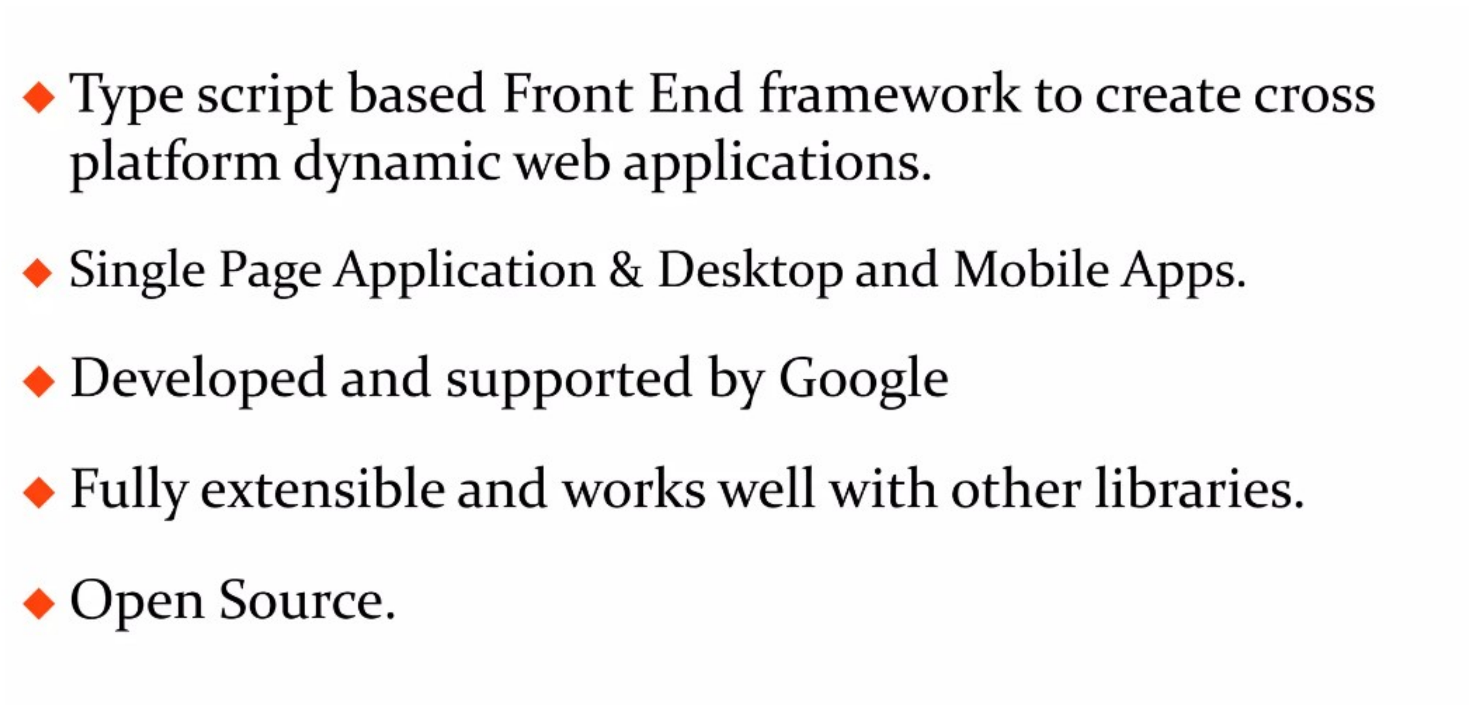
Contents

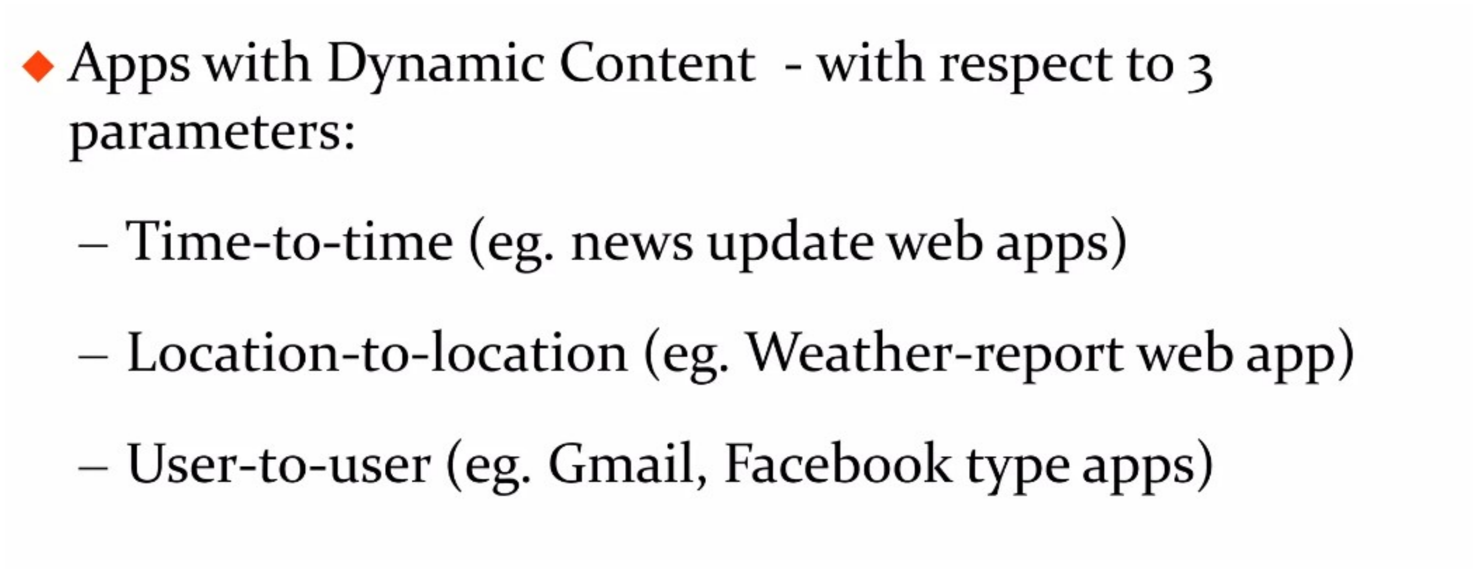
- *What is Angular and when to use?*
- *Pre-requisites*
- *History*
- *Overview of Angular Architecture*
- *Pros and Cons*

What is Angular



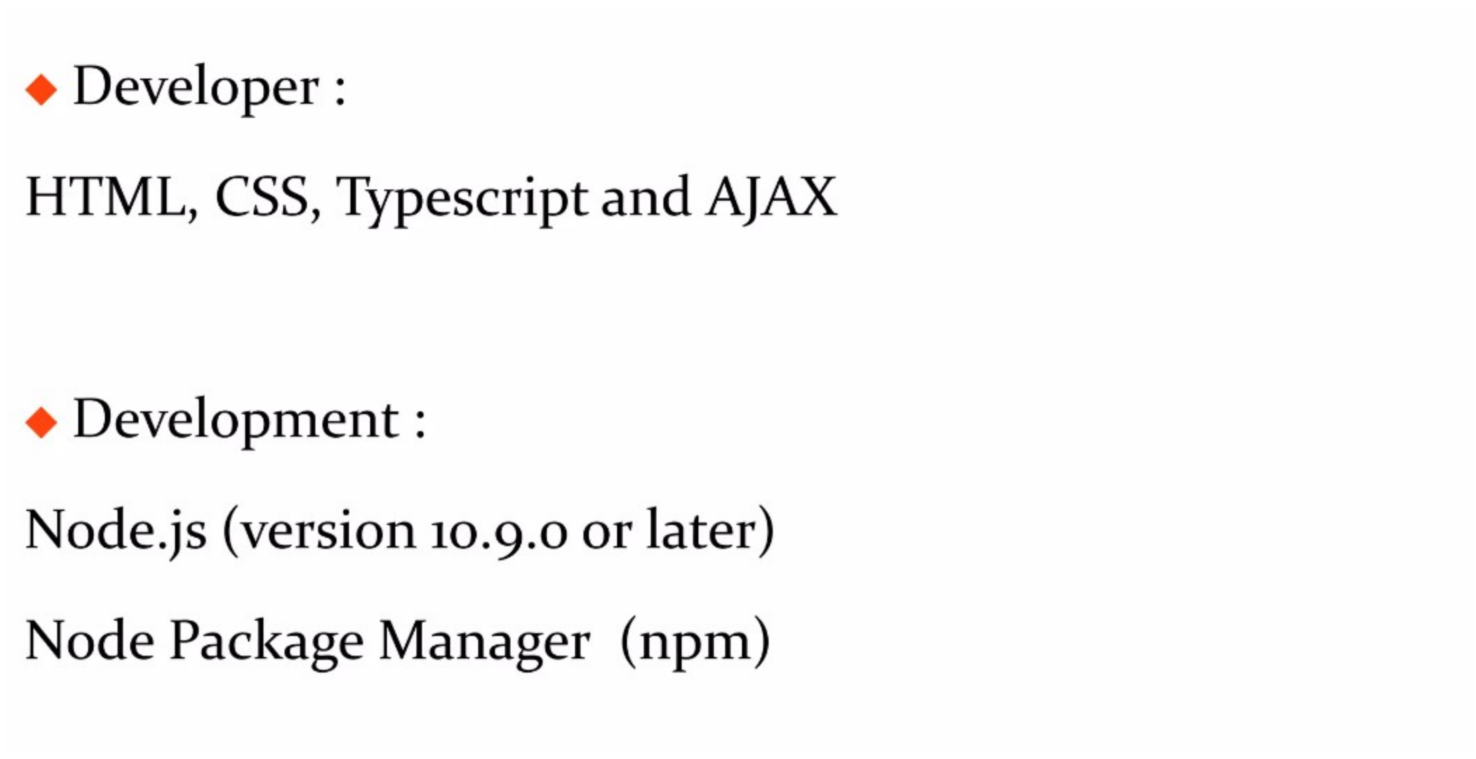
- Type script based Front End framework to create cross platform dynamic web applications.
- Single Page Application & Desktop and Mobile Apps.
- Developed and supported by Google
- Fully extensible and works well with other libraries.
- Open Source.

When to use



- Apps with Dynamic Content - with respect to 3 parameters:
  - Time-to-time (eg. news update web apps)
  - Location-to-location (eg. Weather-report web app)
  - User-to-user (eg. Gmail, Facebook type apps)

Prerequisties



- Developer :

HTML, CSS, Typescript and AJAX

- Development :

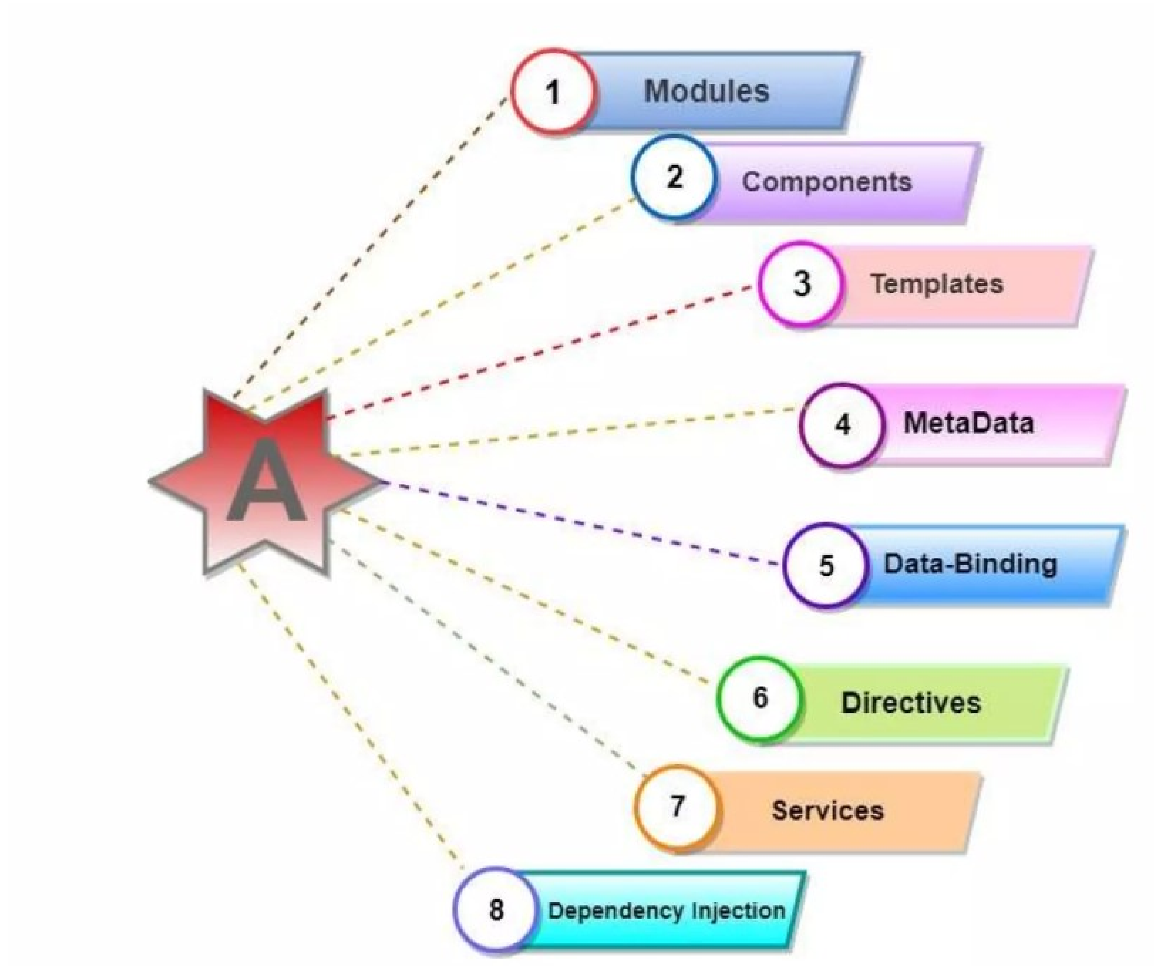
Node.js (version 10.9.0 or later)

Node Package Manager (npm)

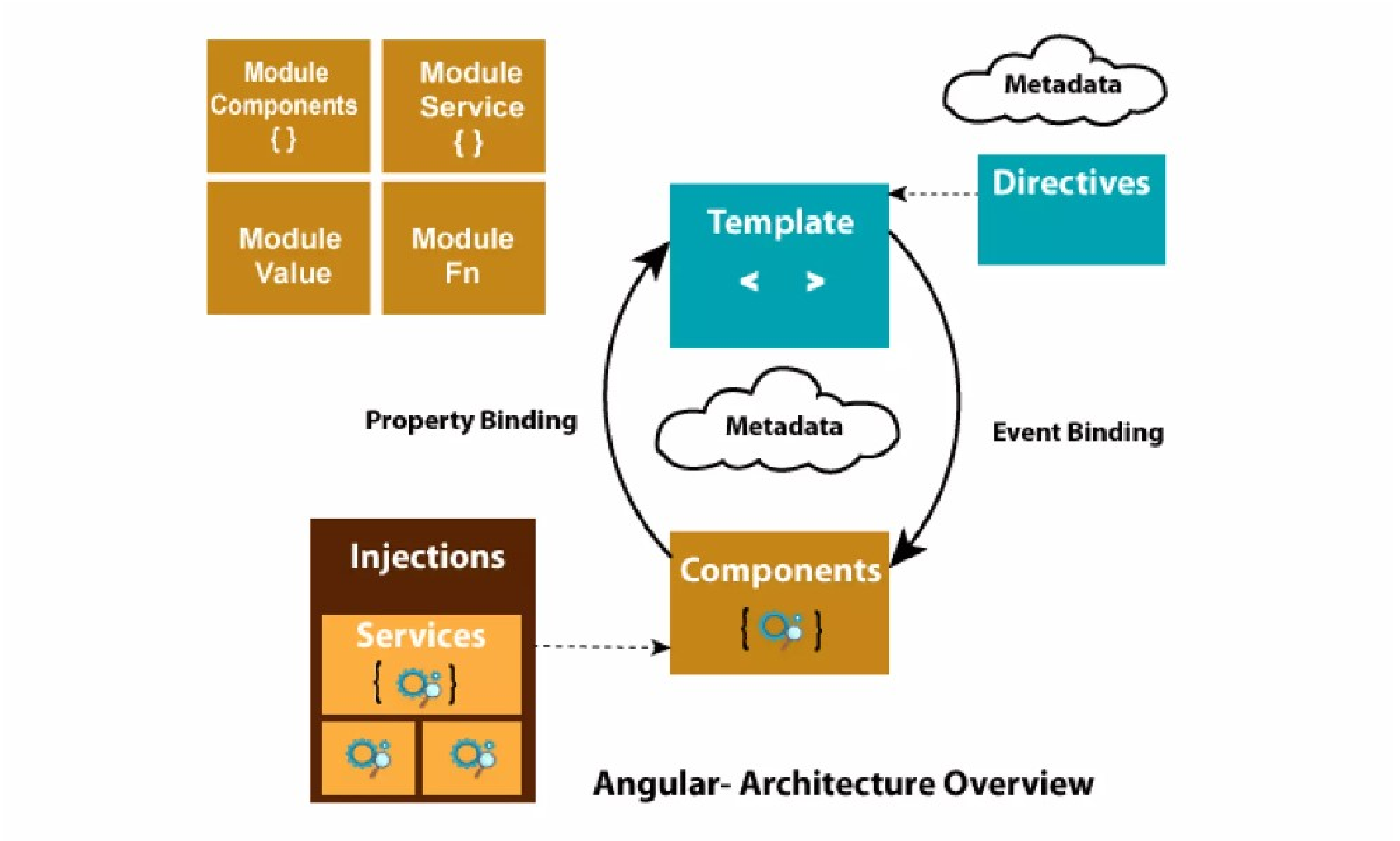
History

Version	Release date
Angular 17	November 8, 2023
Angular 16	3 May 2023
Angular 15	November 18, 2022
Angular 14	2 June 2022
Angular 13	4 November 2021
Angular 12	12 May 2021
Angular 11	11 November 2020
Angular 10	24 June 2020
Angular 9	6 February 2020
Angular 8	28 May 2019
Angular 7	18 October 2018
Angular 6	4 May 2018
Angular 5	1 November 2017
Angular 4	23 March 2017
Angular 2	14 September 2016

Architecture

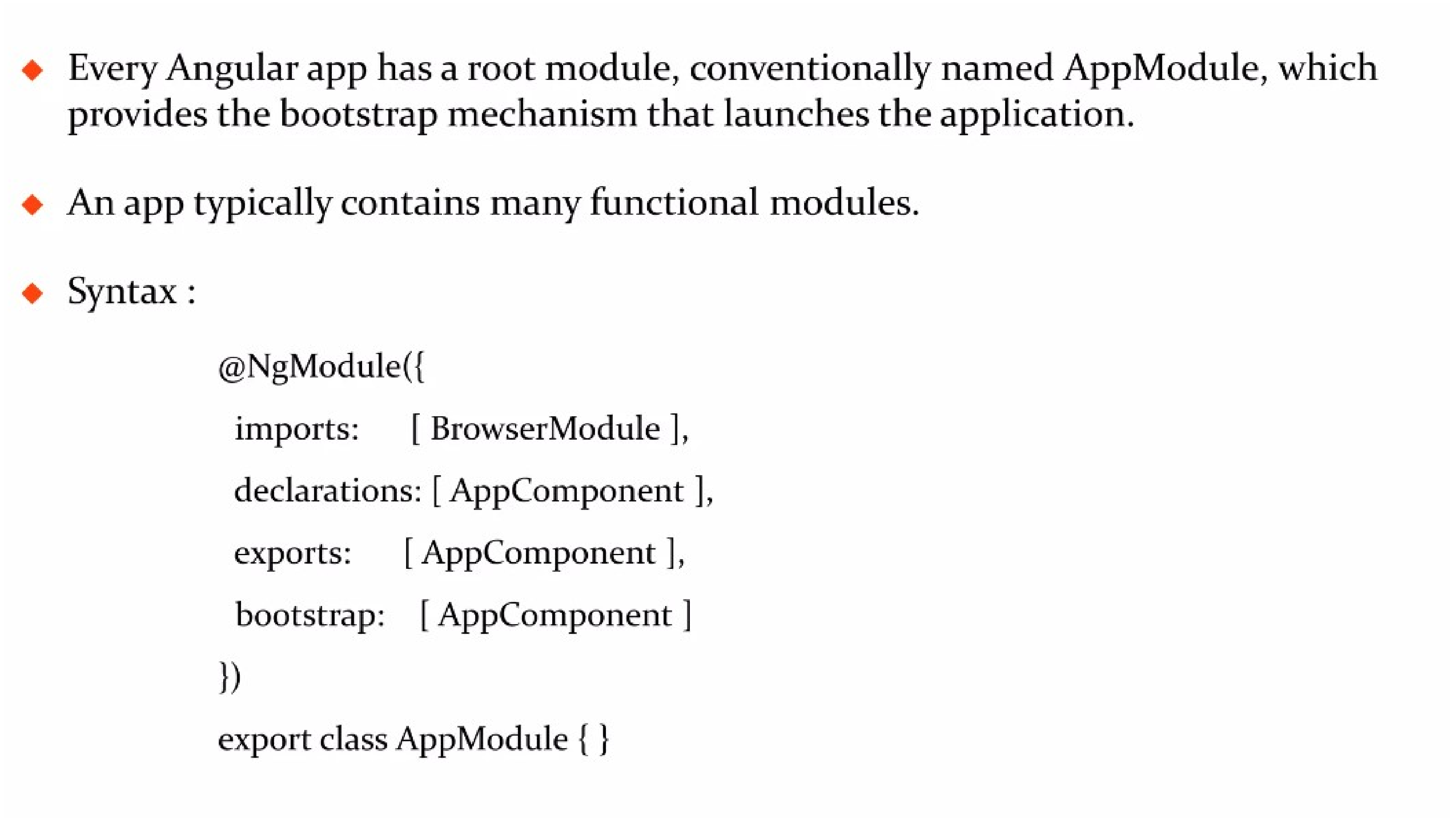


Overview of Architecture



Angular- Architecture Overview

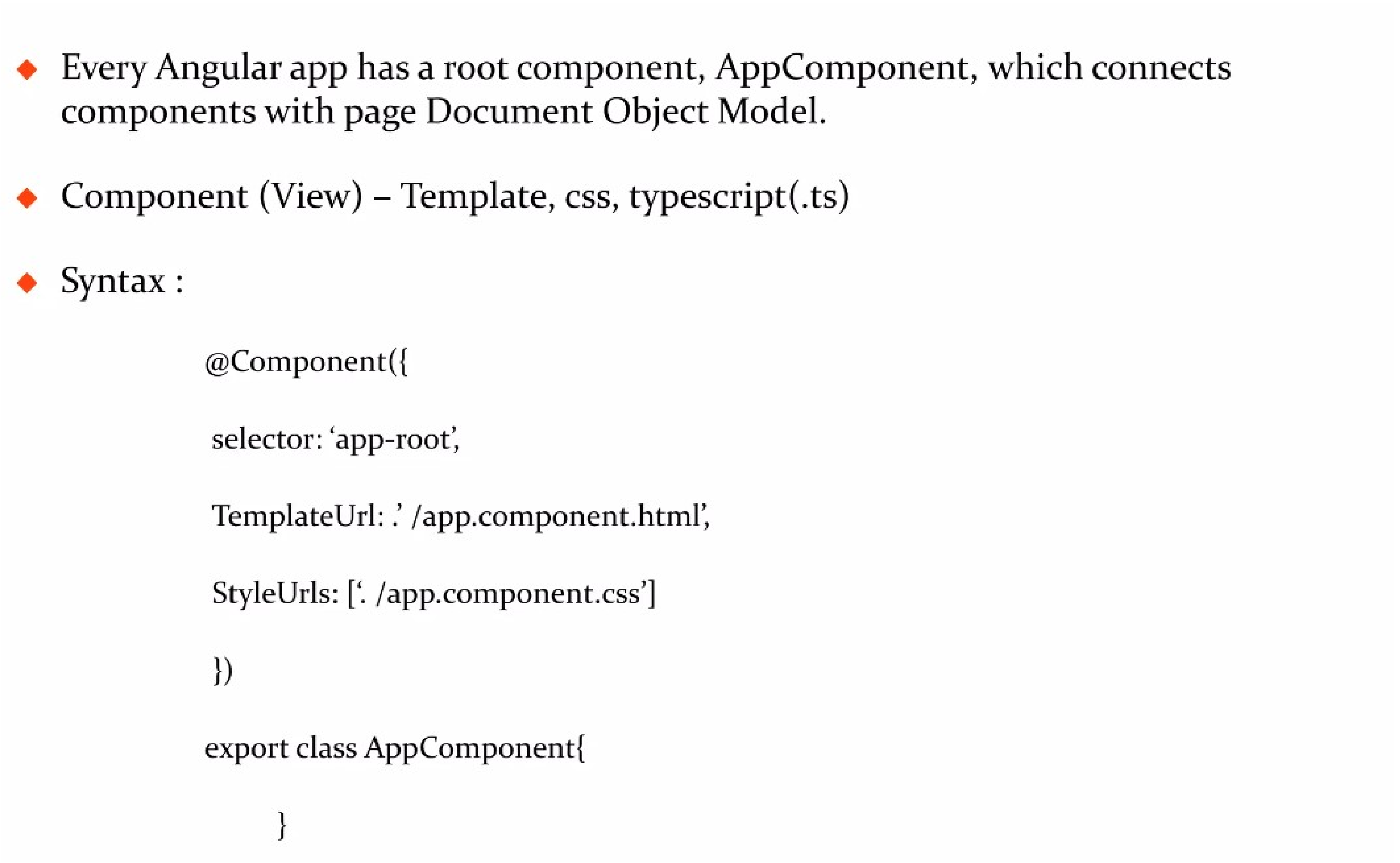
Module



- Every Angular app has a root module, conventionally named AppModule, which provides the bootstrap mechanism that launches the application.
- An app typically contains many functional modules.
- Syntax :

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

Component and metadata



- Every Angular app has a root component, AppComponent, which connects components with page Document Object Model.
- Component (View) – Template, css, typescript(.ts)
- Syntax :

```
@Component({

 selector: ‘app-root’,

 TemplateUrl: .’ /app.component.html’,

 StyleUrls: [‘. /app.component.css’]

 })

export class AppComponent{

        }
```

Template and DataBinding

- The angular template integrates the HTML with Angular mark-up that can modify HTML elements before they are displayed.
- It provides program logic, and binding mark-up connects to your application data and the DOM.
- **Data-Binding :**
  - **Property Binding** (One way and Two way)

    e.g. <p>Name: {{student.name}} </p>

    e.g. <input type="text" [value]="title"/>

    e.g. <input [(ngModel)]="title" placeholder="name"/>
  - **Event binding**

    <button (click)="changeTitle()">Click to update title</button>

Directives

- There are three types of Directives(modified DOM) :

1. Components directives: directives with a template

2. Structural directives : change the DOM layout by adding and removing DOM elements

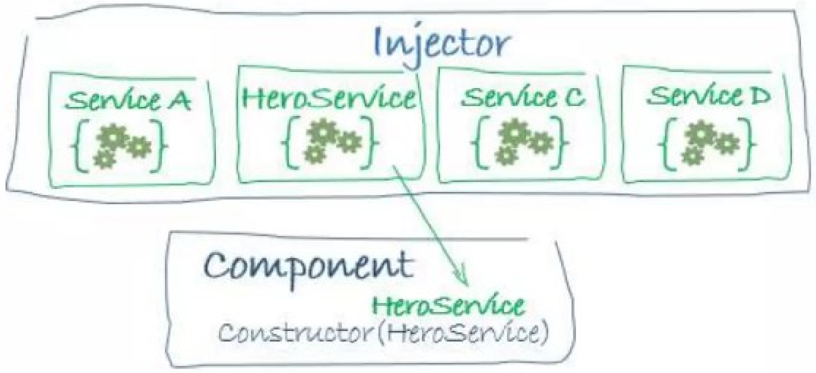
   For example: *ngIf, *ngFor, and *ngSwitch directive.

3. Attribute directives : change the appearance or behavior of an element, component, or another directive.
   For example: ngClass, ngStyle etc.

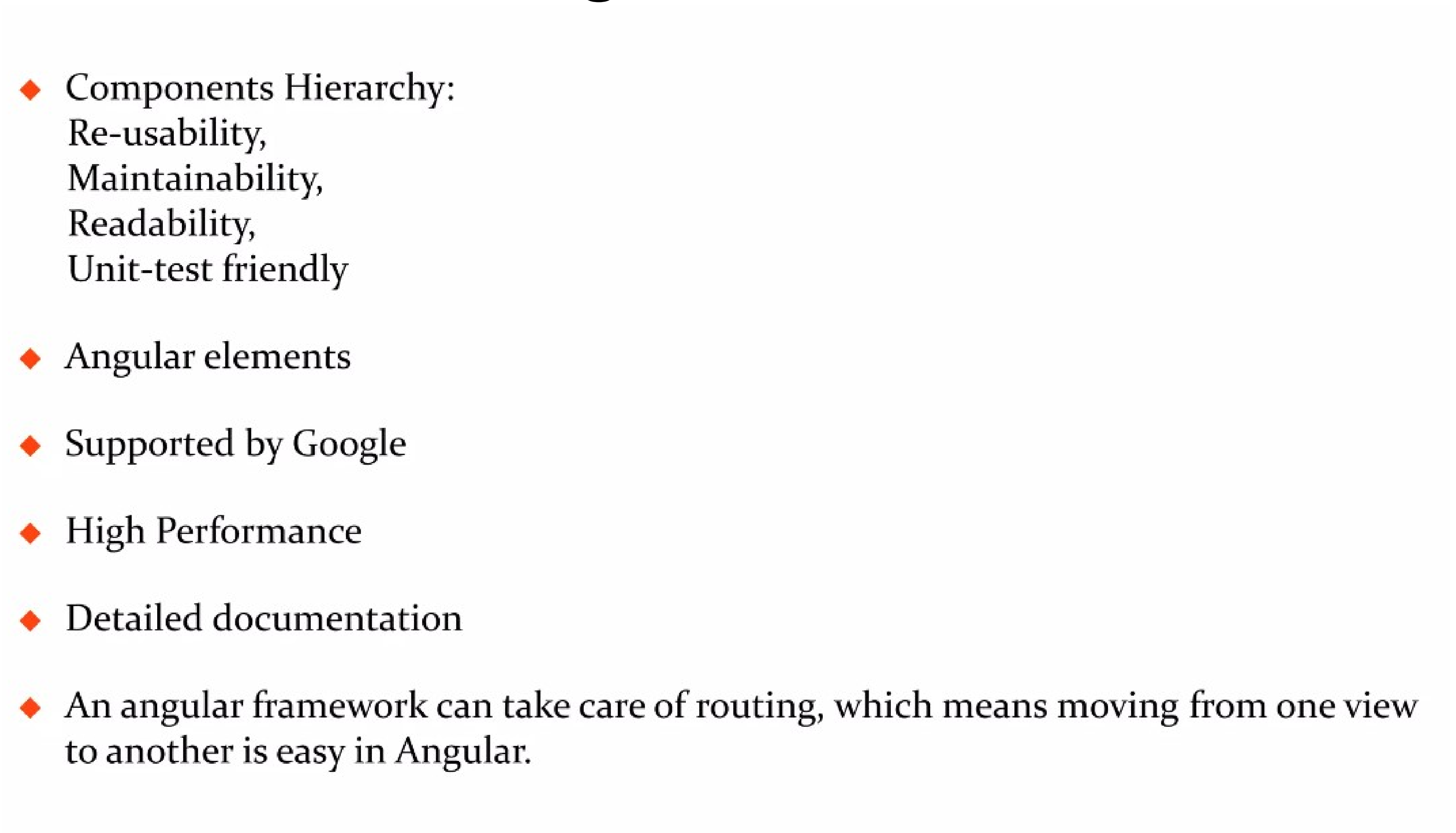
```
<p [ngClass]="{'className': country === 'India'}> {{Name}}</p

<p [ngStyle]="{'background-color':country === 'India' ? 'red' : 'green' }">
    {{Name}}</p
```

Service and dependancy Injection (DI)

- Service : any value, function or feature that an app needs.
- Angular distinguishes components from services to increase modularity and reusability.
- A component can delegate certain tasks to services, such as fetching data from the server, validating user input, or logging directly to the console.
- Angular creates an application-wide injector for you during the bootstrap process.



Pros of Angular



- Components Hierarchy:
  Re-usability,
  Maintainability,
  Readability,
  Unit-test friendly
- Angular elements
- Supported by Google
- High Performance
- Detailed documentation
- An angular framework can take care of routing, which means moving from one view to another is easy in Angular.