2강. Angular 프레임워크 개요

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.

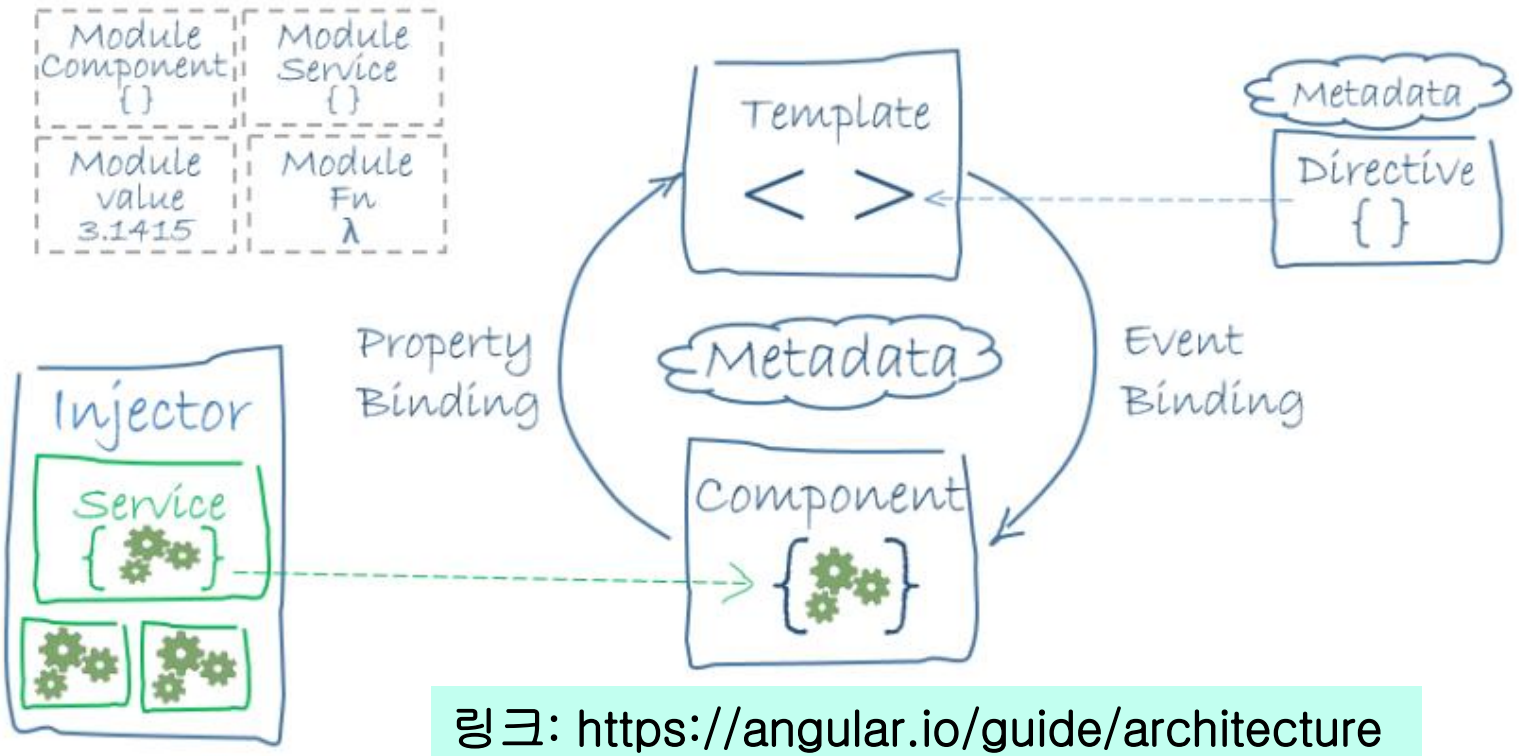
02 강.

Angular 프레임워크 개요

❑ 1) JS Framework 비교

종류5	설명
React	-웹 및 앱 개발 가능(React Native) -페이스북 지원 -커뮤니티 활발 -Virtual DOM -홈페이지: reactjs.org
Vue	-배우기 가장 쉬움 -가볍고 빠른 개발 -Evan You라는 개인이 프로젝트 리터 -가장 최신에 릴리즈된 프레임워크(2014년) -Virtual DOM 및 양방향 바인딩
Angular	-TypeScript 기반 -가장 복잡하고 큰 러닝커브(내부 동작 매커니즘 포함) -구글 지원 -커뮤니티 및 잘 정리된 문서 지원 -가장 빠르게 릴리즈된 프레임워크(2010년) -이전 버전의 AngularJS 보다 랜더링 성능이 향상 -홈페이지: angular.io

❑ 2) Angular 프레임워크 아키텍쳐



링크: https://angular.io/guide/architecture

* Angular 구성요소

- component: @Component 장식자 + component 클래스
- template: 화면구성 역할을 담당하고 html로 작성
- directive: template의 html태그내의 속성 핸들링
- service: 컴포넌트가 공통적으로 사용되는 비즈니스 로직 처리 담당
- module: 각 구성요소 관리 (Component, Service, Directive 등)

1. Component

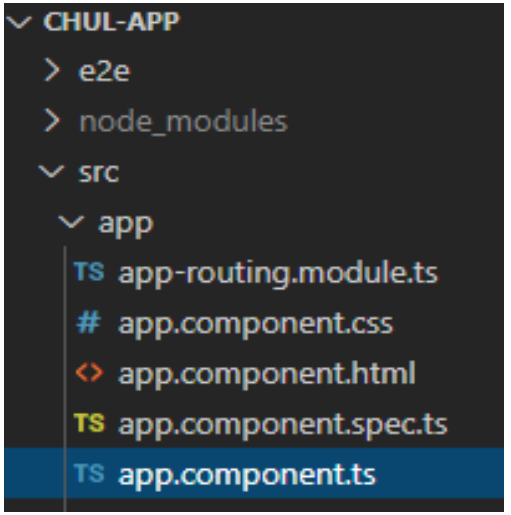
- Component는 HTML과 CSS 및 다양한 자바스크립트 코드로 구성
- Component 클래스 안에서 화면을 동적으로 처리하기 위한 속성 및 메서드를 사용.

Component는 다음과 같이 2가지 구성요소로 구성됨.
- @Component + component 클래스

TS app.component.ts ×

my-app > src > app > TS app.component.ts > ...

```
1   import { Component } from '@angular/core';
2
3   @Component({
4     selector: 'app-root',
5     templateUrl: './app.component.html',
6     styleUrls: ['./app.component.css']
7   })
8   export class AppComponent {
9     title = 'my-app';
10  }
```



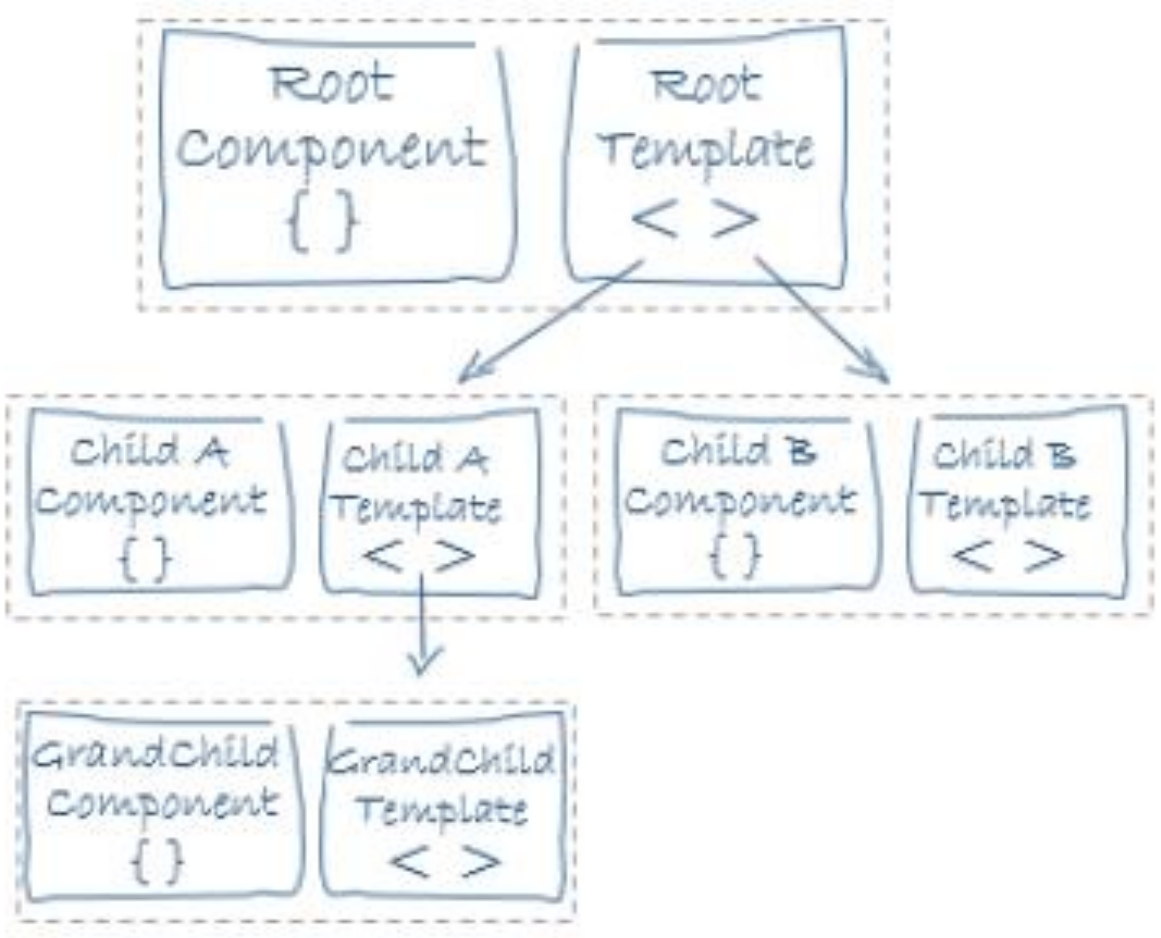
2. template

- template는 뷰(view)를 정의한다.
- HTML를 통해서 Angular에게 component를 rendering 하는 방법을 알려준다.
- HTML 코드 안에는 필요시 Angular 표현식(directive)이 사용된다.

```
1.  <h2>Hero List</h2>
2.
3.  <p><i>Pick a hero from the list</i></p>
4.  <ul>
5.    <li *ngFor="let hero of heroes" (click)="selectHero(hero)">
6.      {{hero.name}}
7.    </li>
8.  </ul>
9.
10. <hero-detail *ngIf="selectedHero" [hero]="selectedHero"></hero-detail>
```

❑ 2) Angular 프레임워크 아키텍쳐

- template 간에는 상속 관계 설정 가능(중첩 형태).



3. directive

-Angular가 렌더링 될 때 directive가 제시하는 지침에 따라서 template의 DOM이 동적으로 변경된다.
- Custorm directive는 @Directive 가 있는 클래스로서 필요시 생성이 가능하다.

다음과 같이 2가지 종류가 제공된다.

가. Structural directive

DOM의 요소를 추가,삭제 및 수정하여 화면 레이아웃을 변경.

src/app/hero-list.component.html (structural)

```
<li *ngFor="let hero of heroes"></li>
<hero-detail *ngIf="selectedHero"></hero-detail>
```

나. Attribute directive

- 기존 요소의 데이터를 변경.
- 대표적으로 ngModel

src/app/hero-detail.component.html (ngModel)

```
<input [(ngModel)]="hero.name">
```

4. service

- 여러 컴포넌트에서 사용하는 공통적인 기능 및 데이터 필요시 서비스를 사용한다.
- 일반적으로 @Injectable 사용하여 서비스 클래스를 생성한다.

대표적인 서비스 형태는 다음과 같다.
-http service
-logging service
-data service
-특정 계산 서비스등

src/app/heroes/hero.service.ts (CLI-generated)

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

@Injectable({
  providedIn: 'root',
})
export class HeroService {
  constructor() { }
}
```

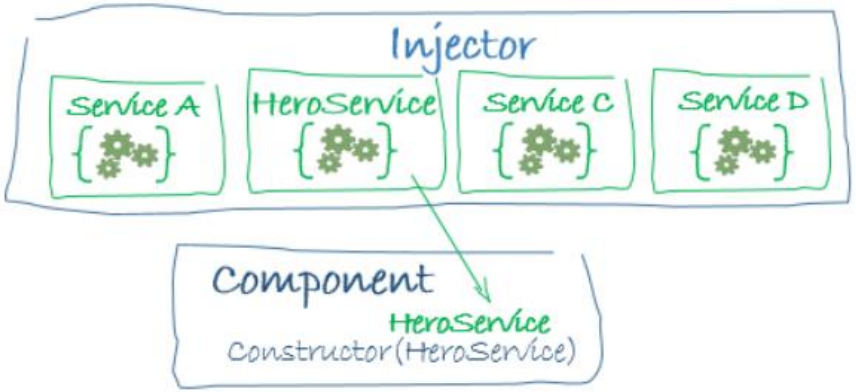
5. Dependency Injection (의존성 주입, DI)

- 특정 인스턴스가 외부에서 필요한 인스턴스를 주입 받는 방법을 의미.
- 대부분 외부에서 주입 받는 인스턴스는 service이다.

즉 Angular는 새로운 component에 필요한 service를 제공하기 위하여 DI를 사용하고 **생성자(constructor)를 사용하여 주입된다.**

src/app/hero-list.component.ts (constructor)

```
constructor(private service: HeroService) { }
```



일반적으로 필요한 모든 곳에서 서비스를 사용하기 위하여 root module에 등록한다.

src/app/app.module.ts (module providers)

```
providers: [
  BackendService,
  HeroService,
  Logger
],
```

선택적으로 @Component에 등록 가능.

src/app/hero-list.component.ts (component providers)

```
@Component({
  selector:    'hero-list',
  templateUrl: './hero-list.component.html',
  providers:  [ HeroService ]
})
```

6. module

-Angular 어플리케이션에는 적어도 하나의 모듈 클래스인 root 모듈(bootstrap모듈)이 있다. 일반적으로 AppModule 이라고 부른다.
-대부분의 응용 프로그램에서는 root 모듈 이외의 여러 가지 기능을 포함하는 기능 모듈이 존재한다.
-모듈은 @NgModule로 설정한다.

app.module.ts

my-app > src > app > app.module.ts > ...

```ts
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';


@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [
    BrowserModule,
    // AppRoutingModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

현재 모듈에서 필요한 component,directive,pipes 설정

현재 모듈에서 필요한 export된 외부 모듈 설정

현재 모듈에서 필요한 서비스(service) 설정

메인 화면 담당 component 설정. Root 모듈에서만 사용 가능.

❑ 2) Angular 프레임워크 아키텍쳐

main.ts 에서 root모듈인 AppModule을 부트스트랩한다.

TS main.ts

my-app > src > TS main.ts > ...

```ts
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

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

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

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.error(err));
```

수고하셨습니다.