Data Binding: Class and Style Binding

In Angular, class and style bindings allow you to dynamically set CSS classes and inline styles on elements based on component data or conditions. Here's how you can use them:

1. Class Binding

Class binding allows you to add or remove CSS classes conditionally.

Syntax:

<div [class.class-name]="condition">

- class-name: The CSS class you want to add or remove.
- condition: A boolean expression. If true, the class is added; if false, the class is removed.

Example:

<div [class.active]="isActive">This div is active

Here, the active class is applied to the div if isActive is true.

2. Multiple Class Binding

You can bind multiple classes conditionally using [ngClass].

Syntax:

<div [ngClass]="{'class1': condition1, 'class2': condition2}">

- 'class1': condition1: Adds class1 if condition1 is true.
- 'class2': condition2: Adds class2 if condition2 is true.

Example:

<div [ngClass]="{'active': isActive, 'disabled': isDisabled}">This div has dynamic classes

3. Style Binding

Style binding allows you to set inline styles dynamically.

Syntax:

<div [style.style-name]="expression">

- style-name: The name of the CSS style property you want to set.
- expression: The value to assign to the style property.

Example:

<div [style.background-color]="isRed ? 'red' : 'blue'">This div has a dynamic background color

Here, the background color is set to red if isRed is true, otherwise it's set to blue.

4. Multiple Style Binding

You can bind multiple styles at once using [ngStyle].

Syntax:

<div [ngStyle]="{'style1': value1, 'style2': value2}">

- 'style1': value1: Sets style1 to value1.
- 'style2': value2: Sets style2 to value2.

Example:

<div [ngStyle]="{'color': textColor, 'font-size': fontSize + 'px'}">This div has dynamic styles

Here, color and font-size are set based on the component properties textColor and fontSize.

Example Component:

Here's a simple Angular component that demonstrates both class and style bindings:

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


@Component({
  selector: 'app-example',
  template: `
    <div [class.active]="isActive">Active Class Binding</div>
    <div [ngClass]="{'red': isRed, 'bold': isBold}">Multiple Class Binding</div>
    <div [style.color]="textColor">Style Binding</div>
    <div [ngStyle]="{'background-color': bgColor, 'font-size': fontSize + 'px'}">Multiple Style Binding</div>
  `,
  styles: [`
    .active { font-weight: bold; }
    .red { color: red; }
    .bold { font-weight: bold; }
  `]
})
export class ExampleComponent {
  isActive = true;
  isRed = true;
  isBold = false;
```

```
  textColor = 'blue';
  bgColor = 'yellow';
  fontSize = 16;
}
```

In this example, the div elements will have their classes and styles dynamically applied based on the component's properties.

Angular Lab Exercise: Class Binding and Style Binding

Objective:

To practice and understand the concepts of class binding and style binding in Angular by creating a dynamic UI that responds to user interactions.

Prerequisites:

- Basic knowledge of Angular components.
- Understanding of Angular templates and data binding.
- Angular CLI installed.

Exercise Overview:

In this exercise, you will create an Angular component that allows users to dynamically change the appearance of elements on the page by toggling CSS classes and styles using Angular's class binding and style binding features.

Part 1: Setup

1. **Create a new Angular project (if not already created):**

```
ng new angular-binding-exercise
cd angular-binding-exercise
```

2. **Generate a new component:**

```
ng generate component dynamic-styles
```

3. **Open the project in your code editor:**

Part 2: Implementing Class Binding

1. **Add Component Template:**

Open src/app/dynamic-styles/dynamic-styles.component.html and replace the content with the following:

```
<div>
  <h2>Class Binding Example</h2>
```

```
  <button (click)="toggleActive()">Toggle Active Class</button>
  <button (click)="toggleBold()">Toggle Bold Class</button>
</div>

<div [class.active]="isActive" [class.bold]="isBold">
  This div will change classes dynamically!
</div>
```

2. **Add Component Logic:**

Open src/app/dynamic-styles/dynamic-styles.component.ts and add the following logic:

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

@Component({
  selector: 'app-dynamic-styles',
  templateUrl: './dynamic-styles.component.html',
  styleUrls: ['./dynamic-styles.component.css']
})
export class DynamicStylesComponent {
  isActive = false;
  isBold = false;

  toggleActive() {
    this.isActive = !this.isActive;
  }

  toggleBold() {
    this.isBold = !this.isBold;
  }
}
```

3. **Add CSS Classes:**

Open src/app/dynamic-styles/dynamic-styles.component.css and add the following styles:

```
.active {
```

```
  background-color: lightblue;
}

.bold {
  font-weight: bold;
}

div {
  margin: 10px 0;
  padding: 10px;
  border: 1px solid #ccc;
}
```

4. **Explanation:**
    - The toggleActive and toggleBold methods toggle the isActive and isBold properties between true and false.
    - The div element's classes (active and bold) are conditionally applied based on these properties.

Part 3: Implementing Style Binding

1. **Add Additional Template:**

Extend the template in dynamic-styles.component.html:

```
<div>
  <h2>Style Binding Example</h2>
  <button (click)="changeColor()">Change Color</button>
  <button (click)="changeFontSize()">Change Font Size</button>
</div>

<div [style.color]="textColor" [style.font-size.px]="fontSize">
  This div will change styles dynamically!
</div>
```

2. **Add Component Logic:**

Extend the component logic in dynamic-styles.component.ts:

```
textColor = 'black';
```

```
fontSize = 14;


changeColor() {
  this.textColor = this.textColor === 'black' ? 'red' : 'black';
}


changeFontSize() {
  this.fontSize = this.fontSize === 14 ? 20 : 14;
}
```

3. **Explanation:**
    - The changeColor method toggles the textColor property between 'black' and 'red'.
    - The changeFontSize method toggles the fontSize property between 14px and 20px.
    - The div element's color and font-size styles are dynamically updated based on these properties.

Part 4: Running the Application

1. **Serve the Angular Application:**

```
ng serve
```

2. **Interact with the Application:**
    - Open a browser and navigate to http://localhost:4200.
    - You should see the buttons and div elements as defined.
    - Click the buttons to see how the classes and styles change dynamically.

Part 5: Challenge

Extend the exercise by adding more dynamic styles or classes:

- Create a text box to allow the user to input a color or font size and apply it to the div.
- Add a class that makes the div italic and allow the user to toggle it.
- Bind more CSS properties like background-color, margin, etc.

Submission:

Once you have completed the exercise, submit the code or a link to your repository for review. Make sure your application is working as expected and covers all the required features.

This lab exercise should help you understand and practice class binding and style binding in Angular, making your components more dynamic and interactive.