How to Display Validation or Error Messages in Angular Forms

Introduction

Angular is making it easier to build apps for the web. Angular combines dependency injection and declarative templates, and it has integrated best practices to solve development problems.

Forms are one of the most common features in any app. Users can use forms to log in, book a flight, or order food. You can enhance the overall data quality by validating user input for accuracy.

This guide will cover how to display validation or error messages with Angular forms.

Create a New Angular Project

Install Angular globally by using the following command.

1npm install -g @angular/cli

javascript

Then type below command to create a new Angular project.

1ng new ngValidation

javascript

Add a few files and install some dependencies. Navigate to the root app directory and type the following command to start the server.

1cd my-app

2ng serve --open

javascript

You'll create a form with a single input to get an understanding of Angular forms. First, you must include reactive forms in app.module.ts.

1// app.module.ts

2

3import { ReactiveFormsModule } from '@angular/forms';

4

5imports: [

6 BrowserModule, ReactiveFormsModule

7 ],

javascript

Now, we can add the below code to display the form.

1<div style="text-align:left">

2 <h1>

3 Welcome to {{title}}!!

4 </h1>

5 <form [formGroup]="myForm" novalidate>

6 <div class="form-group">

7 <label class="center-block">Username:

8 <input class="form-control" formControlName="username">

9 </label>

10 </div>

11 <div \*ngIf="myForm.controls['username'].invalid && (myForm.controls['username'].dirty || myForm.controls['username'].touched)" class="alert">

12 <div \*ngIf="myForm.controls['username'].errors.required">

13 Please enter username

14 </div>

15 </div>

16 </form>

17<p>myForm value: {{ myForm.value | json }}</p>

18<p>myForm status: {{ myForm.status | json }}</p>

19</div>

javascript

You also need to update the app.component.ts file.

1// app.component.ts

2

3import { Component } from '@angular/core';

4import { FormGroup, FormBuilder, Validators } from '@angular/forms';

5

6@Component({

7 selector: 'app-root',

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

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

10})

11export class AppComponent {

12 title = 'My Angular Form Validation Example';

13 myForm: FormGroup;

14 constructor(private fb: FormBuilder) {

15 this.createForm();

16 }

17 createForm() {

18 this.myForm = this.fb.group({

19 username: ['', Validators.required ]

20 });

21 }

22}

javascript

Template-Driven Forms

You can create a template-driven form in Angular template syntax using the form directives. Build template-driven forms by following these steps:

1. Create a new Angular component with an initial form layout.
2. Add data properties to bind for each form control and add an attribute to each form-input control.
3. Show/hide validation messages.
4. Handle form submit using ngSubmit.
5. Optional: Add feature to disable the submit button initially and enable only after the form is valid.

Reactive Forms

Reactive forms are used to create forms that contain a reactive style. A reactive style of programming is enabled by Angular reactive forms that supports detailed management of the data flowing between a non-UI data model (typically retrieved from a server) and a UI-oriented form model that contains the states and values of the HTML controls on the screen. Reactive forms provide the comfort of utilizing reactive patterns, testing, and validation.

Let's look at an example.

1// app.component.ts

2

3import { Component } from '@angular/core';

4import { FormGroup, FormBuilder, Validators } from '@angular/forms';

5

6@Component({

7 selector: 'app-root',

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

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

10})

11export class AppComponent {

12 title = 'Angular Form Validation Tutorial';

13 myForm: FormGroup;

14 constructor(private fb: FormBuilder) {

15 this.createForm();

16 }

17 createForm() {

18 this.myForm = this.fb.group({

19 username: ['', Validators.required ]

20 });

21 }

22}

javascript

Here, you need to import a few modules from @angular/forms.

1. FormGroup
2. FormBuilder
3. Validators and ReactiveFormsModule

1// app.component.ts

2

3constructor(private fb: FormBuilder) {

4 this.createForm();

5}

javascript

Use a FormBuilder to handle validations. Create a form with all the validation rules inside the constructor.

1// app.component.ts

2

3createForm() {

4 this.myForm = this.fb.group({

5 username: ['', Validators.required ]

6 });

7}

javascript

Assign the validation rules to the form object in this code.

1<form [formGroup]="myForm" novalidate>

2 <div class="form-group">

3 <label class="center-block">Username:

4 <input class="form-control" formControlName="username">

5 </label>

6 </div>

7 <div \*ngIf="myForm.controls['username'].invalid && (myForm.controls['username'].dirty || myForm.controls['username'].touched)" class="alert">

8 <div \*ngIf="myForm.controls['username'].errors.required">

9 Please enter username

10 </div>

11 </div>

12 </form>

13<p>Form value: {{ myForm.value | json }}</p>

14<p>Form status: {{ myForm.status | json }}</p>

javascript

Conclusion

You have now seen both template-driven forms and reactive forms. When a user touches any input field but does not enter a value, an error message will be shown. Again, when the user blurs the input field without entering a value, an error message will be displayed. Thus, we are able to display validation messages for the form fields.