# Template-driven forms

* Template-driven Forms is one of the two ways of building forms in Angular.
* First, we build a simple HTML form using a few form elements.
* The ngForm directive will convert it to the Template-driven form and create the top-level FormGroup control.
* Next, we use the ngModel directive to create the FormControl instance for each of the HTML form elements.
* Later, we will learn how to submit the form data to the component class.

In Template Driven Forms we specify behaviors/validations using directives and attributes in our template and let it work behind the scenes. All things happen in Templates hence very little code is required in the component class.

1. The form is set up using ngForm directive
2. Controls are set up using the ngModel directive
3. ngModel also provides the two-way data binding
4. The Validations are configured in the template via directives

Create the Example Application

ng new AngularApp

Import FormsModule

In app.module.ts

import { FormsModule } from '@angular/forms';

imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule

  ],

HTML Form

Following is the HTML Form.

<span><h2>{{ title }}</h2></span>

<form>

    <p>

      <label for="bookName">Book Name</label>

      <input type="text" id="bookName" name="bookName">

    </p>

    <p>

      <label for="authorName">Author</label>

      <input type="text" id="authorName" name="authorName">

    </p>

    <p>

        <label for="bookPrice">Price </label>

        <input type="text" id="bookPrice" name="bookPrice">

    </p>

    <p>

      <label for="publisher">Publisher </label>

      <input type="text" id="publisher" name="publisher">

    </p>

    <p>

      <label for="genre">Genre</label>

      <input type="radio" value="fiction" id="genre" name="genre"> Fiction

      <input type="radio" value="science" id="genre" name="genre"> Science

      <input type="radio" value="novel" id="genre" name="genre"> Novel

      <input type="radio" value="self help" id="genre" name="genre"> Sefl Help

      <input type="radio" value="autobiography" id="genre" name="genre"> Autobiography

      <input type="radio" value="technology" id="genre" name="genre"> Technology

    </p>

    <p>

      <button type="submit">Submit</button>

    </p>

  </form>

### ngForm

The ngForm directive is what makes the Angular template-driven forms work. But we do not need to add it explicitly. Angular adds it automatically.

When we include FormsModule, the Angular is going to look out for any <form> tag in our HTML template. Angular does this via ngForm directive. ngForm directive automatically detects the <form> tag and automatically binds to it.

The ngForm does the following

1. Binds itself to the <Form> directive
2. Creates a top-level FormGroup instance
3. Creates FormControl instance for each of child control, which has ngModel directive.
4. CreatesFormGroup instance for each of the NgModelGroup directive.

We can export the ngForm instance into a local template variable using ngForm as the key (ex: #bookForm="ngForm").

Hence, update the form element as shown below.

<form #bookForm="ngForm">

### FormControl

1. The FormControl is the basic building block of the Angular Forms.
2. It represents a single input field in an Angular form.
3. The Angular Forms Module binds the input element to a FormControl.
4. This can be done by using the ngModel directive.
5. Add the ngModel directive to each control as shown below.

<input type="text" id="bookName" name="bookName" ngModel>

### Submit Form

We use the ngSubmit event, to submit the form data to the component class.

When the user clicks on the submit button, the ngSubmit event will fire

<form #bookForm="ngForm" (ngSubmit)="onSubmit(bookForm)">

### Final Template

<span><h2>{{ title }}</h2></span>

<form #bookForm="ngForm" (ngSubmit)="onSubmit(bookForm)">

    <p>

      <label for="bookName">Book Name</label>

      <input type="text" id="bookName" name="bookName" ngModel>

    </p>

    <p>

      <label for="authorName">Author</label>

      <input type="text" id="authorName" name="authorName" ngModel>

    </p>

    <p>

        <label for="bookPrice">Price </label>

        <input type="text" id="bookPrice" name="bookPrice" ngModel>

    </p>

    <p>

      <label for="publisher">Publisher </label>

      <input type="text" id="publisher" name="publisher" ngModel>

    </p>

    <p>

      <label for="genre">Genre</label>

      <input type="radio" value="fiction" id="genre" name="genre" ngModel> Fiction

      <input type="radio" value="science" id="genre" name="genre" ngModel> Science

      <input type="radio" value="novel" id="genre" name="genre" ngModel> Novel

      <input type="radio" value="self help" id="genre" name="genre" ngModel> Sefl Help

      <input type="radio" value="autobiography" id="genre" name="genre" ngModel> Autobiography

      <input type="radio" value="technology" id="genre" name="genre" ngModel> Technology

    </p>

    <p>

      <button type="submit">Submit</button>

    </p>

  </form>

### Receive Form Data

1. To receive the form data we need to create the onSubmit method in our component class.
2. The submit method receives the reference to the ngForm directive, which we named as bookForm.
3. The bookForm exposes the value method which returns the form fields as a Json object.

onSubmit(bookForm: NgForm){

    console.log(bookForm.value);

  }

### @ViewChild

1. An alternate way to submit the form using @ViewChild
2. This is used mostly before submitting the form

Code for Template

<form #bookForm="ngForm" (ngSubmit)="onSubmit()">

Code for Component

@ViewChild('bookForm') bForm: NgForm;

onSubmit(){

    console.log(this.bForm)

  }

1. We can add new [FormGroup](https://www.tektutorialshub.com/angular/formgroup-in-angular/) using the ngModelGroup directive.

Code for Template

<div class="container">

<span><h2>{{ title }}</h2></span>

 <form #bookForm="ngForm" (ngSubmit)="onSubmit(bookForm)">

    <p>

      <label for="bookName">Book Name</label>

      <input type="text" id="bookName" name="bookName" required ngModel>

    </p>

    <p>

      <label for="authorName">Author</label>

      <input type="text" id="authorName" name="authorName" required ngModel>

    </p>

    <p>

        <label for="bookPrice">Price </label>

        <input type="text" id="bookPrice" name="bookPrice" required ngModel>

    </p>

    <p>

      <label for="publisher">Publisher </label>

      <input type="text" id="publisher" name="publisher" required ngModel>

    </p>

     <p>

      <label for="genre">Genre</label>

      <input type="radio" value="fiction" id="genre" name="genre" ngModel> Fiction

      <input type="radio" value="science" id="genre" name="genre" ngModel> Science

      <input type="radio" value="novel" id="genre" name="genre" ngModel> Novel

      <input type="radio" value="self help" id="genre" name="genre" ngModel> Sefl Help

      <input type="radio" value="autobiography" id="genre" name="genre" ngModel> Autobiography

      <input type="radio" value="technology" id="genre" name="genre" ngModel> Technology

    </p>

    <div ngModelGroup="publisher\_contact">

      <p>

        <label for="city">City</label>

        <input type="text" name="city" ngModel>

      </p>

      <p>

        <label for="phone">Phone</label>

        <input type="text" name="phone" ngModel>

      </p>

      <p>

        <label for="email">email</label>

        <input type="text" name="email" ngModel>

      </p>

  </div>

    <p>

      <button type="submit" [disabled]="!bookForm.valid">Submit</button>

    </p>

  </form>

<b>valid</b> {{bookForm.valid}} <br>

  <b>touched</b> {{bookForm.touched}} <br>

  <b>pristine</b> {{bookForm.pristine}} <br>

  <b>dirty</b> {{bookForm.dirty}}<br>

  <b>Value: </b> {{ bookForm.value | json}}

</div>

Code for Component

@Component({

  selector: 'my-app',

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

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

})

export class AppComponent  {

  title = "Template Driven Form";

  bookForm: NgForm;

  constructor() { }

  ngOnInit(): void {

  }

   onSubmit(bookForm: NgForm){

     console.log(bookForm.value);

  }

}

Validation

Template after Validation

<div class="container">

<span><h2>{{ title }}</h2></span>

<form #bookForm="ngForm" (ngSubmit)="onSubmit(bookForm)">

<p>

<label for="bookName">Book Name</label>

<input type="text" id="bookName" name="bookName" required ngModel>

</p>

<p>

<label for="authorName">Author</label>

<input type="text" id="authorName" name="authorName" required ngModel>

</p>

<p>

<label for="bookPrice">Price </label>

<input type="text" id="bookPrice" name="bookPrice" required ngModel>

</p>

<p>

<label for="publisher">Publisher </label>

<input type="text" id="publisher" name="publisher" required ngModel>

</p>

<p>

<label for="genre">Genre</label>

<input type="radio" value="fiction" id="genre" name="genre" ngModel> Fiction

<input type="radio" value="science" id="genre" name="genre" ngModel> Science

<input type="radio" value="novel" id="genre" name="genre" ngModel> Novel

<input type="radio" value="self help" id="genre" name="genre" ngModel> Sefl Help

<input type="radio" value="autobiography" id="genre" name="genre" ngModel> Autobiography

<input type="radio" value="technology" id="genre" name="genre" ngModel> Technology

</p>

<div ngModelGroup="publisher\_contact">

<p>

<label for="city">City</label>

<input type="text" name="city" required ngModel>

</p>

<p>

<label for="phone">Phone</label>

<input type="text" name="phone" required ngModel>

</p>

<p>

<label for="email">email</label>

<input type="text" name="email" required email ngModel>

</p>

</div>

<p>

<button type="submit" [disabled]="!bookForm.valid">Submit</button>

</p>

</form>

<b>valid</b> {{bookForm.valid}} <br>

<b>touched</b> {{bookForm.touched}} <br>

<b>pristine</b> {{bookForm.pristine}} <br>

<b>dirty</b> {{bookForm.dirty}}<br>

<b>Value: </b> {{ bookForm.value | json}}

</div>

Css

.container {

margin-top: 30px;

margin-left: 30px;

}

input.ng-invalid.ng-touched{

border: 1px solid red;

}