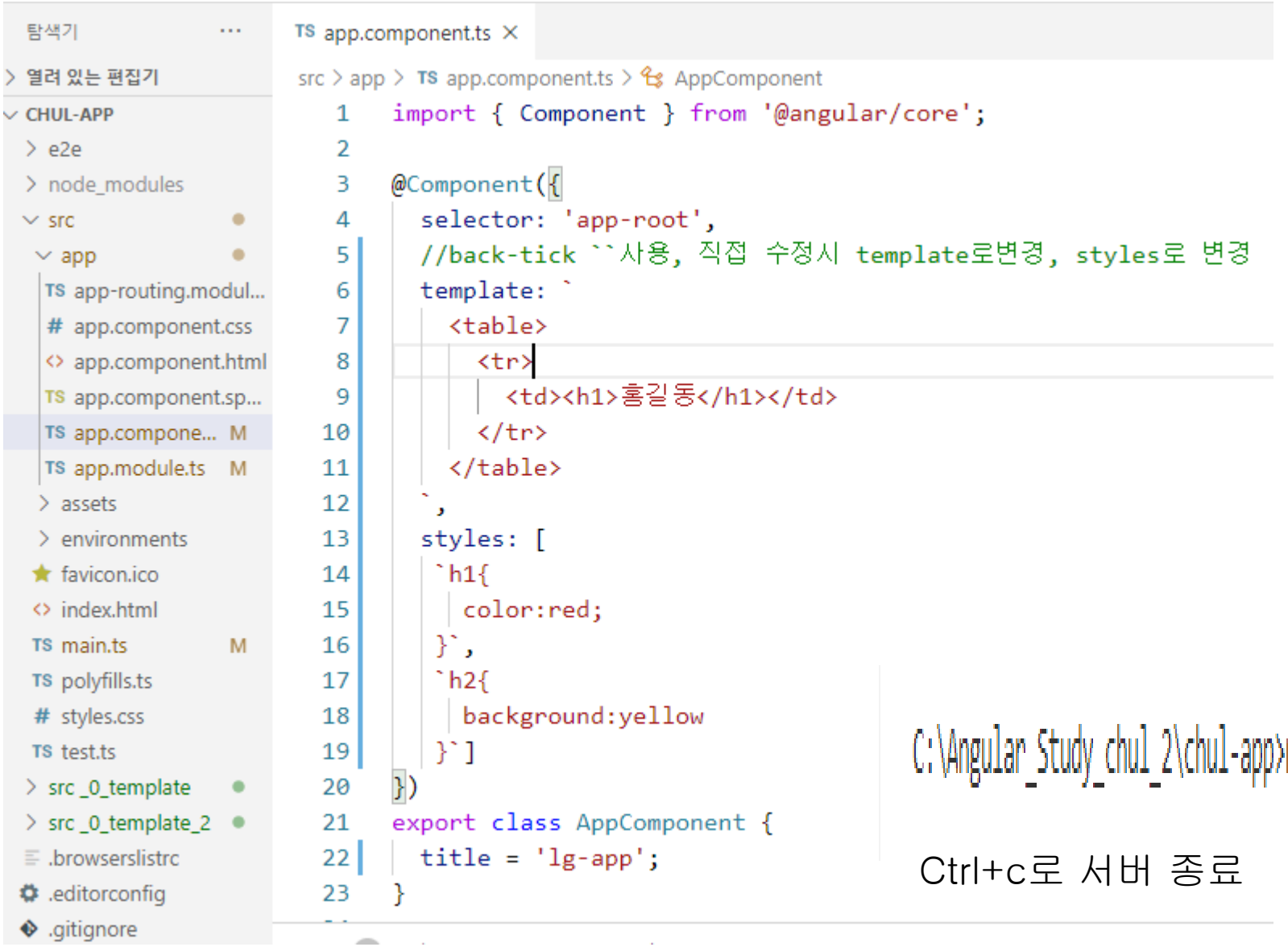
## - styleUrls, styles 속성

CSS 스타일시트 파일의 상대 경로 또는 CSS 코드
- styleUrls-> styles로 변경 `` back-tickt사용

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
```

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styles: [`
    h1{
      color:red
    }`,
    `h2{
      background:yellow
    }`]
})
```

# ❑ 실습




C:\Angular_Study_chul_2\chul-app>ng serve

Ctrl+c로 서버 종료

## 다. 컴포넌트 클래스 영역

template 데이터 출력과 관련된 로직을 처리하는 영역.
외부에서 컴포넌트 클래스를 사용하기 위하여 export 처리.
( 다른 컴포넌트에서 필요시 import 하여 사용 가능 )

다음은 대표적인 로직 처리 예이다.

- HTTP서비스를 이용한 요청 결과를 받아서 template에 데이터를 반영하는 로직.
- template에서 이벤트를 받아서 이벤트에 대한 처리를 수행하는 로직.

# ❑ 2) Angular의 Component 중심의 개발

- 새로운 컴포넌트 생성 방법– 터미널의 실행 후
- Ng g component My 실행 My Comnent생성

**Component**

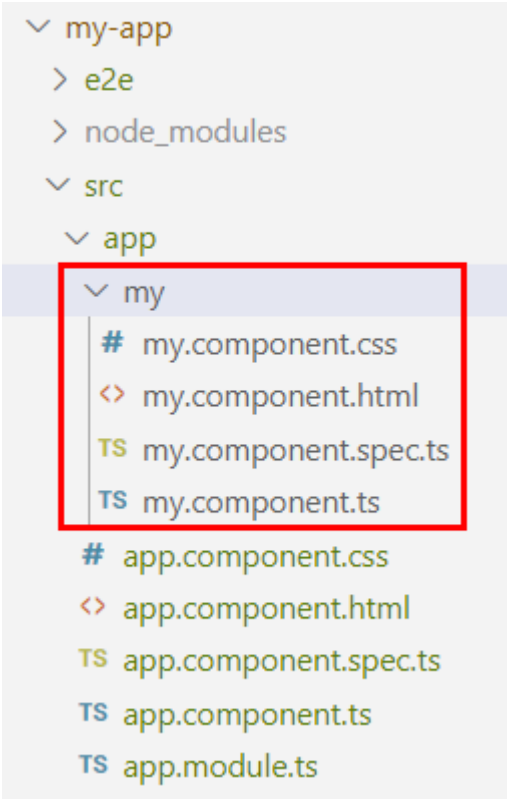
App.Module

```
ng g component My // Creates MyComponent
```

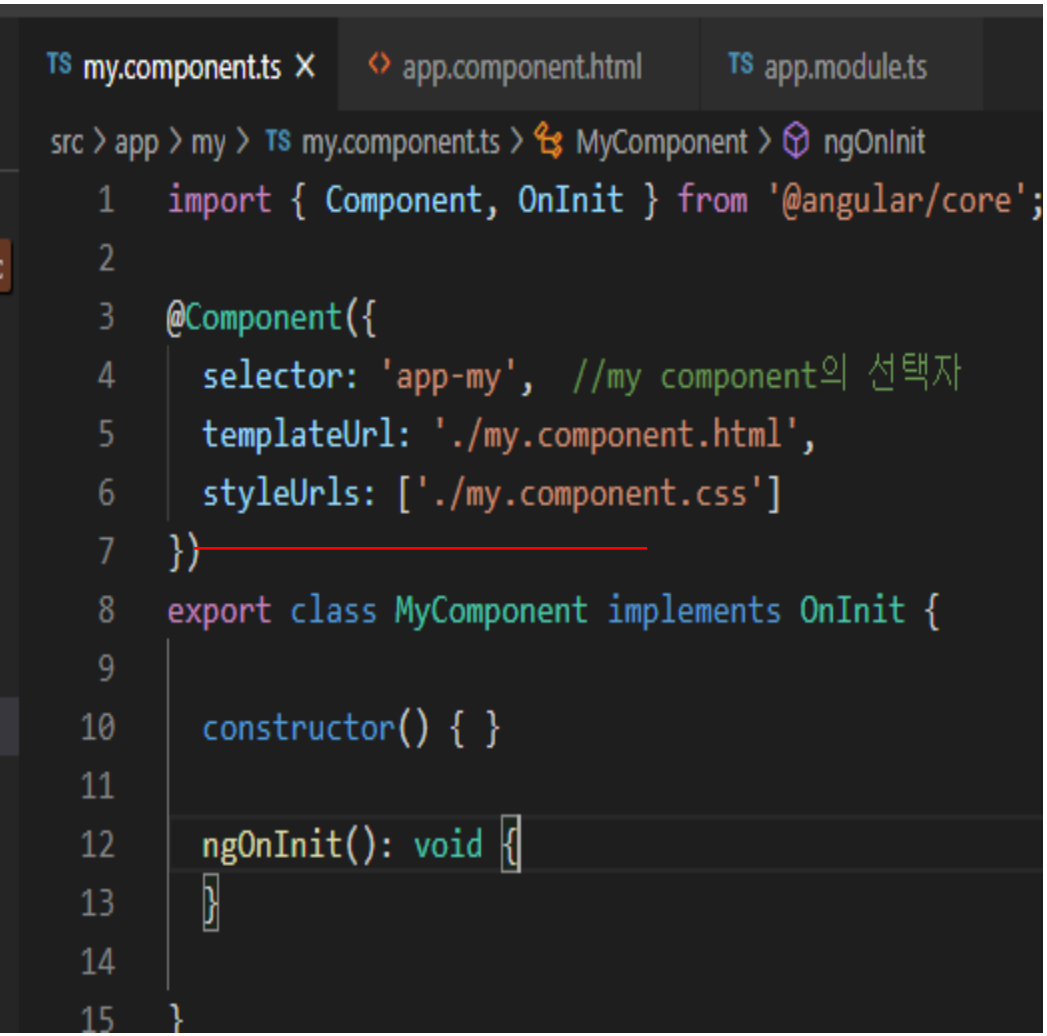
예>

```
C:\angular\my-app>ng g component My
```

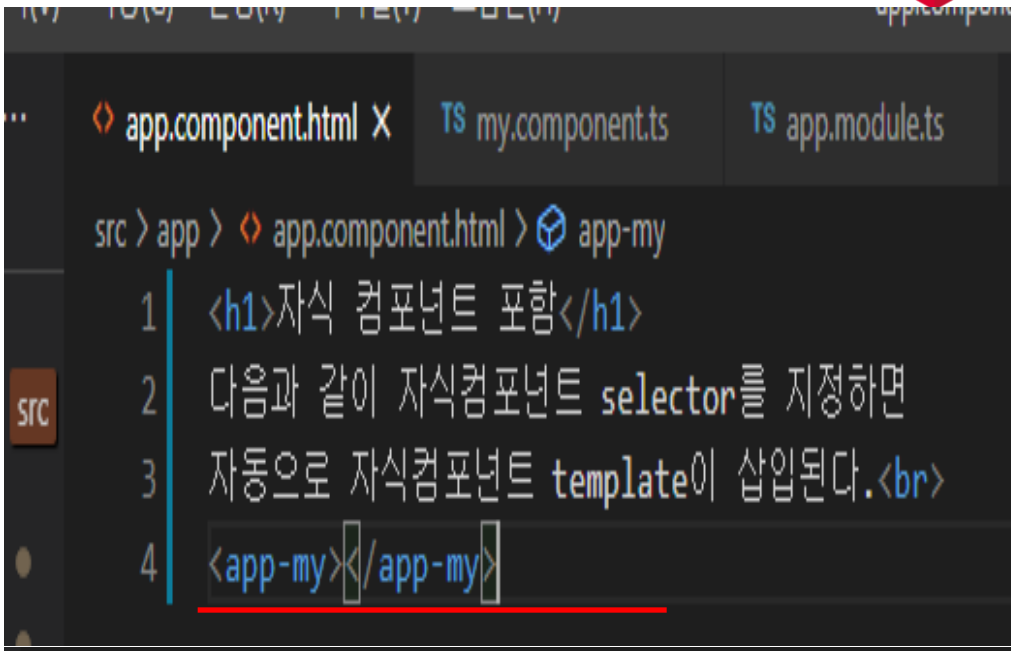
```
import { AppComponent } from './app.component';
import { MyComponent } from './my/my.component';

@NgModule({
  declarations: [
    AppComponent,
    MyComponent
  ],
```



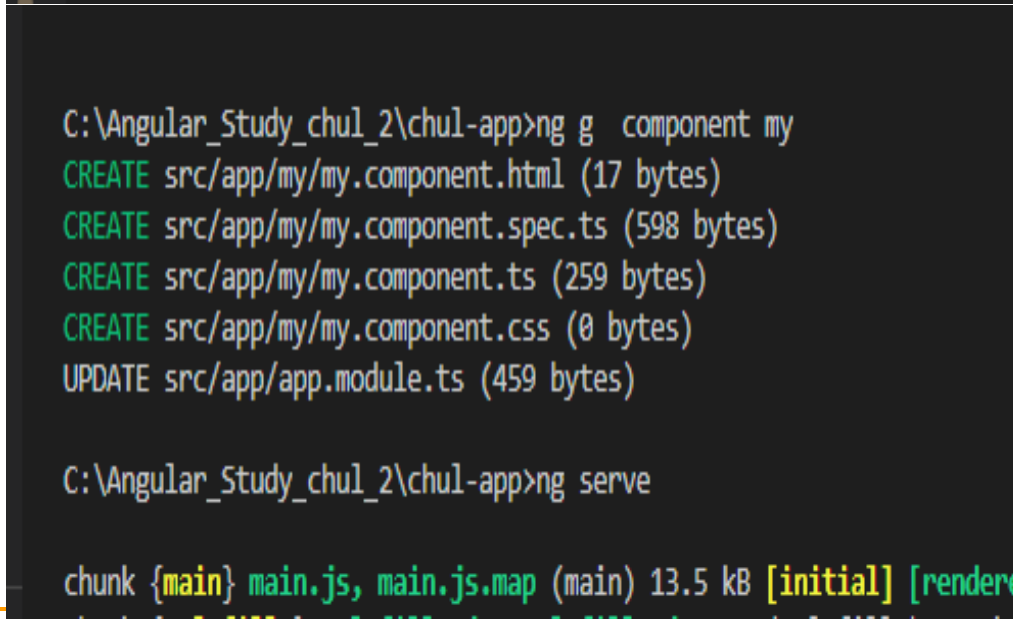
# ❑ 자식 컴포넌트를 부모컴포넌트에서 불러오기



```
TS my.component.ts X   app.component.html   TS app.module.ts
src > app > my > TS my.component.ts > MyComponent > ngOnInit
1   import { Component, OnInit } from '@angular/core';
2
3   @Component({
4     selector: 'app-my',  //my component의 선택자
5     templateUrl: './my.component.html',
6     styleUrls: ['./my.component.css']
7   })
8   export class MyComponent implements OnInit {
9
10    constructor() { }
11
12    ngOnInit(): void {
13    }
14
15  }
```



```
app.component.html X   TS my.component.ts   TS app.module.ts
src > app > app.component.html > app-my
1   <h1>자식 컴포넌트 포함</h1>
2   다음과 같이 자식컴포넌트 selector를 지정하면
3   자동으로 자식컴포넌트 template이 삽입된다.<br>
4   <app-my></app-my>
```

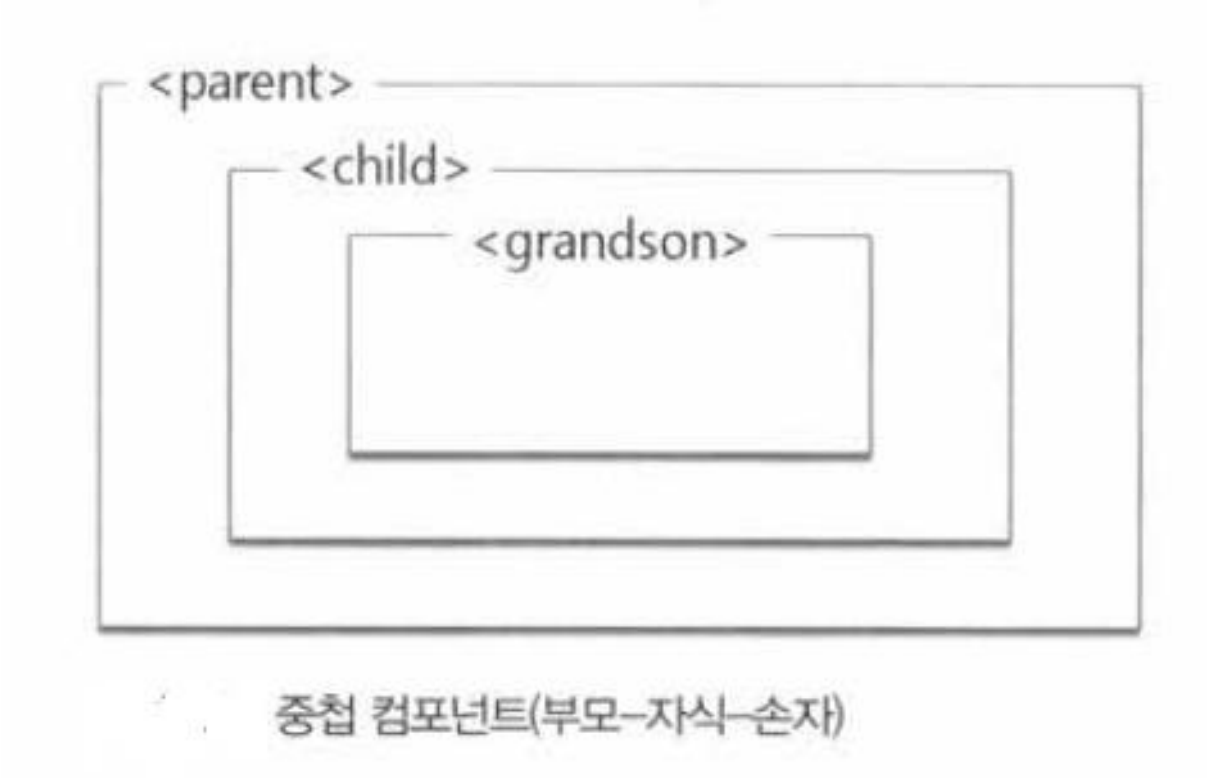


```
C:\Angular_Study_chul_2\chul-app>ng g  component my
CREATE src/app/my/my.component.html (17 bytes)
CREATE src/app/my/my.component.spec.ts (598 bytes)
CREATE src/app/my/my.component.ts (259 bytes)
CREATE src/app/my/my.component.css (0 bytes)
UPDATE src/app/app.module.ts (459 bytes)

C:\Angular_Study_chul_2\chul-app>ng serve

chunk {main} main.js, main.js.map (main) 13.5 kB [initial] [rendere
```

## ❑ 2) Angular의 Component 중심의 개발

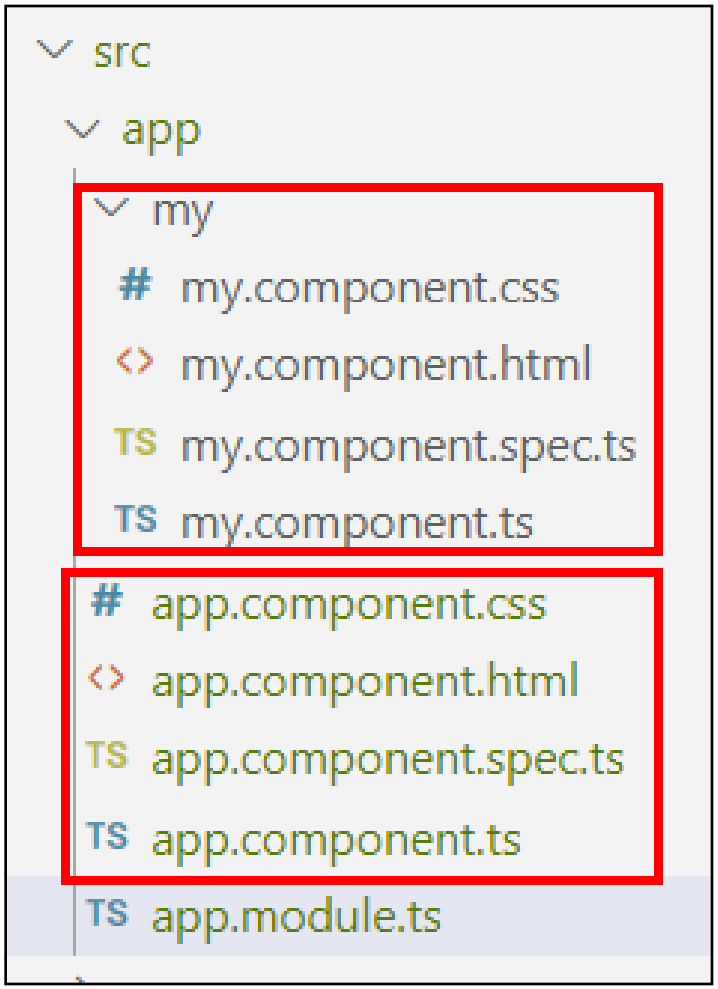
* 새로운 컴포넌트를 생성하는 이유는 **중첩**된 레이아웃으로 화면을 구현하기 위함이다.

컴포넌트는 포함 관계에 따라서 계층구조로 관리됨.




중첩 컴포넌트(부모-자식-손자)

# ❑ 2) Angular의 Component 중심의 개발




자식 컴포넌트

부모 컴포넌트

# ❑ 2) Angular의 Component 중심의 개발

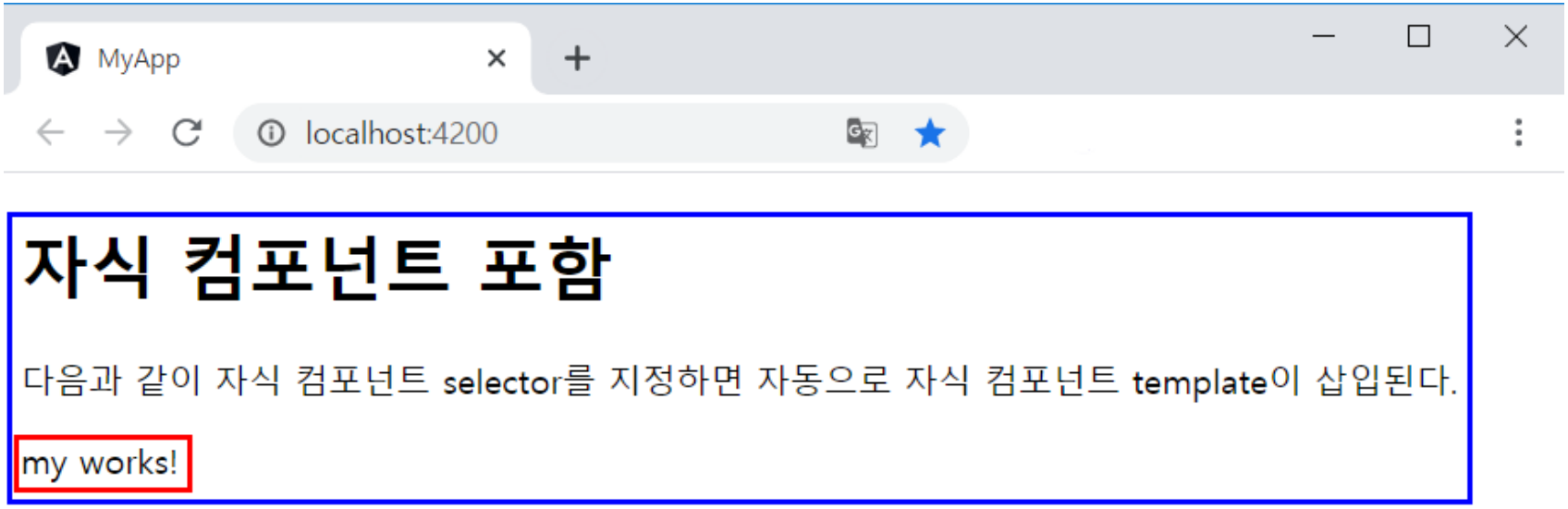
부모 컴포넌트인 app.component에서 자식 컴포넌트를 포함하기 위해서는
다음과 같이 부모 컴포넌트에서 자식 컴포넌트의 selector 값을 사용하면 된다.

```
@Component({
  selector: 'app-my',
  templateUrl: './my.component.html',
  styleUrls: ['./my.component.css']
})
```

app.component.html

my-app > src > app > app.component.html > ...

```
1  <h1>자식 컴포넌트 포함</h1>
2  다음과 같이 자식 컴포넌트 selector를 지정하면
3  자동으로 자식 컴포넌트 template이 삽입된다.<br>
4  <app-my></app-my>
5
```



-1 . Chid실습

Ng g component child　　: child 생성 child: app-child

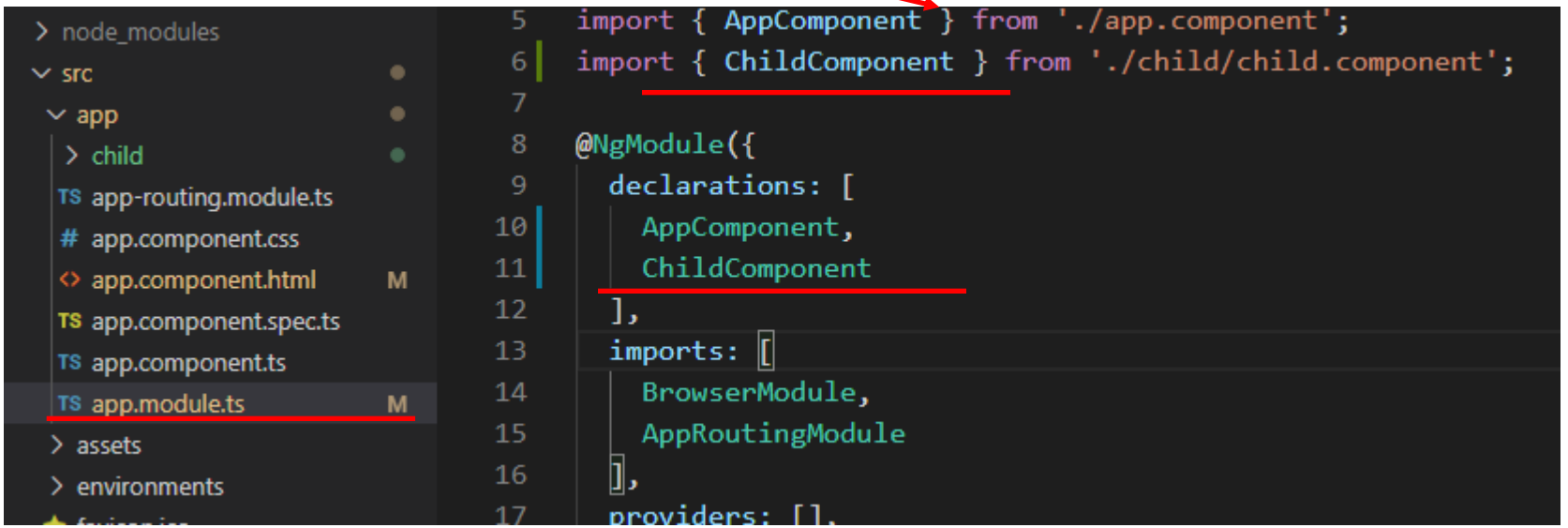
Ng g 명령 사용시 App.module.ts에 Childcomonent 자동 등록

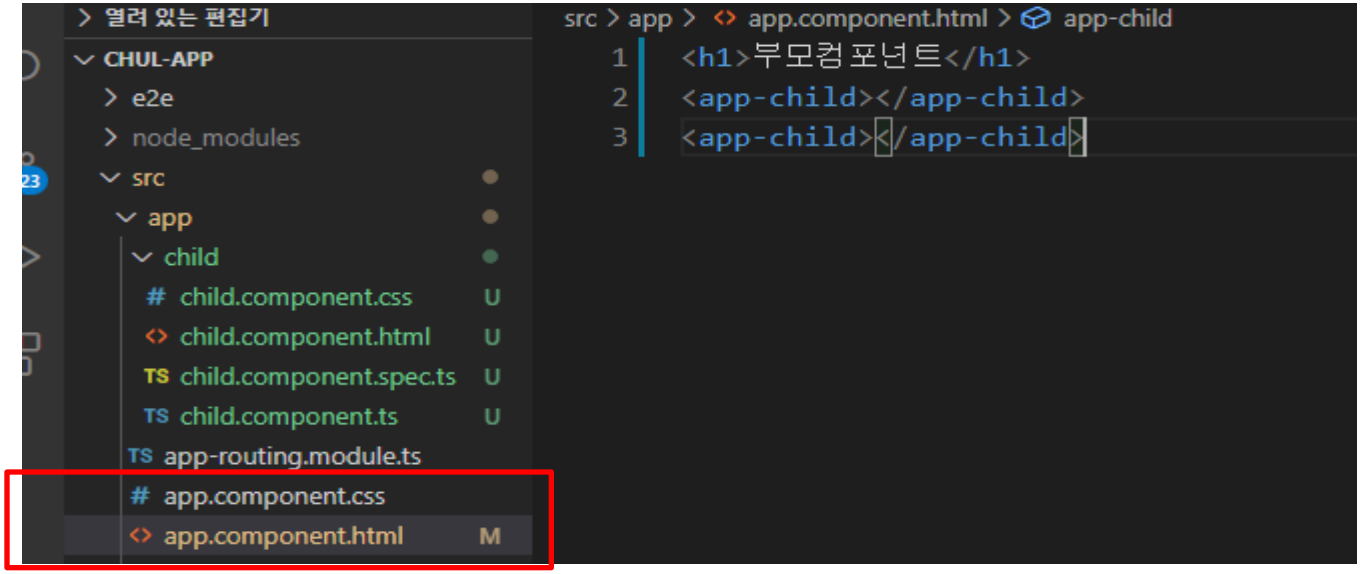
```
C:\angular_chul\chul-app>ng g component child
CREATE src/app/child/child.component.html (20 bytes)
```

1.app.module.ts에 자동등록 (복사시는 수동 등록 시켜야 함.)
Child: app-child ➔ app.modue.ts 에 Childcompent란 이름으로 등록
parent: chul-app ➔ AppCompnent로 등록

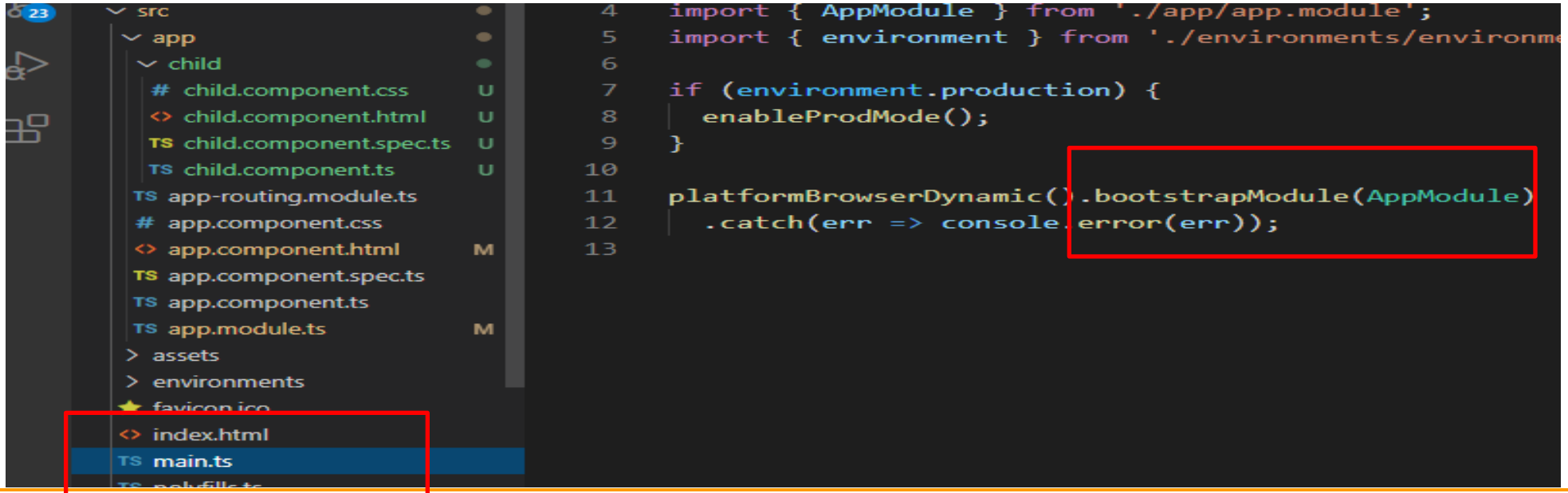
```
p > TS app.component.ts > ...
import { Component } from '@angular/core'

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'chul-app';
}
```

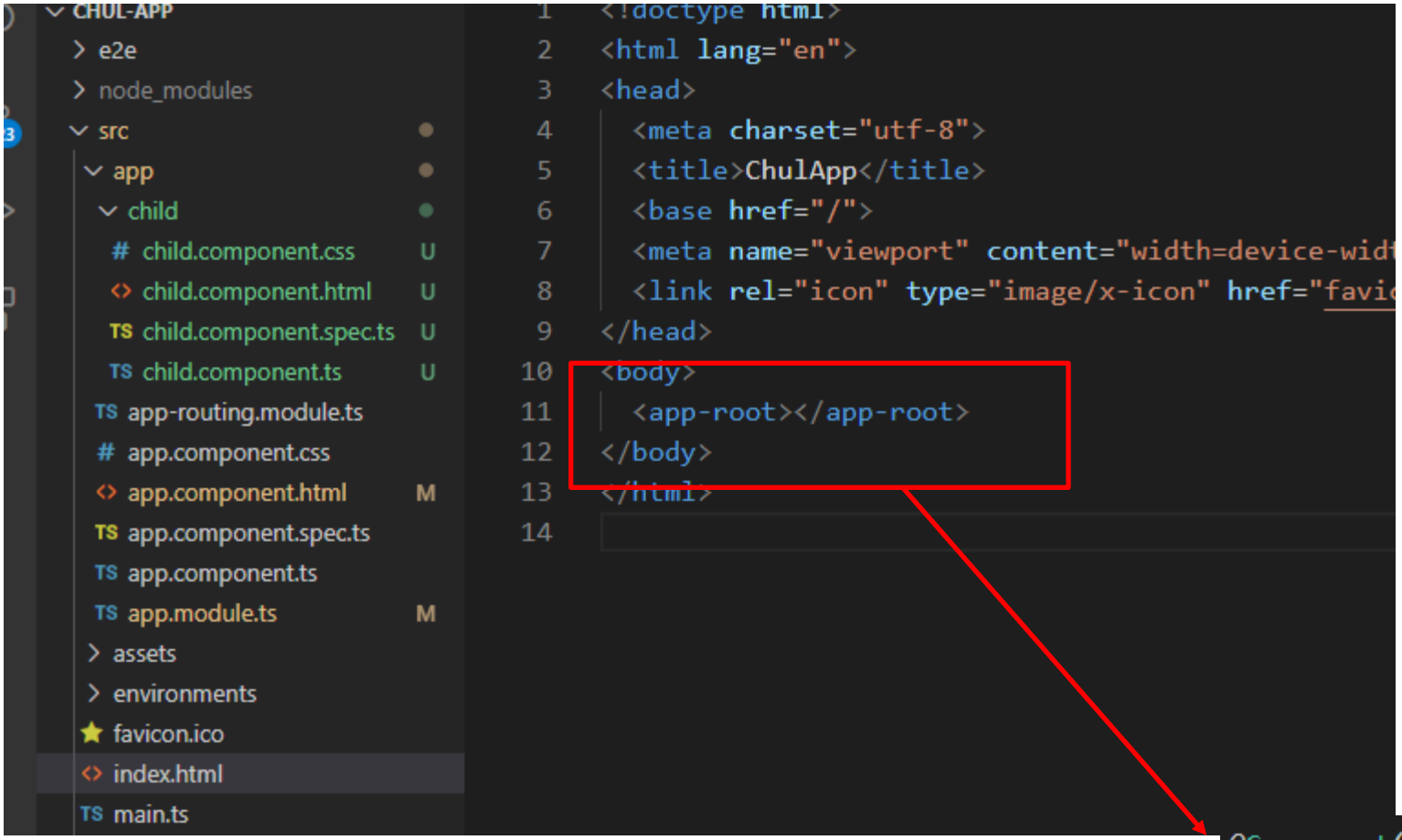
- App.module.ts





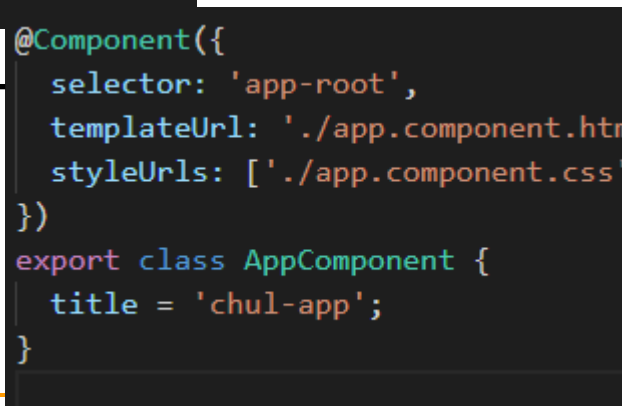

Main.ts의 부트스트랩에 등록




CHUL-APP
e2e
node_modules
src
app
child
child.component.css
child.component.html
child.component.spec.ts
child.component.ts
app-routing.module.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
assets
environments
favicon.ico
index.html
main.ts
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>ChulApp</title>
<base href="/">
<meta name="viewport" content="width=device-wid
<link rel="icon" type="image/x-icon" href="favi
</head>
<body>
<app-root></app-root>
</body>
</html>

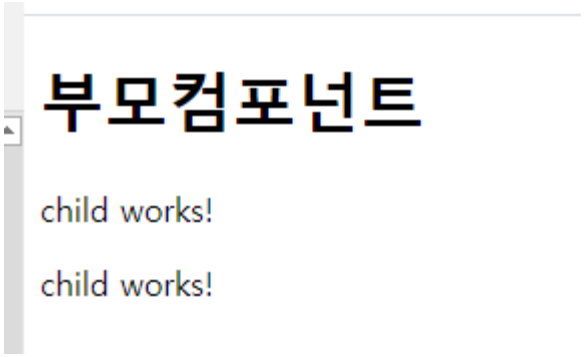

@Component({
selector: 'app-child',
templateUrl: './child.component.html',
styleUrls: ['./child.component.css']
})


src > app > app.component.html > app-child
<h1>부모컴포넌트</h1>
<app-child></app-child>
<app-child></app-child>


@Component({
selector: 'app-root',
templateUrl: './app.component.htm
styleUrls: ['./app.component.css'
})
export class AppComponent {
title = 'chul-app';
}

## ❑ 프로젝트의 실행

```
UPDATE src/app/app.module.ts (471 bytes)

C:\Angular_Study_chul_2\chul-app>ng serve
```



# ❑ 실습 문제 1

다음과 같은 화면UI를 위한 어플리케이션을 작성하시오.


book.component.ts

app.component.ts

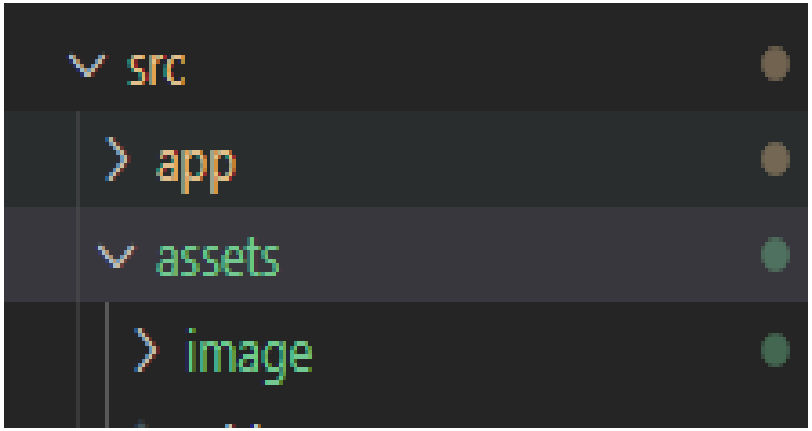
1) Book 컴포넌트 생성
2) 제공된 image 폴더를 src의 assets폴더에 복사
3) <ul>, <table>, <img> 태그를 적절히 적용하여 구현

## ❑ 실습

- Book 컴포넌트 생성

```
C:\angular_chul\chul-app>ng g component book
CREATE src/app/book/book.component.html (19 bytes)
```

- src – assets 밑 image 폴더 생성



# ❑ 실습

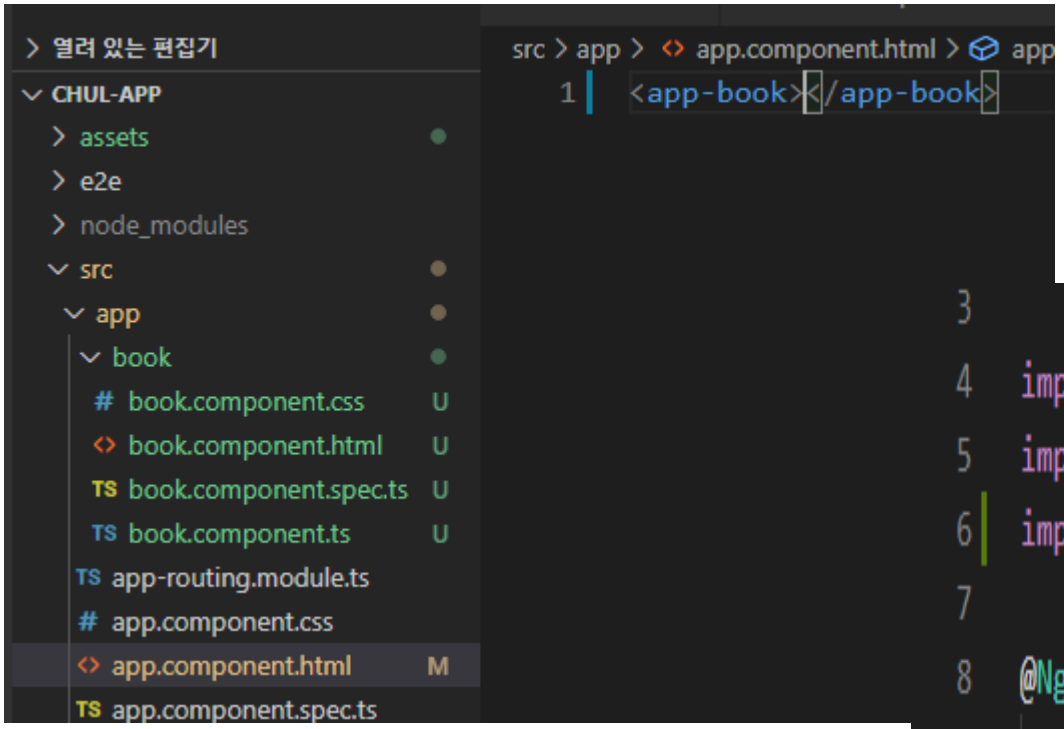
- Book 컴포넌트 : app-book

```
2
3   @Component({
4     selector: 'app-book',
5     templateUrl: './book.component.html',
6     styleUrls: ['./book.component.css']
7   })
8   export class BookComponent implements OnInit {
9
```

- Book 컴포넌트의
- html 수정

```
rc > app > book > <> book.component.html > ul > li
1   <h1>도서목록 6권</h1>
2   <ul>
3       <li>
4           <img src="../../assets/image/a.jpg" width="100" height="100"/>
5           위험한 식탁
6       </li>
7       <li>
8           <img src="../../assets/image/b.jpg" width="100" height="100"/>
9           공부의 비결
10      </li>
11      <li>
12          <img src="../../assets/image/c.jpg" width="100" height="100"/>
13          오메르타
14      </li>
15      <li>
```

-app.html의 수정 : child-book의 selector사용




-app.modules.ts 확인

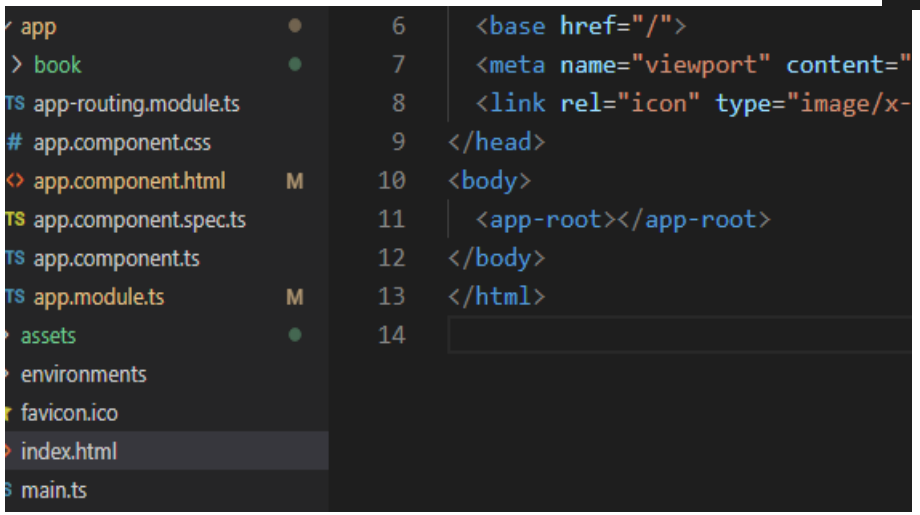
```
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { BookComponent } from './book/book.component';
7
8  @NgModule({
9    declarations: [
10     AppComponent,
11     BookComponent
12   ],
```

## -app-component : app-root 확인

src > app > TS app.component.ts > ...

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'chul-app';
}
```

```ts
import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
```




```html
  <base href="/">
  <meta name="viewport" content="
  <link rel="icon" type="image/x-
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

## ❑ 실습

```
C:\angular_chul\chul-app>ng serve --open

chunk {main} main.js, main.js.map (main) 15.5 k
```

# 도서목록 6권

- 

위험한 식탁
- 

공부의 비결
- 

오메르타
- 

수고하셨습니다.