Angular

Anil K

Forms and Validations

Template Driven Forms

- Template Driven Forms are suitable for development of simple forms with limited no. of fields and simple validations.
- In these forms, each field is represented as a property in the component class.
- Validations rules are defined in the template, using “html 5” attributes. Validation messages are displayed using “validation properties” of angular.
- “FormsModule” should be imported from “@angular/forms” package.

HTML 5 attributes for validations:

- required=”required” : Field is mandatory
- minlength=”n” : Minimum no. of characters
- pattern=”reg exp” : Regular expression

Validation Properties:

- untouched
  - true : Field is not focused.
  - false : Field is focused.
- Touched
  - true : Field is focused.
  - false : Field is not focused.
- Pristine
  - true : Field is not modified by the user.
  - false : Field is modified by the user.
- Dirty
  - true : Field is modified by the user.
  - false : Field is not modified by the user.
- Valid
  - true : Field value is valid.
  - false : Field value is invalid

- invalid
  - true : Field value is invalid.
  - false : Field value is valid.
- errors : Represents the list of errors of the field.
  - required : true / false
  - minlength : true / false
  - pattern : true / false
  - number : true / false
  - email : true / false
  - url : true / false

Sl. No	Description	Regular Expression
1	Digits only	^[0-9]*$
2	Alphabets only	^[a-zA-Z]*$
3	Mobile Number	^[789]\d{9}$
4	Email	\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*
5	Usernames: Alphabets, Digits and Hyphens only	([A-Za-z0-9-]+)
6	Passwords: 6 to 15 characters; atleast one upper case letter, one lower case letter and one digit	((?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{6,15})

Template Driven Forms

- **Example**
- We are going to create a sample template driven form with validations.
  Fields:
  - Firstname
  - Lastname
  - Email
  - Amount
  - Gender
  - Country
- **Creating Application**
- Open Command Prompt and enter the following commands:
- cd c:\angular ng new app1
- cd c:\angular\app1

c:\angular\app1\src\app\app.module.ts

```
import { BrowserModule } from '@angular/platform-browser’;
import { NgModule } from '@angular/core’;
import { AppComponent } from './app.component’;
import { FormsModule } from "@angular/forms";

@NgModule({
declarations: [     AppComponent    ],
imports: [     BrowserModule, FormsModule    ],
providers: [],
bootstrap: [AppComponent]
})
export class AppModule { }
```

c:\angular\app1\src\app\app.component.ts

```
import { Component } from "@angular/core";
@Component({
selector: 'app-root',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
firstname: string = null;
lastname: string = null;
email: string = null;
amount: string = null;
gender: string = null;
country: string = "";
msg: string = null;
      onRegisterClick(f)   {
            if (f.valid)     {       this.msg = "Firstname: " + this.firstname + "<br>Lastname: " + this.lastname +
            "<br>Email: " + this.email + "<br>Amount: " + this.amount + "<br>Gender: " + this.gender +
            "<br>Country: " + this.country;     }
            else     {       this.msg = "Invalid";     }
      }
}
```

c:\angular\app1\src\app\app.component.html

```
<div>
    <h4>Template Drive Forms</h4>
    <form #myform="ngForm">
        Firstname: <input type="text" [(ngModel)]="firstname" name="username" required="required" minlength="3"
            maxlength="20" pattern="^[a-zA-Z ]*$" #control1="ngModel">
        <span class="error" *ngIf="control1.touched && control1.invalid && (control1.errors &&
control1.errors.required)">
         Firstname can't be blank</span>
        <span class="error" *ngIf="control1.touched && control1.invalid && (control1.errors &&
control1.errors.minlength)">
         Min: 3 characters</span>
        <span class="error" *ngIf="control1.touched && control1.invalid && (control1.errors &&
control1.errors.pattern)">
         Alphabets only allowed</span> <br>
        Lastname: <input type="text" [(ngModel)]="lastname" name="lastname" required="required" minlength="3"
            maxlength="20" pattern="^[a-zA-Z ]*$" #control2="ngModel">
        <span class="error" *ngIf="control2.touched && control2.invalid && (control2.errors &&
control2.errors.required)">
         Lastname can't be blank</span>
        <span class="error" *ngIf="control2.touched && control2.invalid && (control2.errors &&
control2.errors.minlength)">
         Min: 3 characters</span>
        <span class="error" *ngIf="control2.touched && control2.invalid && (control2.errors &&
control2.errors.pattern)">
         Alphabets only allowed</span> <br>
```

```
Email: <input type="text" [(ngModel)]="email" name="email" required="required"
        pattern="^[a-z0-9]+(\._a-z09]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,15})$" #control3="ngModel">
    <span class="error" *ngIf="control3.touched && control3.invalid && (control3.errors && control3.errors.required)">
        Email can't be blank</span>
    <span class="error" *ngIf="control3.touched && control3.invalid && (control3.errors && control3.errors.pattern)">
        Email is not valid</span> <br>
Amount: <input type="text" [(ngModel)]="amount" name="amount" required="required" pattern="^[0-9]*$" #control4="ngModel">
    <span class="error" *ngIf="control4.touched && control4.invalid && (control4.errors && control4.errors.required)">
        Amount can't be blank</span>
    <span class="error" *ngIf="control4.touched && control4.invalid && (control4.errors && control4.errors.pattern)">
        Alphabets not allowed</span> <br>
Gender: <input type="radio" [(ngModel)]="gender" value="male" name="gender" required="required"
        #control5="ngModel"> Male
    <input type="radio" [(ngModel)]="gender" value="female" name="gender" required="required" #control5="ngModel">
        Female
    <span class="error" *ngIf="control5.touched && control5.invalid && (control5.errors && control5.errors.required)">
        Please Select Gender</span> <br>
```

```
Country: <select [(ngModel)]="country" name="country" required="required" #control6="ngModel">
    <option value="">Please Select</option>
    <option>USA</option>
    <option>UK</option>
    <option>Japan</option>
</select>
<span class="error" *ngIf="control6.touched && control6.invalid && (control6.errors && control6.errors.required)">
Please Select Country</span> <br>

<input type="submit" value="Register" (click)="onRegisterClick(myform)"><br>

<div [innerHTML]="msg"></div>
</form>
</div>
```

Reactive Forms (or) Model Driven Forms

- Reactive Forms (or) Model Driven Forms are new types of forms in angular, which are suitable for creating large forms with many fields and complex validations.
- In these forms, each field is represented as “FormControl” and group of controls is represented as “FormGroup”.
- “ReactiveFormsModule” should be imported from “@angular/forms” package.
- Validation rules are defined in the component using "Validators" object of angular and validation messages are displayed in the template using "validation properties" of angular.

- **Validations in Reactive Forms:**
  - Validators.required : Field is mandatory
  - Validators.minLength : Minimum no. of characters
  - Validators.maxLength : Maximum no. of characters
  - Validators.pattern : Regular expression

Validation Properties:

- untouched
  - ▪ true : Field is not focused.
  - ▪ false : Field is focused.
- touched
  - ▪ true : Field is focused.
  - ▪ false : Field is not focused.
- pristine
  - ▪ true : Field is not modified by the user.
  - ▪ false : Field is modified by the user.
- dirty
  - ▪ true : Field is modified by the user.
  - ▪ false : Field is not modified by the user.
- valid
  - ▪ true : Field value is valid.
  - ▪ false : Field value is invalid
- invalid
  - ▪ true : Field value is invalid.
  - ▪ false : Field value is valid.
- errors : Represents the list of errors of the field.
  - ▪ required : true / false
  - ▪ minlength : true / false
  - ▪ maxlength : true / false
  - ▪ pattern : true / false

Reactive Forms – Example

- We are going to create a sample reactive form with validations.
- Fields:
  - Firstname
  - Lastname
  - Email
  - Amount
  - Gender
  - Country

Creating Application

- Open Command Prompt and enter the following commands:
- cd c:\angular ng new app1
- cd c:\angular\app1

c:\angular\app1\src\app\app.module.ts

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

c:\angular\app1\src\app\app.component.ts

```
import { Component } from "@angular/core";
import { FormGroup, FormControl, Validators } from "@angular/forms";
@Component({
selector: 'app-root',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
msg: string = "";    myform: FormGroup;
constructor()   {
this.myform = new FormGroup({
firstname: new FormControl("", [Validators.required, Validators.minLength(3),
Validators.maxLength(20), Validators.pattern("^[a-zA-Z ]*$")]),
lastname: new FormControl("", [Validators.required, Validators.minLength(5),
Validators.maxLength(20), Validators.pattern("^[a-zA-Z ]*$")]),
email: new FormControl("", [Validators.required,
Validators.pattern("^[a-z0-9]+(\._a-z0-9]+)*@[a-z0-9-]+(\.[az0-9-]+)*(\.[a-z]{2,15})$")]),
amount: new FormControl("", [Validators.required, Validators.pattern("^[0-9]*$")]),
gender: new FormControl("", [Validators.required]),
country: new FormControl("", [Validators.required])     });
}
```

```
onRegisterClick()  {
    if (this.myform.valid)    {
    this.msg = "First Name: " + this.myform.controls.firstname.value + "<br>Last
    Name: " + this.myform.controls.lastname.value + "<br>Email: " +
    this.myform.controls.email.value + "<br>Amount: " +
    this.myform.controls.amount.value + "<br>Gender: " +
    this.myform.controls.gender.value + "<br>Country: " +
    this.myform.controls.country.value;
    }    else    {
    this.msg = "Invalid";
    }
}
}
```

c:\angular\app1\src\app\app.component.html

```
<div>
<h4>Reactive Drive Forms</h4>
<form [formGroup]="myform">
    First Name:     <input type="text" formControlName="firstname">
    <span class="error" *ngIf="myform.controls.firstname.touched && myform.controls.firstname.invalid
    && myform.controls.firstname.errors.required">Firstname can't be blank</span>
    <span class="error" *ngIf="myform.controls.firstname.touched && myform.controls.firstname.invalid
    && myform.controls.firstname.errors.minlength">Min: 5 characters</span>
    <span class="error" *ngIf="myform.controls.firstname.touched && myform.controls.firstname.invalid
    && myform.controls.firstname.errors.maxlength">Max: 20 characters</span>
    <span class="error" *ngIf="myform.controls.firstname.touched && myform.controls.firstname.invalid
    && myform.controls.firstname.errors.pattern">Alphabets only allowed</span>     <br>
    Last Name:     <input type="text" formControlName="lastname">
    <span class="error" *ngIf="myform.controls.lastname.touched && myform.controls.lastname.invalid
    && myform.controls.lastname.errors.required">Lastname can't be blank</span>
    <span class="error" *ngIf="myform.controls.lastname.touched && myform.controls.lastname.invalid
    && myform.controls.lastname.errors.minlength">Min: 5 characters</span>
    <span class="error" *ngIf="myform.controls.lastname.touched && myform.controls.lastname.invalid
    && myform.controls.lastname.errors.maxlength">Max: 20 characters</span>
    <span class="error" *ngIf="myform.controls.lastname.touched && myform.controls.lastname.invalid
    && myform.controls.lastname.errors.pattern">Alphabets only allowed</span>     <br>
```

```
Email:      <input type="text" formControlName="email">

<span class="error" *ngIf="myform.controls.email.touched &&
myform.controls.email.invalid && myform.controls.email.errors.required">Email
can't be blank</span>

<span class="error" *ngIf="myform.controls.email.touched &&
myform.controls.email.invalid && myform.controls.email.errors.pattern">Email
is not valid</span>      <br>

Amount:      <input type="text" formControlName="amount">

<span class="error" *ngIf="myform.controls.amount.touched &&
myform.controls.amount.invalid &&
myform.controls.amount.errors.required">Amount can't be blank</span>

<span class="error" *ngIf="myform.controls.amount.touched &&
myform.controls.amount.invalid &&
myform.controls.amount.errors.pattern">Alphabets not allowed</span>      <br>

Gender:      <input type="radio" formControlName="gender" value="male">      Male
<input type="radio" formControlName="gender" value="female">      Female
<span class="error" *ngIf="myform.controls.gender.touched &&
myform.controls.gender.invalid &&
myform.controls.gender.errors.required">Plase Select Gender</span>      <br>
```

```
Country:      <select formControlName="country">
<option>Please Select</option>
<option>India</option>
<option>USA</option>
<option>UK</option>
<option>Japan</option>
</select>
<span class="error" *ngIf="myform.controls.country.touched &&
myform.controls.country.invalid &&
myform.controls.country.errors.required">Plase Select Country</span>      <br>
<input type="submit" value="Submit" (click)="onRegisterClick()"><br>      <div
[innerHTML]="msg"></div>
 </form>
</div>
```