Template-driven vs. Reactive Forms

Deborah Kurata
CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/

Module Overview

Angular Form Building Blocks
- FormGroup
- FormControl

Template Syntax for Forms

Template-driven Forms

Complex Scenarios

Sign Up!

First Name

First Name (required)

Last Name

Last Name (required)

Email

Email (required)

☑ Send me your catalog

Address Type   ⦿ Home   ◯ Work   ◯ Other

Street Address 1

Street address

Street Address 2

Street address (second line)

City, State, Zip Code

City   Select a State...   ▾   Zip Code

Save

Sign Up!

First Name

First Name (required)

Please enter your first name.

Last Name

Last Name (required)

Email

Email (required)

☑ Send me your catalog

Address Type   ● Home   ○ Work   ○ Other

Street Address 1

Street address

Street Address 2

Street address (second line)

City, State, Zip Code

City

Select a State...   ▾

Zip Code

Save

State

Form Building Blocks

FormControl

FormGroup

```
▼ controls: Object
  ▶ email: FormControl
  ▶ firstName: FormControl
  ▶ lastName: FormControl
  ▶ sendCatalog: FormControl
  ▶ __proto__: Object
  dirty: true
  disabled: false
  enabled: true
  errors: null
  invalid: false
  pending: false
  pristine: false
  root: (...)
  status: (...)
  statusChanges: (...)
  touched: true
  untouched: false
  valid: true
  validator: null
▼ value: Object
    email: "jack@torchwood.com"
    firstName: "Jack"
    lastName: "Harkness"
    sendCatalog: false
  ▶ __proto__: Object
  valueChanges: (...)
```

Form Model

- Retains form state
- Retains form value
- Retains child controls
  - FormControls
  - Nested FormGroups

Template-driven Forms


```
▼ controls: Object
  ▶ email: FormControl
  ▶ firstName: FormControl
  ▶ lastName: FormControl
  ▶ sendCatalog: FormControl
  ▶ __proto__: Object
  dirty: true
  disabled: false
  enabled: true
  errors: null
  invalid: false
  pending: false
  pristine: false
```

Template

- Form element
- Input element(s)
- Data binding
- Validation rules (attributes)
- Validation error messages
- Form model automatically generated

Component Class

- Properties for data binding (data model)
- Methods for form operations, such as submit

Reactive Forms


```
▼ controls: Object
  ▶ email: FormControl
  ▶ firstName: FormControl
  ▶ lastName: FormControl
  ▶ sendCatalog: FormControl
  ▶ __proto__: Object
  dirty: true
  disabled: false
  enabled: true
  errors: null
  invalid: false
  pending: false
  pristine: false
```

Component Class

- Form model

- Validation rules

- Validation error messages

- Properties for managing data (data model)

- Methods for form operations, such as submit

Template

- Form element

- Input element(s)

- Binding to form model

Directives

Template-driven (FormsModule)

- **ngForm**
- **ngModel**
- **ngModelGroup**

FormGroup

```
<form (ngSubmit)="save()">
</form>
```

Directives

Template-driven (FormsModule)

- **ngForm**
- **ngModel**
- **ngModelGroup**

FormGroup

```
<form (ngSubmit)="save()"
      #signupForm="ngForm">
</form>
```

Directives

Template-driven (FormsModule)

- **ngForm**
- **ngModel**
- **ngModelGroup**

FormGroup

```
<form (ngSubmit)="save()"
      #signupForm="ngForm">
 <button type="submit"
         [disabled]="!signupForm.valid">
   Save
  </button>
</form>
```

Directives

Template-driven (FormsModule)

- **ngForm**
- **ngModel**
- **ngModelGroup**

FormGroup

FormControl

```
<form (ngSubmit)="save()">
  <input id="firstNameId" type="text"
         [(ngModel)]="customer.firstName"
         name="firstName"
         #firstNameVar="ngModel"/>
</form>
```

Directives

Template-driven (FormsModule)	Reactive (ReactiveFormsModule)
• ngForm • ngModel • ngModelGroup	• formGroup • formControl • formControlName • formGroupName • formArrayName

HTML Form

```html
<form>
  <fieldset>
    <div>
      <label for="firstNameId">First Name</label>
      <input id="firstNameId" type="text"
             placeholder="First Name (required)"
             required
             minlength="3" />
    </div>
    ...
    <button type="submit">Save</button>
  </fieldset>
</form>
```

Template-driven Form

```html
<form (ngSubmit)="save()">
  <fieldset>
    <div [ngClass]="{'has-error': firstNameVar.touched && !firstNameVar.valid }">
      <label for="firstNameId">First Name</label>
      <input id="firstNameId" type="text"
             placeholder="First Name (required)"
             required
             minlength="3"
             [(ngModel)]="customer.firstName"
             name="firstName"
             #firstNameVar="ngModel" />
    <span *ngIf="firstNameVar.touched && firstNameVar.errors">
      Please enter your first name.
    </span>
    </div>
    ...
    <button type="submit">Save</button>
  </fieldset>
</form>
```

Reactive Form

```html
<form (ngSubmit)="save()" [formGroup]="signupForm">
  <fieldset>
    <div [ngClass]="{'has-error': formError.firstName }">
      <label for="firstNameId">First Name</label>
      <input id="firstNameId" type="text"
             placeholder="First Name (required)"
             formControlName="firstName" />
      <span *ngIf="formError.firstName">
        {{formError.firstName}}
      </span>
    </div>
    ...
    <button type="submit">Save</button>
  </fieldset>
</form>
```
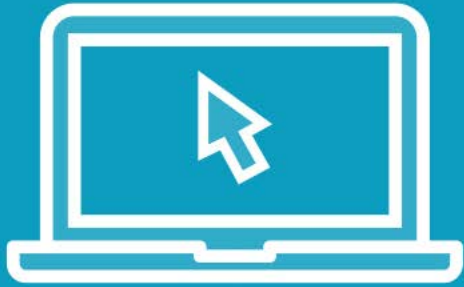
Demo

Template-driven Form

Complex Scenarios

- Dynamically add input elements
- Watch what the user types
- Wait validation until typing stops
- Different validation for different situations
- Immutable data structures

Angular Forms

Template-driven	Reactive
Generated form model	Manually created form model
HTML validation	Validation in the class
Two-way data binding	No two-way data binding

Summary

Angular Form Building Blocks
- FormGroup
- FormControl

Template Syntax for Forms

Template-driven Forms

Complex Scenarios