ANGULAR

**Introduction to Angular :**

Angular is a development platform, built on [TypeScript](https://www.typescriptlang.org/). As a platform, Angular includes:

* A component-based framework for building scalable web applications
* A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
* A suite of developer tools to help you develop, build, test, and update your code

**Components**

Components are the building blocks that compose an application. A component includes a TypeScript class with a @[Component](https://angular.io/api/core/Component)() decorator, an HTML template, and styles. The @[Component](https://angular.io/api/core/Component)() decorator specifies the following Angular-specific information:

* A CSS selector that defines how the component is used in a template. HTML elements in your template that match this selector become instances of the component.
* An HTML template that instructs Angular how to render the component.
* An optional set of CSS styles that define the appearance of the template's HTML elements.

Ex: -> The following is a minimal Angular component :

import { [Component](https://angular.io/api/core/Component) } from '@angular/core';

@[Component](https://angular.io/api/core/Component)({

selector: 'hello-world',

template: ` <h2>Hello World</h2> <p>This is my first component!</p> `,

})

export class HelloWorldComponent {

// The code in this class drives the component's behavior.

}

-> To use this component, you write the following in a template:

Syntax : <hello-world></hello-world>

-> When Angular renders this component, the resulting DOM looks like this:

<hello-world>

<h2>Hello World</h2>

<p>This is my first component!</p>

</hello-world>

### Templates

Every component has an HTML template that declares how that component renders. You define this template either inline or by file path.

-> One application of this feature is inserting dynamic text, as shown in the following example :

<p>{{ message }}</p>

-> The value for message comes from the component class:

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

@Component ({

selector: 'hello-world-interpolation',

templateUrl: './hello-world-interpolation.component.html'

})

export class HelloWorldInterpolationComponent {

message = 'Hello, World!';

}

-> When the application loads the component and its template, the user sees the following:

<p>Hello, World!</p>

-> Angular also supports property bindings, to help you set values for properties and attributes of HTML elements and pass values to your application's presentation logic.

<p [id]="sayHelloId" [style.color]="fontColor">You can set my color in the component!</p>

-> You can also declare event listeners to listen for and respond to user actions such as keystrokes, mouse movements, clicks, and touches. You declare an event listener by specifying the event name in parentheses:

<button (click)="sayMessage()" [disabled]="canClick">Trigger alert message</button>

**Angular CLI :**

The Angular CLI is the fastest, easiest, and recommended way to develop Angular applications. The Angular CLI makes a number of tasks easy. Here are some examples:

ng build - Compiles an Angular app into an output directory.

ng serve - Builds and serves your application, rebuilding on file changes.

ng generate - Generates or modifies files based on a schematic

ng test - Runs unit tests on a given projecting

ng e2e - Builds and serves an Angular application, then runs end-to-end tests.