انگولار یک فریم ورک جاوا اسکریپتی محبوب است برای ایجاد اپلیکیشن های client-side ای

انگولار یک پلتفرم توسعه برای ساخت اپلیکیشن های single-page application یا spa هستش (هم برای موبایل هم وب)

در این پلتفرم برای ساخت اپلیکیشن های client-side ای از html, css, javascript استفاده میشود.

انگولار خودش یک زبان برنامه نویسی نیست بلکه در آن از زبان برنامه نویسی جاوا اسکریپت یا تایپ اسکریپت استفاده میشود.

منظور از فریم ورک چیست؟

فریم ورک در واقع یک پلتفرم برای توسعه نرم افزار می باشد. فریم ورک ها یک سری کلاس ها و فانکشن های از پیش تعریف شده دارند که میتوان از آن ها بار ها در برنامه استفاده کرد (reusable) که این قابلیت کمک میکنه که از نوشتن کد از بیس خودداری کنیم و در واقع کد های clean تری داشته باشیم.

یک وب اپلیکیشن ساده چطوری کار میکنه؟  
وقتی از کلاینت به سرور با آدرسی مشخص درخواست ارسال میشود (http request) تا به یک منبعی دسترسی پیدا کند. مثلا میخواد دیتای مربوط به قسمت about رو از سرور بگیره تا ui اش رو نمایش بده



**منظور از SPA چیست؟**

یعنی بتوان بین لینک های مختلف در یک صفحه navigate کرد تنها با یک resource ینی میری home یا about اما همچنان فقط ریسورس index.html از سرور کال میشود. یعنی در url تغییرات رو میبینیم اما تنها یک content از سرور گرفته میشود. یعنی با این navigate کردنه ما content ای از سرور نمیگیریم و هیچ request ای به سرور ارسال نمیشود خب این یه مزیته چونکه برای تغییر content پیج ما تنها داریم از قابلیت های جاوا اسکریپت استفاده میکنیم نه اینکه ریکویست بزنیم به سرور. این باعث میشه سرعت اپلیکیشن ما و reactive بودنش بیشتر بشه!

چرا از انگولار استفاده میکنیم؟  
  
استفاده محض از کد های جاوا اسکریپت یا jquery یا vanilla javascript قابلیت نگه داری کد رو ضعیف میکنه. کد ها سخت و پیچیده میشه و کار توسعه را سخت تر میکند.

در صورت عدم استفاده از فریم ورک تست نرم افزار فرایندی پیچیده تر و سخت تر خواهد بود

احتمال نوشتن from scratch خیلی از فانکشنالیتی ها بالا میره اگر از انگولار استفاده نکنیم.

ورژن های انگولار:  
  
اولین نسخه angular js بوده که طراحی خیلیی پیچیده داشته که بعد به جای اون اومدن angular 2 رو معرفی کردند که کلا angular js رو کوبیدن و دوباره ساختن. و خیلی از issue های ورژن قبلی را فیکس کردند. بعد از اون angular 4 ,5 , 6 …. 16 هم معرفی شد.  
تا اینجایی که من مینویسم اخرین ورژن انگولار 18.2.7 که ۲ اکتبر ۲۰۲۴ معرفی شد. برای چک کردن آخرین ورژن به این لینک مراجعه کن:

https://versionlog.com/angular/

نصب انگولار:  
  
نصب شامل چهار مرحله است:  
 ۱. نصب nodejs:

نودجی اس یک محیطی برای اجرای کد های جاوا اسکریپتی خارج از مرورگر می باشد. ما مستقیما از نود جی اس قرار نیست استفاده کنیم. تنها برای اینکه بوسیله آن میتوانیم از یکی سری ابزار استفاده کنیم آنرا نصب میکنیم. مثلا یک سری پکیج ها رو از npm دانلود میکنیم. برای نصب نود جی اس به سایتش آدرس زیر مراجعه میکنیم:

<https://nodejs.org/en>

واسه بررسی اینکه چه ورژنی از نود روی سیستم نصبه از دستور node --version در cmd سیستم استفاده کن.

۲. نصب angular CLI :

یک محیط command line ای واسه خود انگولاره که از طریق اون میتونیم پروژه های انگولاری رو ایجاد کنیم یا یک سری پکیج نصب کنیم. با استفاده از دستور زیر میتونیم این angular cli رو نصب کنیم :

Npm install --global @angular/cli@latest

بعدش میتونی با دستور ng version در cmd ورژن انگولاری که نصب شده رو ببینی

۳. ایجاد یک پروژه در انگولار:   
اول یه فولدر بساز که توش میخوای پروژه رو بسازی و بعد برو تو همون فولدر و بعد این دستور رو بزن:

Ng new project-name

تو این مرحله یک سری پکیج از npm نصب میکنه و ممکنه یکم طول بکشه.

۴. اجرا و کامپایل پروژه :   
 میری تو همون مسیری که نصب اش کردی و دستور npm start یا ng serve رو میزنی که میاد کامپایل و run میکنه و bundle ها رو میسازه.

اولین فایلی که خونده میشه فایل index.htnl است به صورت زیره:

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>AngularEkart</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

</head>

<body>

  <app-root></app-root>

</body>

</html>

همه چی زیر سر این app-root هستش که میاد app component را لود میکند. که اونم میره از فایل ts از قسمت selector به فایل app.component.html را نمایش میدهد.

BootStrapping Angular Application:

بوت استرپ کردن یعنی لود کردن یا initialize کردن اپلیکیشن هر اپلیکیشن ای برای استارت شدن یک starting point دارد که این نقطه شروع برای انگولار فایل index.html است. که توش همین تگ app-root است. خب حالا بریم که این فرایند رو توضیح بدیم:

برای شروع برنامه دستور ng serve رو میزنیم. که میاد برنامه رو build میکنه و برنامه ی کامپایل شده رو توی دیسک ذخیره نمیکنه بلکه توی مموری ذخیره میکنه. و هر زمانی که هر تغییری در اپلیکیشن انگولار داده بشود از طریق angular CLI میاد recompile و آپدیت هم میکند. ساختار فایل index.html در انگولار به صورت زیر است.

وقتی ng serve را میزنیم:

برنامه را کامپایل میکند

Generates bundles

باندل ها را به index.html میاد inject میکند.

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>Angular Tutorial</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

</head>

  <body>

    <app-root></app-root>

  </body>

</html>

در این فایل هیچ فایل js ای وجود ندارد. وقتی باندل ها به index.html میاد inject میشه در فایل index.html مشخص نیستند. اگر بخوای ببینی که چجوری inject میشن باید به جای ng serve دستور ng build روبزنی و در فولدر .dist بیای index.html رو باز کنی تا بتونی فایل های inject شده را ببینی.

Angular CLI از Webpack برای باندل کردن فایل های js یا فایل های دیگر پروژه استفاده میکند. یعنی میاد کل اپلیکیشن رو اسکن میکنه ومثلا تمام فایل های js رو پیدا میکنه و همرو در قالب یک فایل در میاره. یا در واقع bundle میکنه.

البته webpack هر نوع فایلی رو میتونه bundle کنه و فقط محدود به js نیست.

پس Angular CLI از webpack به عنوان module bundler استفاده میکند. که البته برای کانفیگ کردن آن کار خاصی نیاز نیست ما انجام بدیم چون خود Angular CLI تمام تنظیمات را در پشت صحنه انجام میدهد!

خب حالا که bundle ها ساخته شدن Angular CLI میاد اونا رو به فایل index.html inject میکنه

حالا انگولار باید بره سراغ main entry point یعنی بره سراغ اون متد آغازکننده ی برنامه. واسه پیدا کردن entry point میره سراغ فایل angular.json که توش یه همچنین ساختاری داره:

         "options": {

            "outputPath": "dist/angular-ekart",

            "index": "src/index.html",

            "main": "src/main.ts",

            "polyfills": [

              "zone.js"

            ],

            "tsConfig": "tsconfig.app.json",

            "assets": [

              "src/favicon.ico",

              "src/assets"

            ],

            "styles": [

              "src/styles.css"

            ],

            "scripts": []

          },

از قسمت options مشخص شده که فایل تایپ اسکریپتی main.ts به عنوان main entry point برنامه در نظر گرفته شده. پس نقطه شروع اینجاست و اپلیکشن ما از اینجا آغاز میشه. اینم محتویات فایل main.ts:

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

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

platformBrowserDynamic().bootstrapModule(AppModule)

  .catch(err => console.error(err));

platformBrowserDynamic برای لود کردن اپلیکیشن انگولاردر مرورگر استفاده میشود. که یه متد به اسم bootstrapModule داره که در داخل ‌آن ما appModule رو لود میکنیم. که ماژول root ما محسوب میشه. پس بوسیله ی خط زیر ما روت ماژول که همان appModule باشه رو لود میکنیم.

platformBrowserDynamic().bootstrapModule(AppModule)

پس حالا میره سراغ appModule و میاد فایل app.module.ts رو لود میکند. که ساختارش این شکلیه:

@NgModule({

  declarations: [

    AppComponent,

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

که در واقع یک کلاس است که ب @ngModule دکوریت شده و در داخلش یه سری متا دیتا پاس داده شده .

داخل declaration تمام کامپوننت ها و دایرکتیو هایی که قرار است در این ماژول استفاده شوند معرفی میشوند.

داخل imports ماژول هایی که قرار است در این ماژول استفاده شوند رو معرفی میکنیم.

داخل providers سرویس هایی که قراره در این ماژول استفاده شوند رو معرفی میکنیم.

داخل bootstrap هم اسم اون کامپوننتی رو میذاریم که وقتی appModule لود شد قراره اون کامپوننت لود شود. که این کامپوننت باید داخل این ماژول هم باشد.

خب پس تو این مرحله میریم سراغ appComponent و فایل app.component,ts لود میشه که ساختارش اینشگلیه:

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

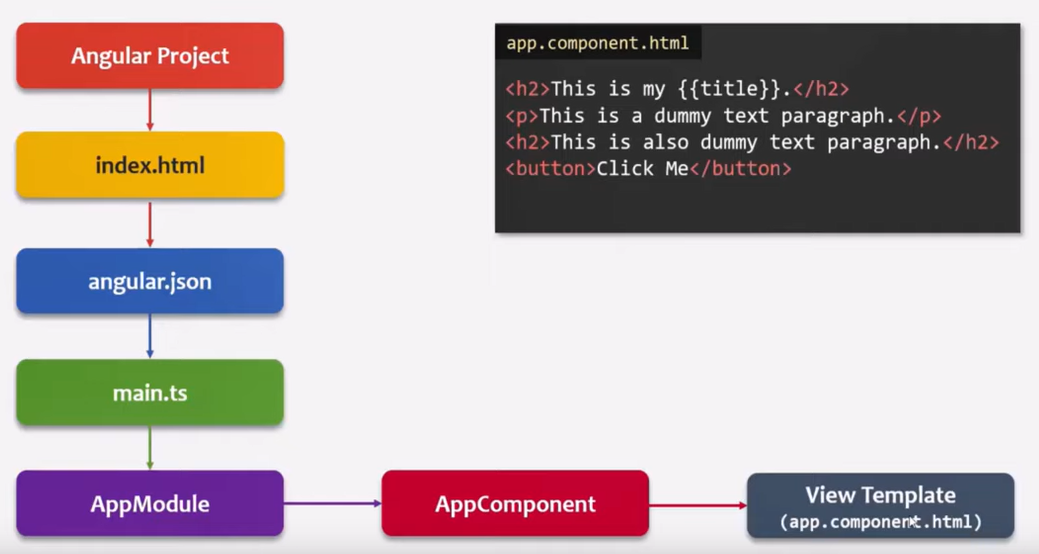
  title = 'angular-parisa';

}

از selector به عنوان html element استفاده میشود. و هرجایی از این html element استفاده شود view template آن لود خواهد شد. منظور از view Template همان فایل های html و css است که در templateUrl و styleUrl قرار داده شده است.

اگه نخوای آدرس html رو بدی باید به جای templateUrl از template استفاده کنی template : ‘<h1>test</h1>’پس حالا میره view template که همان فایل app.component.html است را لود میکند و در مرورگر نمایش میدهد.

مراحل بوت استرپ شدن اپلیکیشن انگولار در عکس زیر به طور خلاصه :



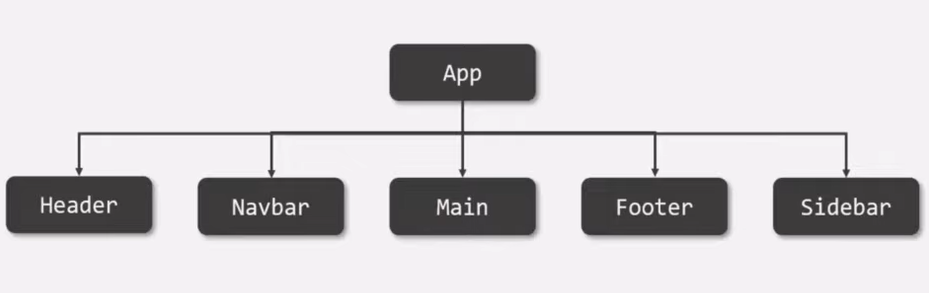
کامپوننت در انگولار:

انگولار یک فیچر کلیدی در انگولار محسوب میشود. کلا انگولار یک فریم ورک جاوا اسکریپتی component-base محسوب میشود.

یک کامپوننت تکه ای از user interface می باشد. که با ترکیب کامپوننت ها در واقع UI ساخته میشود.

هر اپلیکیشن انگولار حداقل یک کامپوننت را دارد. که در واقع همان root component است و طبق قرار داد همان appComponent می باشد.

هر اپلیکیشن انگولاری در واقع یک مجموعه ای از کامپوننت ها می باشد.

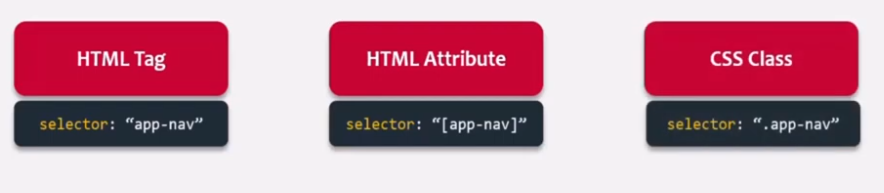


مراحل ساخت کامپوننت:

1. create typescript Class and export it
2. Decorate the class with @component decorator
3. Declare the class in main module file

با دستور :  
ng g c ….

انواع سلکتور کامپوننت در انگولار:



تا الان در مورد html tag گفتیم و میدونیم چحوری کار میکنه. حالا از این به بعد راجع به اون دو تای دیگه توضیح داده میشود.

مفهوم Data Binding در انگولار:

Data binding این امکان را میدهد که بین class component و view template (یا برعکس) ارتباط برقرار شود.

دو قسمت خیلی مهم در انگولار همین class component و view template می باشد. کامپوننت که همان لاجیک یا منطق ui ما رو در بردارد و view template هم شامل کد های html است. و ارتباط بین این دو بخش از طریق data binding انجام میشود.



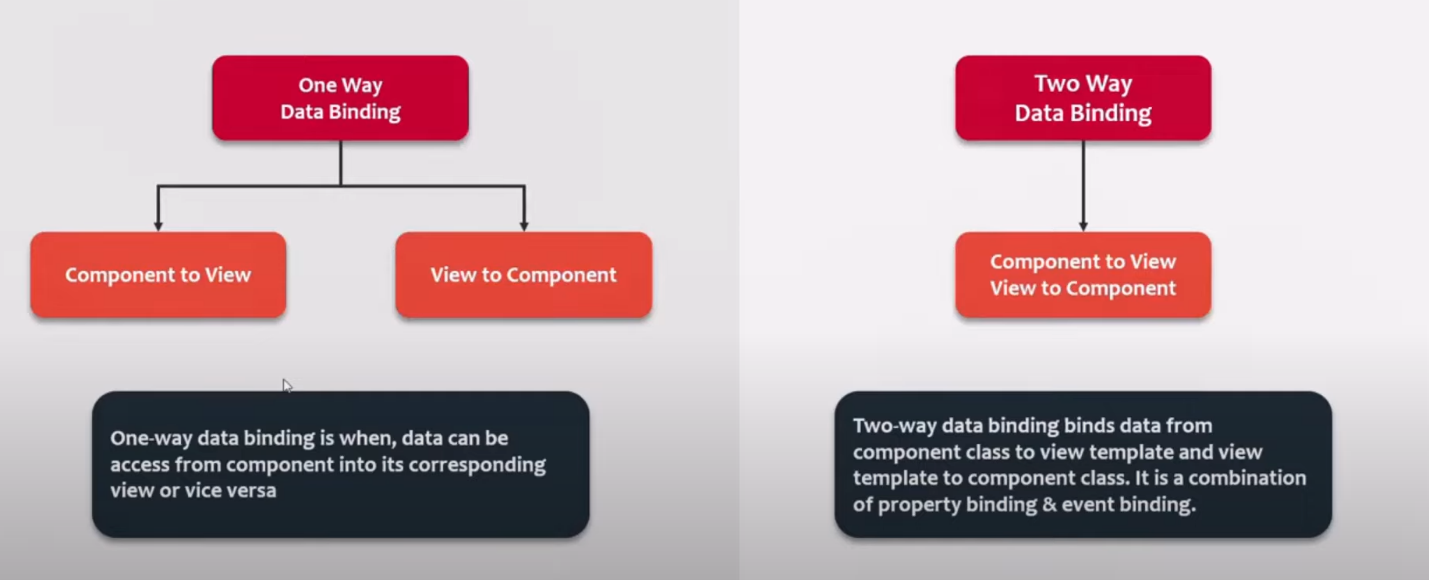
عکس زیر یک نمونه مثال از data binding می باشد:



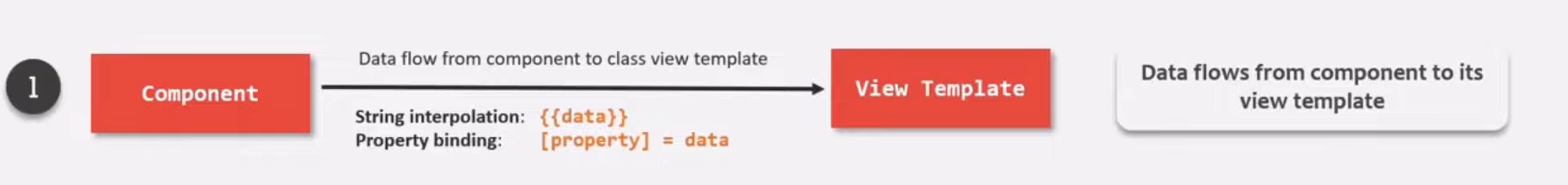
توجه شود که اینجا hidden داخل square bracet قرار داره یعنی یک attribute است و مقدار display که false است به این attribute تخصیص داده میشود.

ما دو مدل data binding داریم:

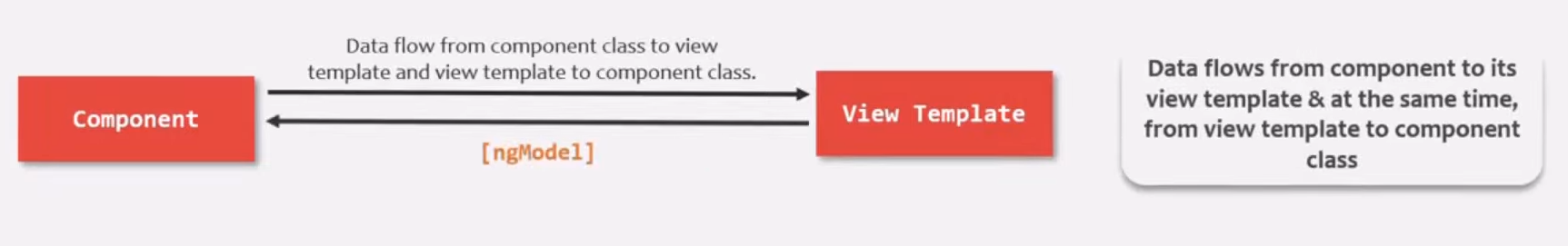
On way و two way



One way: یعنی ارتباط یا از طریق class component به view template یا برعکس انجام میشه. (ارتباط در هر لحظه فقط یک طرفه است). سه روش برای این مدل وجود دارد به نام های string interpolation و property bindingو event binding:  
  
  
  

Two way data binding : یعنی به طور همزمان و ترکیبی هم ارتباط از طریق class component به view template و هم از view template به class component انجام میگیرد. برای این مدل یک روش به اسم ngModel وجود دارد:



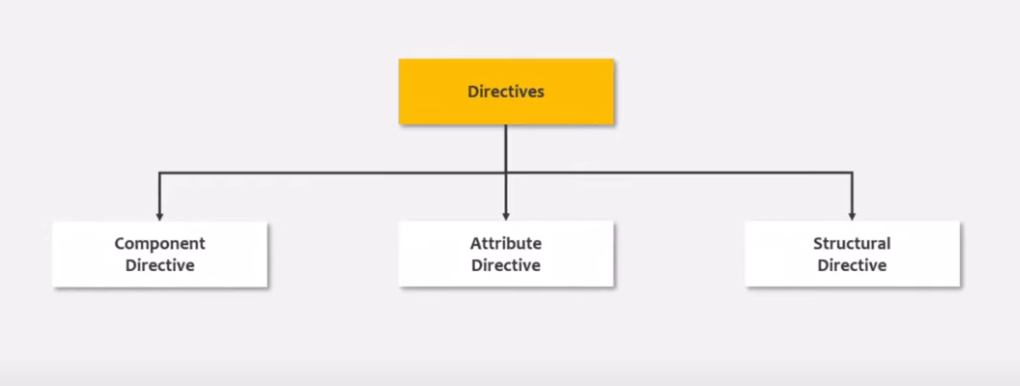
Two way data binding ترکیبی از property binding و event binding می باشد. دیتا به طور همزمان میتواند از class component به view template و برعکس منتقل شود.

Directive:

A directive is an instructions to the DOM

در واقع دایرکتیو DOM را دستکاری میکند.

سه مدل دایرکتیو وجود دارد :



Component directive همان کامپوننت انگولار است .

Attribute directive : بوسیله ی این دایرکتیو میتوانیم ظاهر و رفتار DOM را تغییر دهیم. به عنوان مثال مثلا دایرکتیوی تعریف کنیم که بک گراند را سبز کند.

<div changeToGreen>Some content</div>

چند مورد از attribute directive: ngStyle و ngClass

Structural Directive: برای افزودن یا حذف کردن DOM element در وب پیج استفاده میشود. مثال: ngIf - ngFor – ngSwitch

قبل از structural directive ها از \* استفاده میشود.

دایرکتیو هم یک کلاس است که با @Directive دکوریت میشود.

