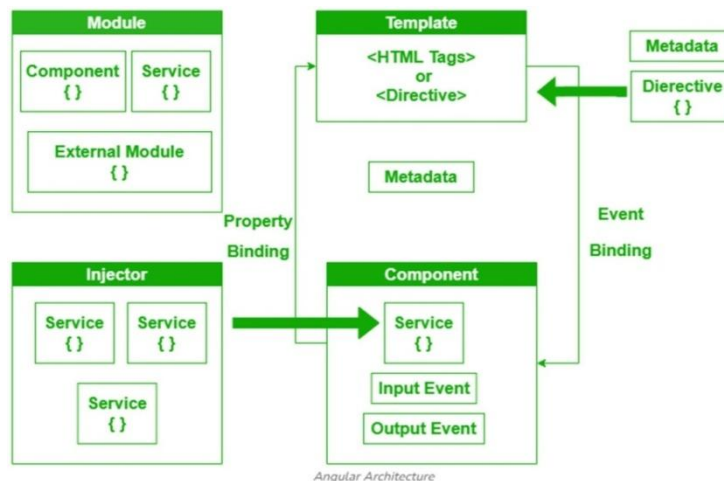


What is Angular?

- **Angular** is an open source framework and development platform used to build Web, Window and Mobile Application.
- Angular is built on Type Script (a superset of JavaScript).
- **Angular** was build and owned by Google although Microsoft was also involved in Angular Development.
- Angular provides tools and libraries for **routing, forms, HTTP client, animations, testing, and more.**
- It is **Frontend Framework** to create SPA (Single Page Application).



Angular Versions



Characteristics of Angular:

- **Component-based architecture:** Applications are built from reusable, encapsulated components that manage their own logic and views.
- **TypeScript foundation:** Strong typing, decorators, and modern JavaScript features improve maintainability and tooling.
- **Dependency injection:** Services can be shared across components, making code modular and testable.
- **Routing system:** Built-in navigation with route guards, lazy loading, and data resolution.
- **Reactive forms:** Standardized system for form handling and validation.
- **Server-side rendering:** Supports SSR and static site generation for faster performance and SEO.
- **CLI tools:** Automates project setup, builds, testing, and updates.
- **DevTools:** Debugging and performance profiling integrated with browser developer tools.

Concepts of Angular

1. Component-based Architecture

- Angular applications are built using components.
- A component is like a building block of UI.
- Each component has:
 - **Template (HTML)** → view
 - **Styles (CSS)** → design
 - **Class (TS)** → logic

2. Modules

- Angular apps are divided into **modules** for better organization.
- The main module is AppModule.
- Modules group components, services, and directives together.

3. Templates

- Templates define **how data is displayed on the screen**.
- Angular uses HTML + Angular syntax ({{ }} for data binding).

```
<h1>Welcome {{studentName}}</h1>
```

4. Data Binding

- Connects **TypeScript code (logic)** with **HTML (view)**.
- Types:
 - Interpolation {{ }}
 - Property Binding []
 - Event Binding ()
 - Two-way Binding [()]

5. Directives

- Special instructions in HTML to change the behavior of DOM.
- Types:
 - **Structural** (*ngIf, *ngFor)
 - **Attribute** (ngStyle, ngClass)

6. Services and Dependency Injection (DI)

- Services are used to **share data or logic** across components.
- Dependency Injection allows reusing services easily.

7. Routing

- Used to navigate between different views/pages of an application without refreshing the page.
- Example: /home, /about

8. Single Page Application (SPA)

- Angular apps work as SPA.
- Only **one HTML page loads**, and content updates dynamically using JavaScript.
- No full-page reloads, making the app faster.

Angular TypeScript:

- Most importantly you must know that Angular Code is written in **TypeScript**.

- Superset of JavaScript with static typing, interfaces, and classes.
- **TypeScript** is object oriented programming language which is developed by Microsoft. Typescript is the superset of JavaScript ES6. TypeScript code is converted into JavaScript by TypeScript Transpiler. Although we can also use plain Vanilla JS ES6 in TypeScript.

JavaScript ES6 + Types = TypeScript

Dependencies in an Angular + TypeScript App

- @angular/core – Core Angular framework
- @angular/forms – Template-driven & reactive forms
- @angular/router – Routing and navigation
- Example: TypeScript in Angular

Example: Write a Hello World! program in Angular.

app.ts

```
import { Component } from '@angular/core';
import { RouterOutlet } from '@angular/router';
@Component({
  selector: 'app-root',
  imports: [RouterOutlet],
  templateUrl: './app.html',
  styleUrls: ['./app.css']
})
export class AppComponent {
  title = 'Hello World App';
}
```

app.html

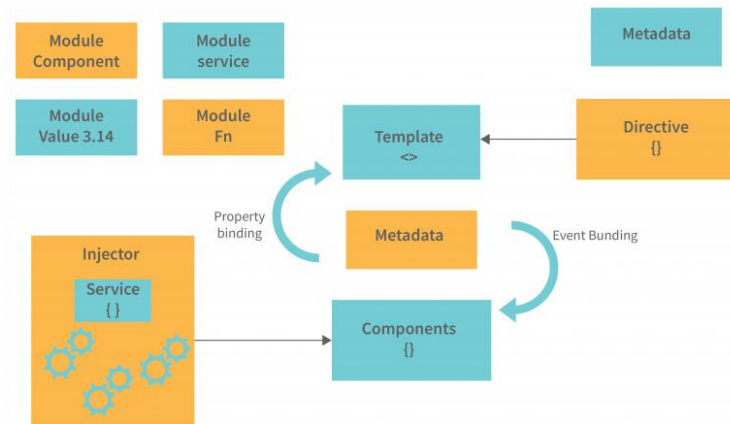
```
<h3>{{title}}</h3>
<p>Hello World!</p>
```

Output:



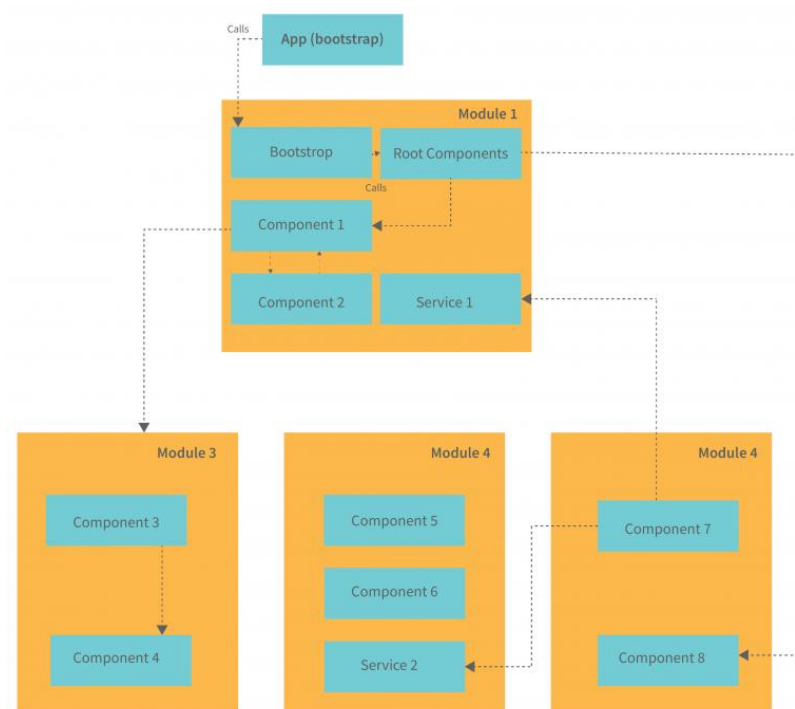
Component-based Architecture/Angular Architecture

Angular is an MVC framework that has detailed diagrams on how to create the application and how to pass data between the view and the controller. It also provides bi-directional data flow so that the markup is not altered.



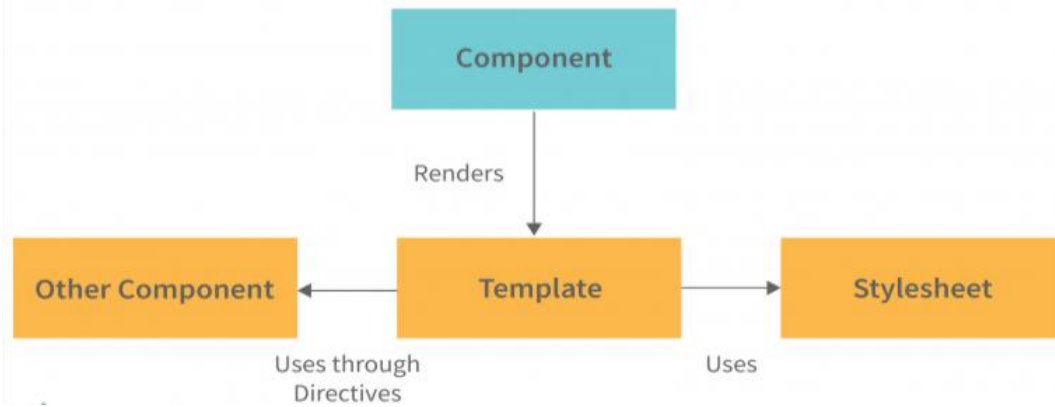
An Angular application is composed of the following eight building blocks:

Modules:



Component

- Every Angular application, consisting of one or more Angular Components, is built from Components. Angular Components are simple JavaScript/Typescript classes with HTML templates and names.
- A component's HTML template may access data from its corresponding JavaScript / Typescript class or component if it has a selector's value (name). Angular components may have CSS styles associated with them and the HTML template may access them.



Templates

- The template is an HTML super set. It includes all the features of HTML, as well as additional functionality that binds component data into the HTML and generates HTML DOM elements dynamically.
- Event binding allows your app to **respond to user actions (events)** such as clicks, key presses, mouse movements, etc.

```

<button(click)="onClick()">ClickMe</button>
onClick(){
  alert('Button clicked!');
}

```

- Used to bind values from **component class** → **DOM element properties**.

```

<img[src]="imageUrl"[alt]="imageAlt">
imageUrl='assets/angular-logo.png';
imageAlt = 'Angular Logo';

```

- Embeds **component data in to HTML** by using `{{ }}`.

```

<p>Hello, {{username}}!</p>
username='Ami';

```

Basic Angular development environment setup and project workflow.

1. Setting Up the Development Environment

To develop Angular applications, you generally need:

- Node.js** – provides the JavaScript runtime and npm package manager.
- npm** – installs Angular and other project dependencies.
- Angular CLI** – command-line tool for creating, developing, testing, and building Angular applications.
- Code editor** – commonly Visual Studio Code.
- Web browser** – Chrome, Edge, Firefox, etc.

Verify Node.js and npm:

```

node --version
npm --version

```

2. Installing Node.js and Angular CLI

- Install the current suitable **LTS version of Node.js** from the official Node.js website:
- Then install Angular CLI globally:

```
npm install -g @angular/cli
```

Verify the installation:

```
ng version
```

3. Creating an Angular Project

- Create a new project with:

```
ng new my-angular-app
```

The CLI will ask configuration questions such as whether to use routing and which stylesheet format to use.

Move into the project directory:

```
cd my-angular-app
```

4. Running an Angular Project

Start the development server:

```
ng serve  
Or  
npm start
```

Angular normally makes the application available at: `http://localhost:4200/`

You can also automatically open the browser: `ng serve --open`

5. Angular Project Structure

A newly created Angular project contains several important files and directories:

```
my-angular-app/  
├── node_modules/  
├── public/  
├── src/  
│   ├── app/  
│   ├── index.html  
│   ├── main.ts  
│   └── styles.css  
├── angular.json  
├── package.json  
├── package-lock.json  
├── tsconfig.json  
└── README.md
```

Important parts:

- src/ – contains the application's source code.
- src/app/ – contains Angular application code such as components, services, and other application features.
- main.ts – entry point used to bootstrap the Angular application.
- index.html – main HTML page loaded by the browser.
- styles.css – global styles.
- angular.json – Angular workspace/project configuration.
- package.json – lists dependencies, scripts, and project metadata.
- node_modules/ – contains installed npm packages.
- tsconfig.json – TypeScript compiler configuration.

6. Angular Workspace

An **Angular workspace** is the overall development environment managed by Angular CLI. It contains configuration and one or more Angular projects.

A workspace can contain:

Angular Workspace

```
|  
├── Project 1  
├── Project 2  
├── Shared configuration  
└── Workspace configuration
```

For a simple application, the workspace and application are usually created together with:

```
ng new my-angular-app
```