1.0) AIM: Understanding one way data binding with Property and its attribute binding in Angular

2.0) SCOPE:

To demonstrate the concept of property binding in Angular by creating a simple web application that dynamically updates the DOM elements based on the data properties in the Angular component. Property binding in Angular helps to set values for properties of HTML elements or directives. Use property binding to do things such as toggle button features, set paths programmatically, and share values between components.

3.0) FACILITIES/ APPARATUS:

i)Hardware: I3 Processor, 8GB RAM, HD Monitor, Windows 10, Internet connectivity

ii) Software: Updated Web browser, Node js, Angular CLI, VS Code.

4.0) THEORY:

Property binding in Angular is a mechanism to bind the component's data properties to the HTML element properties in the view. It allows data to flow from the component's model to the view. Property binding is denoted using square brackets [] around the HTML element's property.

For example:

<input [value]="dataProperty">

In this example, the value attribute of the <input> element is bound to the data Property in the Angular component. Any change in dataProperty will automatically update the value attribute in the DOM

1. Set up the Angular Project:

Create a new Angular project using Angular CLI:

ng new property-binding-demo

Navigate to the project directory:

cd property-binding-demo

2. Create a Component for the Experiment:

Generate a new component using Angular CLI:

ng generate component databinding

This command will create a new component with the necessary files:

- databinding.ts, databinding.html, and databinding.css.

3. Implement Property Binding in the Component:

- Open the property-binding-example.component.ts file in your text editor.
- Define a property in the PropertyBindingExampleComponent class:

```
import { Component } from '@angular/core';
import { RouterOutlet } from '@angular/router';

@Component
({
  selector: 'app-property-binding-example',
  standalone: true,
  imports: [RouterOutlet],
  templateUrl: './property-binding-example.component.html',
  styleUrls: ['./property-binding-example.component.css']
})
export class PropertyBindingExampleComponent
{
courseName: string = "Angular 18";
  inputType = "radio";
  myClassName: string = "bg-primary";

}
```

4. Data Binding in the HTML Template:

Open the property-binding-example.component.html file. use property binding to display the title property in a heading and bind it to an input element:

```
<h2>Data Binding in Angular 20</h2>

<!-- INTERPOLATION -->
<p>Welcome, {{ userName }}!</p>
<p>2 + 2 = {{ 2 + 2 }}</p>

<!-- PROPERTY BINDING -->
<img [src]="imageUrl" [alt]="userName" width="200" />
<br /><br />

<!-- EVENT BINDING -->
<button (click)="changeName()">Change Name</button>
<button (click)="changeImage()">Change Image</button>

<!-- INPUT with EVENT + PROPERTY BINDING -->
<input
  [value]="userName"
  (input)="onNameInput($event)"
  placeholder="Type your name"
/>
<p>Updated Name: {{ userName }}</p>
```

The [innerText] binds the title property to the h1 tag's text content, and [value] binds the same property to the input element's value.

5.Updations in .ts File of component

```
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './app.component.html'
})
export class AppComponent {
  userName = 'Angular Developer';
  imageUrl = 'https://angular.io/assets/images/logos/angular/angular.png';

  // EVENT BINDING METHODS
  changeName() {
    this.userName = 'Super Coder 🚀';
  }

  changeImage() {
    this.imageUrl = 'https://upload.wikimedia.org/wikipedia/commons/6/6a/JavaScript-logo.png';
  }

  onNameInput(event: Event) {
    const target = event.target as HTMLInputElement;
    this.userName = target.value;
  }
}
```

6. Update the App Component:

Open the app.component.html file.Replace its content with the following to include the newly created component:

<app-databinding></app-databinding>

7. Run the Angular Application

ng serve

Open a web browser and navigate to http://localhost:4200 .You should see the heading and input field displaying the text "Welcome to Angular Property Binding!".

Conclusion

The experiment successfully demonstrated the concept of data binding in Angular. By binding component properties to DOM element properties, we can dynamically update the user interface based on the component's data model, which is a powerful feature for building interactive web applications.