Data Binding

Data Binding is available right from AngularJS, Angular 2 and is now available in Angular 4 as well. We use curly braces for data binding - {{}}; this process is called interpolation. We have already seen in our previous examples how we declared the value to the variable title and the same is printed in the browser.

The variable in the **app.component.html** file is referred as {{title}} and the value of title is initialized in the **app.component.ts** file and in **app.component.html**, the value is displayed.

Let us now create a dropdown of months in the browser. To do that , we have created an array of months in **app.component.ts** as follows −

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

// declared array of months.

months = ["January", "Feburary", "March", "April", "May",

"June", "July", "August", "September",

"October", "November", "December"];

}

The month’s array that is shown above is to be displayed in a dropdown in the browser. For this, we will use the following line of code −

<!--The content below is only a placeholder and can be replaced. -->

<div style="text-align:center">

<h1>

Welcome to {{title}}.

</h1>

</div>

<div> Months :

<select>

<option \*ngFor="let i of months">{{i}}</option>

</select>

</div>

We have created the normal select tag with option. In option, we have used the **for loop**. The **for loop** is used to iterate over the months’ array, which in turn will create the option tag with the value present in the months.

The syntax **for** in Angular is **\*ngFor = “let I of months”** and to get the value of months we are displaying it in {{i}}.

The two curly brackets help with data binding. You declare the variables in your **app.component.ts** file and the same will be replaced using the curly brackets.

Let us see the output of the above month’s array in the browser



The variable that is set in the **app.component.ts** can be bound with the **app.component.html** using the curly brackets; for example, **{{}}**.

Let us now display the data in the browser based on condition. Here, we have added a variable and assigned the value as true. Using the if statement, we can hide/show the content to be displayed.

Example

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

//array of months.

months = ["January", "February", "March", "April",

"May", "June", "July", "August", "September",

"October", "November", "December"];

isavailable = true; //variable is set to true

}

<!--The content below is only a placeholder and can be replaced.-->

<div style = "text-align:center">

<h1>

Welcome to {{title}}.

</h1>

</div>

<div> Months :

<select>

<option \*ngFor = "let i of months">{{i}}</option>

</select>

</div>

<br/>

<div>

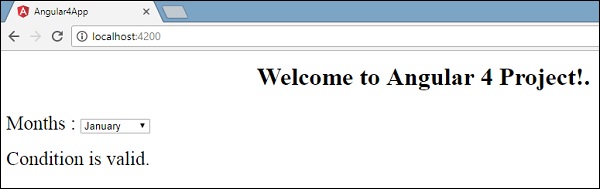
<span \*ngIf = "isavailable">Condition is valid.</span>

//over here based on if condition the text condition is valid is displayed.

If the value of isavailable is set to false it will not display the text.

</div>

Output



Let us try the above example using the **IF THEN ELSE** condition.

Example

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

//array of months.

months = ["January", "February", "March", "April",

"May", "June", "July", "August", "September",

"October", "November", "December"];

isavailable = false;

}

In this case, we have made the **isavailable** variable as false. To print the **else** condition, we will have to create the **ng-template** as follows −

<ng-template #condition1>Condition is invalid</ng-template>

The full code looks like this −

<!--The content below is only a placeholder and can be replaced.-->

<div style="text-align:center">

<h1>

Welcome to {{title}}.

</h1>

</div>

<div> Months :

<select>

<option \*ngFor="let i of months">{{i}}</option>

</select>

</div>

<br/>

<div>

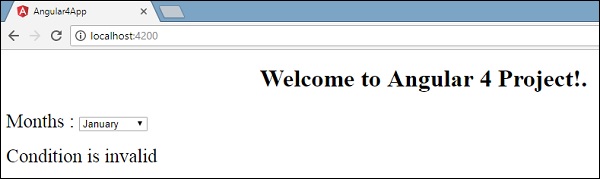
<span \*ngIf="isavailable; else condition1">Condition is valid.</span>

<ng-template #condition1>Condition is invalid</ng-template>

</div>

**If** is used with the else condition and the variable used is **condition1**. The same is assigned as an **id** to the **ng-template**, and when the available variable is set to false the text **Condition is invalid** is displayed.

The following screenshot shows the display in the browser.



Let us now use the **if then else** condition.

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

//array of months.

months = ["January", "February", "March", "April",

"May", "June", "July", "August", "September",

"October", "November", "December"];

isavailable = true;

}

Now, we will make the variable **isavailable** as true. In the html, the condition is written in the following way −

<!--The content below is only a placeholder and can be replaced.-->

<div style="text-align:center">

<h1>

Welcome to {{title}}.

</h1>

</div>

<div> Months :

<select>

<option \*ngFor="let i of months">{{i}}</option>

</select>

</div>

<br/>

<div>

<span \*ngIf="isavailable; then condition1 else condition2">Condition is valid.</span>

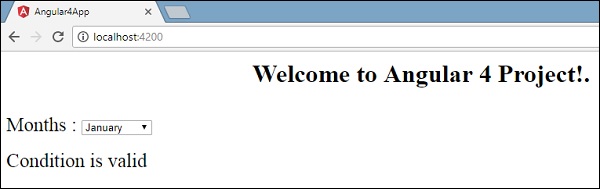
<ng-template #condition1>Condition is valid</ng-template>

<ng-template #condition2>Condition is invalid</ng-template>

</div>

If the variable is true, then **condition1**, else **condition2**. Now, two templates are created with id **#condition1** and **#condition2**.

The display in the browser is as follows −



Event Binding

app.component.html

<!--The content below is only a placeholder and can be replaced.-->

<div style = "text-align:center">

<h1>

Welcome to {{title}}.

</h1>

</div>

<div> Months :

<select>

<option \*ngFor = "let i of months">{{i}}</option>

</select>

</div>

<br/>

<div>

<span \*ngIf = "isavailable; then condition1 else condition2">

Condition is valid.

</span>

<ng-template #condition1>Condition is valid</ng-template>

<ng-template #condition2>Condition is invalid</ng-template>

</div>

<button (click)="myClickFunction($event)">

Click Me

</button>

In the **app.component.html** file, we have defined a button and added a function to it using the click event.

Following is the syntax to define a button and add a function to it.

(click)="myClickFunction($event)"

The function is defined in the **.ts** file: **app.component.ts**

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

//array of months.

months = ["January", "Feburary", "March", "April",

"May", "June", "July", "August", "September",

"October", "November", "December"];

isavailable = true;

myClickFunction(event) {

//just added console.log which will display the event details in browser on click of the button.

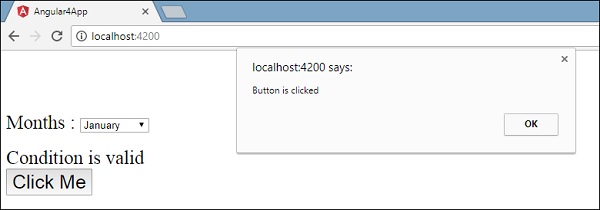
alert("Button is clicked");

console.log(event);

}

}

Upon clicking the button, the control will come to the function **myClickFunction** and a dialog box will appear, which displays **the Button is clicked** as shown in the following screenshot −



Let us now add the change event to the dropdown.

The following line of code will help you add the change event to the dropdown −

<!--The content below is only a placeholder and can be replaced.-->

<div style = "text-align:center">

<h1>

Welcome to {{title}}.

</h1>

</div>

<div> Months :

<select (change) = "changemonths($event)">

<option \*ngFor = "let i of months">{{i}}</option>

</select>

</div>

<br/>

<div>

<span \*ngIf = "isavailable; then condition1 else condition2">

Condition is valid.

</span>

<ng-template #condition1>Condition is valid</ng-template>

<ng-template #condition2>Condition is invalid</ng-template>

</div>

<button (click) = "myClickFunction($event)">Click Me</button>

The function is declared in the **app.component.ts** file −

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

//array of months.

months = ["January", "Feburary", "March", "April",

"May", "June", "July", "August", "September",

"October", "November", "December"];

isavailable = true;

myClickFunction(event) {

alert("Button is clicked");

console.log(event);

}

changemonths(event) {

console.log("Changed month from the Dropdown");

console.log(event);

}

}

The console message “**Changed month from the Dropdown**” is displayed in the console along with the event.



Let us add an alert message in **app.component.ts** when the value from the dropdown is changed as shown below −

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

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 4 Project!';

//array of months.

months = ["January", "February", "March", "April",

"May", "June", "July", "August", "September",

"October", "November", "December"];

isavailable = true;

myClickFunction(event) {

//just added console.log which will display the event details in browser

on click of the button.

alert("Button is clicked");

console.log(event);

}

changemonths(event) {

alert("Changed month from the Dropdown");

}

}

When the value in dropdown is changed, a dialog box will appear and the following message will be displayed - “**Changed month from the Dropdown**”.

