Reactive forms are Angular way of creating forms in a reactive style. With Reactive forms, we create form control objects in a component class and bind them with HTML form elements in the template.

As we create and manipulate form control objects directly in the component class, we can push model values into the form controls and fetch user-changed values back from the form. The component can observe changes in form control state and react to those changes.

One advantage of working with form control objects directly is that value and validity updates are always synchronous and under our control.

We will be using **FormBuilder** class to create reactive forms which has simplified syntax. We need to import **ReactiveFormsModule** to create reactive forms.

We can use built-in validators using **Validators** class. For example, if we want to use required validator, it can be accessed as **Validators.required**

**Example:**

**app.module.ts**

1. ....
2. import { ReactiveFormsModule } from '@angular/forms';
3. ....
4. @NgModule({
5. ....
6. imports: [
7. ReactiveFormsModule
8. ],
9. ....
10. })

**Line 3:** Import ReactiveFormsModule from @angular/forms module

**Line 11:** Add ReactiveFormsModule in the imports declaration to create reactive forms

**registration-form.component.ts**

1. ....
2. import { FormBuilder, FormGroup, Validators } from '@angular/forms';
3. export class RegistrationFormComponent implements OnInit {
4. registerForm!: FormGroup;
5. submitted:boolean=false;
6. constructor(private formBuilder: FormBuilder) { }
7. ngOnInit() {
8. this.registerForm = this.formBuilder.group({
9. firstName: ['', Validators.required],
10. lastName: ['', Validators.required],
11. address: this.formBuilder.group({
12. street: [],
13. zip: [],
14. city: []
15. })
16. });
17. }

**Line 3:** Import FormBuilder class to create a reactive form. Also, import FormGroup class to create a group of form controls and Validators for validation

**Line 7:** Create a property **registerForm** of type FormGroup

**Line 9:** Inject a formBuilder instance using constructor

**Line 12:** formBuilder.group() is a factory method that creates a FormGroup. It takes an object whose keys and values are FormControl names and their definitions

**Line 13-19:** Create form controls such as firstName, lastName, and address as subgroup with fields street, zip and city. These fields are form controls. Configure built-in validators for each control using [' ', Validators.required] syntax where the first parameter is the default value for the control and the second parameter is an array of validations. If multiple validators are to be applied, then we should give it as [' ', [Validators.required, Validators.maxlength(10)]]

**Advantages of Model Driven Forms**

1. We can now perform unit testing(using Jasmine framework) on the validation logic as it is written inside component class
2. We can listen to form changes or events easily using reactive forms. Each FormGroup or FormControl has few events like valueChanges, statusChanges etc.,  which we can subscribe to.
3. Model driven forms are used in creating medium to large scale applications.