Angular 16 : Online Class

Reactive forms in Angular

By: Sahosoft Solutions

Presented by : Chandan Kumar

Reactive forms

In a model-driven approach, the model which is created in the .ts file is responsible for handling all the user interactions/validations. For this, first, we need to create the Model using Angular’s inbuilt classes like **formGroup** and **formControl** and then, we need to bind that model to the HTML form.

This approach uses the Reactive forms for developing the forms which favor the explicit management of data between the UI and the Model. With this approach, we create the tree of Angular form controls and bind them in the native form controls. As we can create the form controls directly in the component, it makes it a bit easier to push the data between the data models and the UI elements.

Model driven forms are more powerful and more flexible.

There are some things that we can't do with template driven form but with model driven form we can perform those things. Like if we want to detect if any changes occur in any variable we can do it with model driven form , we can easily do two-way data binding with model drivven forms and that is very important in large applications.

Reactive forms

Reactive form in Angular is a technique to manage your form in a reactive manner, it means that you can manage your form and validation from our component itself .

These forms are the best option when we have a complex form requirement .

Compared to Template deriven forms, reactive forms are more suitable because we can define validations and model from component , that gives us more control on form .

Primarily it is also called "Model Driven Forms" , because model acts as a mediator between component and template .

It's really easy to implement, like using model in controller with form control object and binding it to component formcontrol template .

Reactive forms

Prerequisite

we need to import **ReactiveFormsModule** in our app.module.ts file.

Reactive Forms Features

- More flexible, but needs a lot of practice
- Handles any complex scenarios
- No data binding is done (immutable data model preferred by most developers)
- More component code and less HTML markup
- Easier unit testing

Reactive forms

You can see that in app.module.ts file, we have FormsModule in it, replace it with ReactiveFormsModule as below,

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { ReactiveFormsModule } from '@angular/forms';
import { AppComponent } from './app.component';
@NgModule({
    declarations: [
    AppComponent
    ],
    imports: [
        BrowserModule,
        ReactiveFormsModule
    ],
    bootstrap: [AppComponent]
    })
export class AppModule{}
```

Reactive forms

Now lets come back to app.component.ts file,we need to import 3 more libraries
import {FormGroup,FormControl,FormBuilder} from '@angular/forms'

Now in our app.component, we first need to add a form class and after that we need to define a function in order to detect if any of the elements in html changed or not.

```
ngOnInit() {
    this.form = new FormGroup({
        firstname: new FormControl(""),
        lastname: new FormControl(""),
        languages: new FormControl(""),
    })
}
```

Reactive forms

FormControl

It tracks the value of the controls and validates the individual control in the form.
In Reactive forms we initialize FormControl object to use form functionality into our component , which engaged with our html form .
And when we update our form control's value than it will directly be reflected to our FormControl object that we created previously .

FormGroup

Tracks the validity and state of the group of FormControl Instance or moreover, we can say the formgroup to be a collection of FormControls.
like :- Validations , name of input control etc ...

FormBuilder

This helps us to develop the forms along with their initial value and there validations.
Angular has a new helper Class called FormBuilder. FormBuilder allows us to explicitly declare forms in our components. This allows us to also explicitly list each form control’s validators.

Angular 16 : Online Class

Reactive forms with Validation

By: Sahosoft Solutions

Presented by : Chandan Kumar

Reactive forms with Validation

The most important part of any form is to add the validations in the application.
When we do the fname:new FormControl(), the first argument in the FormControl is the initial value and the second value is the Validations that we can add there.
The only change in the component would be like below.

```
this.signupForm= frmbuilder.group({
    fname:['',Validators.compose([Validators.required,Validators.maxLength(16),Validators.minLength(1)])],
   lname:['',[Validators.required,Validators.maxLength(19)]],
   Emailid:['',[Validators.required,Validators.email]],
   userpassword:['',Validators.required]
```

Angular 16 : Online Class

FormGroup Get Value and reset()

By: Sahosoft Solutions

Presented by : Chandan Kumar

FormGroup Get Value and reset()

- To get the value of form control named as name after form submit, we need to write the code as below.

```
this.userForm.get('name').value
```

- If we want to get all values of the form then we can write code as given below.

```
this.userForm.value
```

- To reset the form we need to call reset() method on FormGroup instance. Suppose userForm is the instance of FormGroup then we create a method as given below.

```
resetForm() {
this.userForm.reset();
}
```

We can call the above method using a button as following.

```
<button type="button" (click) = "resetForm()">Reset</button>
```

- If we want to reset form with some default values, then assign the default values with form control name of the form.

```
resetForm() {
userForm.reset({
   name: 'Mahesh',
  age: 20
  });}
```

Angular 16 : Online Class

FormGroup setValue() and patchValue()

By: Sahosoft Solutions

Presented by : Chandan Kumar

FormGroup setValue() and patchValue()

Angular FormGroup has methods such as setValue() and patchValue(). setValue() and patchValue() both sets the value in form control of FormGroup.

setValue() sets the value in each and every form control of FormGroup. We cannot omit any form control in setValue()

but when we want to assign only few form controls of FormGroup then we need to use patchValue().

Both methods are used in the same way.

It is necessary to mention all from controls in setValue() otherwise it will throw error.

When we use patchValue() then it is not necessary to mention all form controls.

Angular 16 : Online Class

valueChanges and statusChanges

By: Sahosoft Solutions

Presented by : Chandan Kumar

valueChanges and statusChanges

valueChanges and statusChanges are properties of **FormControl**, **FormArray** and **FormGroup** classes. valueChanges and statusChanges both return Observable instance and we can subscribe them to get data.

valueChanges and statusChanges

valueChanges

We can see that it returns Observable of any type. We can subscribe valueChanges to get data.

1. If we subscribe valueChanges of a FormControl instance, we get latest value of that control whenever there is any change.

2. If we subscribe valueChanges of a FormArray instance, we get latest value of those array controls whenever there is any change.

3. If we subscribe valueChanges of a FormGroup instance, we get latest value of the form controls whenever there is a change in any control of the form.

statusChanges

StatusChanges is a property of AbstractControl that emits an event every time when the validation status of the control is recalculated. statusChanges property is available
in FormControl, FormArray and FormGroup classes because they inherit AbstractControl class.

1. If we subscribe statusChanges of a FormControl instance, we get latest validation status of that control whenever validation status is recalculated for that control.

2. If we subscribe statusChanges of a FormArray instance, we get latest validation status of those array controls whenever validation status is recalculated for those array controls.

3. If we subscribe statusChanges of a FormGroup instance, we get latest validation status of the form controls whenever validation status is recalculated in any control of the form.

Angular 16 : Online Class

FormArray

By: Sahosoft Solutions

Presented by : Chandan Kumar

FormArray

It is a class that tracks the value and validity state of array of FormControl, FormGroup and FormArray instances. A FormArray aggregates the values of each child FormControl into an array.

Form arrays allow you to create new form controls dynamically.

setValue() and patchValue() are also the methods of FormArray class.

setValue(): Sets the value of the FormArray. We need to pass an array that must match the structure of the control. If our array does not match the structure of the control, setValue() will throw error.

patchValue(): Patches the value of FormArray. We need to pass an array that should match the structure of the control. If the array that we pass, does not match the structure of the control completely, then patchValue() will patch only those fields which are matching with the structure of the control and rest will be as it is.

npm Package @angular/forms
Module import { FormArray } from '@angular/forms';