**Discussion 7.1 – Angular Template-Driven Vs Reactive Forms**

Patrick Cuauro

Bellevue University

WEB 425-307O Angular with TypeScript

Professor Krasso

September 18, 2023

# Template-driven vs Reactive Forms

Almost all enterprise applications are heavily form-driven. Some apps have massive forms that span multiple steps and dialogs and integrate complex validation logic. If developers were to build these complex forms from scratch, they would have to:

* Keep track of the state of the complex, lengthy forms.
* Compute the validity of form sections (with multiple-step forms, this is highly complex).
* Display relevant error messages based on the validation errors. For example, you may have a field with three to five validations on it. For each validation error, you must ensure that the correct error message is displayed, so the user is aware of the error.

These challenges show that building forms for enterprise apps from scratch is inefficient, time-consuming, and error-prone.

Angular provides two modules for creating and managing forms for complex apps. This article will explore the two modules readily available, provide in-depth comparisons to determine when to use each module, and finally look at a demonstration on managing forms using the two modules.

## Template-Driven Forms

A template-driven form is the simplest way to build a form in Angular. It uses Angular’s two-way data-binding directive (**ngModel**) to create and manage the underlying form instance.

Additionally, as the name suggests, a template form is mainly driven by the view component. So, it uses directives placed in HTML rather than TypeScript or JavaScript to manage the form. A template-driven form is **asynchronous** due to the use of “directives” because the creation of form controls is delegated to the declared directives (IoC).

## Reactive Forms

The second way to develop forms in Angular is to use reactive forms. Reactive forms utilize the component class to programmatically declare the form controls and the required validators synchronously.

We then bind the controls to the input fields in the HTML template. Reactive forms use classes such as **FormGroup** and **FormControl** to define groups and controls wired to the template.

Sources:

* Angular Template Driven vs. Reactive Forms

<https://www.syncfusion.com/blogs/post/angular-template-driven-vs-reactive-forms.aspx>