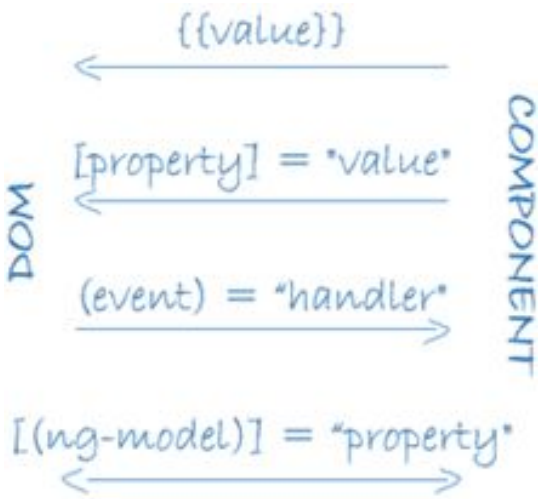
Angular Data Binding

What and Why Data Binding?



Tanpa adanya framework, kita akan bertanggungjawab secara manual untuk mendefinisikan value ke html, membuat kontrol, menjadikan response dari user sebagai aksi untuk mengupdate value di html. Menuliskan semua kode ini, tentunya akan membuat kode susah dibaca dan memperbesar tendensi error

Data binding akan membuat halaman otomatis terupdate berdasar state aplikasi. Penggunaaan data binding bisa digunakan untuk menentukan image source, menentukan state dari suatu button, dll.

Tipe dan Target Binding

Type	Target	Examples

Property	Element property
	Component property
	Directive property
		`src`, `hero`, and `ngClass` in the following:
		`<img [src]="heroImageUrl">`
		`<app-hero-detail [hero]="currentHero"></app-hero-detail>`
		`<div [ngClass]="{'special': isSpecial}"></div>`

Event	Element event
	Component event
	Directive event
		`click`, `deleteRequest`, and `myClick` in the following:
		`<button (click)="onSave()">Save</button>`
		`<app-hero-detail (deleteRequest)="deleteHero()"></app-hero-detail>`
		`<div (myClick)="clicked=$event" clickable>click me</div>`

Two-way	Event and property	`<input [(ngModel)]="name">`

Type	Target	Examples
Attribute	Attribute (the exception)	`<button [attr.aria-label]="help">help</button>`
Class	`class` property	`<div [class.special]="isSpecial">Special</div>`
Style	`style` property	`<button [style.color]="isSpecial ? 'red' : 'green'">`

Referensi: https://angular.io/guide/binding-syntax

Macam-macam Data Binding

Type	Syntax	Category

Interpolation
Property
Attribute
Class
Style
	`{{expression}}`
	`[target]="expression"`
	`bind-target="expression"`
		One-way
		from data source
		to view target

Event	`(target)="statement"` `on-target="statement"`	One-way from view target to data source
Two-way	`[(target)]="expression"` `bindon-target="expression"`	Two-way

Referensi: https://angular.io/guide/binding-syntax

Text Interpolation

Interpolasi berfungsi untuk meng-embed ekspresi/value yang dinamis dalam template html.

Dibungkus dengan {{ your_value_here }} Sebagai delimiter.

app.component.ts

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css'],
})
export class AppComponent {
  currentCustomer = 'Maria';

  title = 'Featured product:';
  itemImageUrl = '../assets/potted-plant.png';
}
```

app.component.html

```html
<div>
  <h1>Interpolation and Template Expressions</h1>
  <hr />

  <div>
    <h2>Interpolation Example</h2>
    <h3>Current customer: {{ currentCustomer }}</h3>

    <p>{{ title }}</p>
    <div><img src="{{ itemImageUrl }}" /></div>
  </div>

  <hr />
</div>
```

Referensi: https://angular.io/guide/interpolation

Property Binding

Property binding dalam Angular akan menge-set nilai ke properti/attribute HTML maupun properti directives. Property binding dapat digunakan untuk membuat toggle button, menge-set value dinamis, dan sharing value antara komponen.

Contoh:

```
<h2>Property binding and interpolation</h2>
<p><img src="{{itemImageUrl}}"> is the <i>interpolated</i> image.</p>
<p><img [src]="itemImageUrl"> is the <i>property bound</i> image.</p>
```

```
<h2>Binding to the colSpan property</h2>
<table border=1>
  <tr><td>Column 1</td><td>Column 2</td></tr>
  <!-- Notice the colSpan property is camel case -->
  <tr><td [colSpan]="2">Span 2 columns</td></tr>
</table>
```

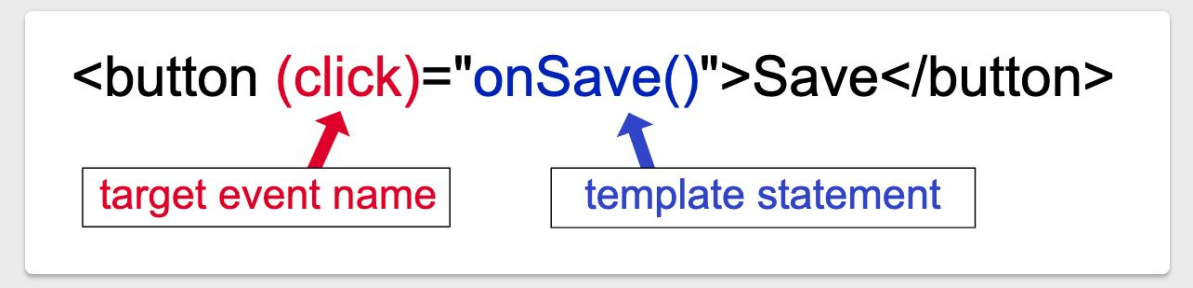
```
<h2>Model property of a custom component:</h2>
<app-item-detail [childItem]="parentItem"></app-item-detail>
<app-item-detail childItem="parentItem"></app-item-detail>

<h3>Pass objects:</h3>
<app-item-list [items]="currentItems"></app-item-list>
```

Referensi: https://angular.io/guide/property-binding

Event Binding

Event binding berguna untuk “listen” atau merespon aksi dari user event seperti keystrokes, mouse movements, clicks, and touches. Diapit oleh **(** dan **)** sebagai delimiter

Contoh:



Referensi: https://angular.io/guide/event-binding

Custom Event: EventEmitter

EventEmitter digunakan di @Output decorator, untuk meng-emit custom event secara synchronous atau asynchronous.

Contoh:

add-new-item.component.ts U

src > app > add-new-item > add-new-item.component.ts > ...

```
1   import { Component, EventEmitter, Output } from '@angular/core';
2
3   @Component({ ...
7   })
8   export class AddNewItemComponent {
9     @Output() newItemEvent = new EventEmitter<string>();
10
11    addNewItem(value: string) {
12      this.newItemEvent.emit(value);
13    }
14  }
```

.emit akan mengirimkan data ke Parent Component.

Referensi: https://angular.io/guide/event-binding

Two Way Binding

Jika data binding sebelumnya mensinkronisasikan satu arah saja (one-way binding), dalam AngularJS ada juga two-way binding, yaitu sinkronisasi antara model di javascript dan tampilan.

Ketika data dalam model berubah, tampilan mencerminkan perubahan, dan ketika data dalam tampilan berubah, model juga diperbarui. Ini terjadi secara otomatis dan memastikan bahwa model dan tampilan diperbarui setiap saat

Sintaks Two Way Data Binding diapit oleh **([** dan **])**

Referensi: https://angular.io/guide/two-way-binding

Two Way Binding: NgModel

Two Way Binding dapat digunakan pada elemen Form, dengan menggunakan **ngModel**.

Contoh dibawah ini, adalah penggunaan two way binding pada elemen input:

app.component.ts

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css'],
})
export class AppComponent {
  message = '';
}
```

app.component.html

```html
<h1 id="two-way">Two-way Binding</h1>
<div id="two-way-1">
  <p>{{ message }}</p>
  <label>Masukkan Pesan: <input [(ngModel)]="message" /></label>
</div>
<br />
```

Referensi: https://angular.io/guide/two-way-binding