

1604C054 – Hybrid Mobile Programming

# Angular Binding and Angular Directive Part 2

**WEEK 04**

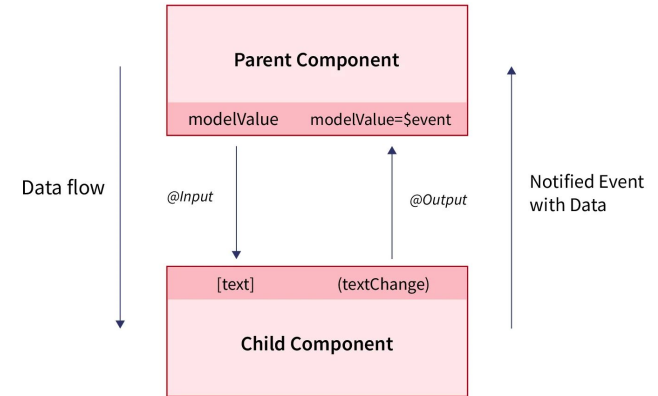
Informatics Engineering  
Universitas Surabaya



# Two Way Binding

# What is Two Way Binding?

- Two-way data binding is a concept in Angular that allows you to establish a synchronization between a property in your component class and an input element's value in your template.
- It enables changes made to the input element in the template to automatically update the corresponding property in your component class, and vice versa.
- Two-way binding simplifies the process of capturing and reflecting user input in real-time.



## What is Two Way Binding? (2)

In Angular, two-way data binding is typically achieved using the **ngModel** directive, which combines property binding and event binding to create a seamless bidirectional data flow.

example:

```
<input [(ngModel)]="username" />
```

- [(ngModel)] is the two-way binding syntax.
- "username" is the component property (in the component class) that you want to bind to the input element's value.
- There is variable in page.ts → `username: string = "`

*Text in Input element will be changed if username variable change. conversely, username variable value will be updated when user write any text in input element.*

## IMPORTANT !!

Two way binding must import **FormsModule** from `@angular/forms` and add it to the imports array of your Angular module .

Please check your `list.module.ts` .

- The `FormsModule` import now is a default import when we generate page.
- Therefore we not need to add anything here and we can directly use **ngModel**.

# IMPLEMENTATION

We will add a text input for user fill coupon code. The text input value will relate with couponcode class property.

1. Set default value for the coupon code as, for example, '0000' this value will reflect in the text input component. This is the binding from class property to input component.
2. We will modify the text input value. It will reflect automatically in the couponcode variable. We will add interpolation binding to check the value of couponcode variable. And this is proving the second binding, from input component to class property.


Add couponcode variable in list.page.ts :

```
couponcode:string="0000"  
strvalid:string="Invalid"  
discount:number=0
```

# IMPLEMENTATION




Add input text and interpolation binding for checking in list.page.html :

```
<ion-list lines="full">
  <ion-item>
    <ion-input labelPlacement="floating" label="Masukkan kode : " [(ngModel)]="couponcode"> </ion-input>
  </ion-item>
  <ion-item>
    <ion-text>Coupon code {{couponcode}} is {{strvalid}} you get {{discount}}% discount</ion-text>
  </ion-item>
</ion-list>
```

 List

Masukkan kode :  
0000

Coupon code 0000 is Invalid you get 0% discount

Product List per Sunday, 15 September 2024   

## Self Practice 1 (Max 10 Minutes)

If (**keyup**) is an event binding after that raise after user press a character in keyboard, can you put function **checkValid** that validate the couponcode in hardcoded (next, it should check in the database) and change the response text below:

- If couponcode is '1234' the text shows :
  - **Coupon code 1234 is valid. you get 5% discount.**
- If couponcode is '6789' the text shows :
  - **Coupon code 6789 is valid. you get 10% discount.**

Masukkan kode :

1234



Coupon code 1234 is valid you get 5% discount

Masukkan kode :

6789



Coupon code 6789 is valid you get 10% discount



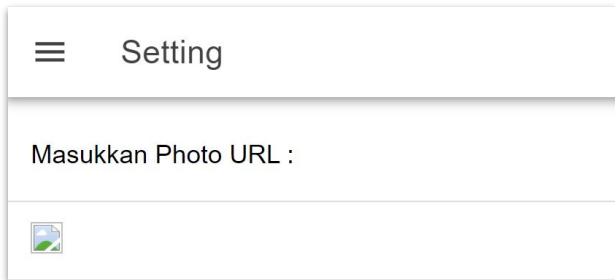
## Self Practice 2 (Max 10 Minutes)

Use setting page. The Setting page will be use as setting out profile. It can store our profile photo.

Question :

Can you link a text input and an image, when user put url of a photo in internet, this url will be use in image src and will be shown directly.

hint : `<ion-img></ion-img>`



## Self Practice 3 (Max 20 Minutes) → 8.30

Still use setting page. Setting page will also be for set up our password.

If **checked** is a property that define a checkbox is checked or not, can you create a **Password strength** checkers ?

What to be check is :

- ☐ More than 6 characters
- ☐ Include number
- ☐ Include special character (!@#\$\$%^&\*)

Masukkan Password :

hmp1seasy



More than 6 Characters



Include Number



Include Special Characters (!@#\$\$%^&\*)



Masukkan Password :

hmp1seasy###



More than 6 Characters



Include Number



Include Special Characters (!@#\$\$%^&\*)



***Each checkbox above will be checked if user input is satisfied the related criteria***

# Angular Directives

The **ngStyle** directive in Angular allows you to dynamically apply inline CSS styles to HTML elements based on expressions in your component. It's a powerful way to style elements based on component data or conditions.

## Example

```
<div [ngStyle]="{ 'font-size.px': fontSize, 'color': textColor }">Styled Text</div>
```



fontSize and textColor is class properties

## ngStyle Example

We will set color of responded coupon code text ("Coupon code xxxx is yyyy") to become red when the code is Invalid, and green when valid.

Add ngStyle in the text. Because it doesn't placed in any element, we have to put it in element, like <div> or <span> or <p>

```
<span [ngStyle]="{'color': textcolor }">  
  Coupon code {{couponcode}} is {{strvalid}} ...  
</span>
```

Add class properties : `textcolor:string="red"`

Masukkan kode :

0000

Coupon code 0000 is Invalid you get 0% discount

## Self Practice 4 (Max 5 Minutes)

Modify your **checkValid()** function in exercise 3, so it will set textcolor to **red** when invalid and **green** when valid

Masukkan kode :

0000

Coupon code 0000 is Invalid you get 0% discount



Masukkan kode :

1234

Coupon code 1234 is valid you get 5% discount

The `ngIf` directive in Angular is a structural directive that allows you to **conditionally render or remove** elements from the DOM (Document Object Model) based on the evaluation of a provided expression. It's a fundamental directive for controlling the **visibility and existence** of HTML elements in your templates.

example:

```
<div *ngIf="isVisible">Cilukba</div>
```

There is `isVisible` class property.

- If **isVisible** variable is **True**, the div will appear and text "Cilukba" will be visible
- If **isVisible** variable is **False** the div will be not rendered and user can not see "Cilukba" text.

## ngIf Example

Example:

We will add congratulation animated gif image when the couponcode inputted by user is valid.

Add in list.page.html :

```
<ion-img *ngIf="strvalid=='valid'"  
src="https://www.animatedimages.org  
/data/media/1103/animated-congratula  
tion-image-0092.gif"></ion-img>
```

Masukkan kode :

1234

Coupon code 1234 is valid you get 5% discount





# ngFor

The **\*ngFor** directive in Angular is a structural directive used for iterating over a collection, such as an array, and rendering HTML elements for each item in the collection. It allows you to create repeated content dynamically in your templates.

Example:

```
<div *ngFor="let item of items; let i = index">  
  <!-- Content to repeat for each item -->  
</div>
```

let item of items:

let item is a local template variable that represents the current item in the iteration. You can name this variable whatever you like; in this case, it's named item.

of items specifies the collection (array or iterable) over which you want to iterate. items is the name of the array or iterable in your component that you want to loop through.

let i = index (Optional):

let i is an optional local variable that represents the current index of the item in the iteration.

= index assigns the index value to the i variable. You can use i to track the index of each item in the collection.

## ngFor Example

We will show books data. Books is an array of objects. Objects here is still 'ugly', we not define it in class or interface in order to shorten time.

Add in list.page.ts



```
books = [  
  {  
    title: 'To Kill a Mockingbird',  
    author: 'Harper Lee',  
    publishedDate: new Date('1960-07-11'),  
    price: 7.99  
  },  
  {  
    title: 'The Great Gatsby',  
    author: 'F. Scott Fitzgerald',  
    publishedDate: new Date('1925-04-10'),  
    price: 10.99  
  },  
  {  
    title: 'Pride and Prejudice',  
    author: 'Jane Austen',  
    publishedDate: new Date('1813-01-28'),  
    price: 12.75  
  }  
]
```

## ngFor Example (2)

In list.page.html :

```
<ion-list *ngFor="let book of books">
  <ion-list-header>
    <ion-label><h1>{{book.title}}</h1></ion-label>
  </ion-list-header>
  <ion-item>
    <ion-label>Author : {{book.author}}</ion-label>
  </ion-item>
  <ion-item>
    <ion-label>Price : ${{book.price}}</ion-label>
  </ion-item>
</ion-list>
```

Product List per Sunday, 15 September 2024	
To Kill a Mockingbird	
Author : Harper Lee	
Price : \$7.99	
The Great Gatsby	
Author : F. Scott Fitzgerald	
Price : \$10.99	
Pride and Prejudice	
Author : Jane Austen	
Price : \$12.75	

## Exercise 1 (Max 15 Minutes)

Add discount in every book object.  
Show the discount after book  
name.

If there is discount, **Strikeout** the  
Price text and add Price after  
discount below it.

### To Kill a Mockingbird

Author : Harper Lee

Discount : 10%

~~Price : \$7.99~~

Price : \$7.19

### The Great Gatsby

Author : F. Scott Fitzgerald

Discount : 5%

~~Price : \$10.99~~

Price : \$10.44

### Pride and Prejudice

Author : Jane Austen

Discount : 15%

~~Price : \$12.75~~

## Exercise 2 (Max 10 Minutes)

Apply more discount when user input the Valid couponcode.

Strikeout again the price with discount in the item, and add new price that apply discount from item and discount from couponcode.

**Please submit the project you worked on class  
(Self Practice 1 – 4 and Exercise 1 – 2) → ULS**

# Thanks.