**Angular 7 Forms**

Angular forms are used to handle user's input. We can use Angular form in our application to enable users to log in, to update profile, to enter information, and to perform many other data-entry tasks.

In Angular 7, there are 2 approaches to handle user's input through forms:

* Reactive forms
* Template-driven forms

Both approaches are used to collect user input events from the view, validate the user input, create a form model and data model to update, and provide a way to track changes.

Reactive Forms vs. Template-driven Forms

Both Reactive forms and Template-driven forms manage and process data differently. Each offers different advantages.

**Reactive Forms**

* Reactive forms are more robust.
* Reactive forms are more scalable, reusable, and testable.
* They are most preferred to use if forms are a key part of your application, or your application is already built using reactive patterns. In both cases, reactive forms are best to use.

**Template-driven Forms**

* Template-driven forms are best if you want to add a simple form to your application. For example: email list signup form.
* Template-driven forms are easy to use in the application but they are not as scalable as Reactive forms.
* Template-driven forms are mainly used if your application requires a very basic form and logic. It can easily be managed in a template.

**Difference between Reactive Forms and Template-driven Forms**

|  |  |  |
| --- | --- | --- |
| Comparison Index | Reactive Forms | Template-driven Forms |
| Setup  (form model) | Reactive forms are more explicit. They are created in component class. | Template-driven forms are less explicit. They are created by directives. |
| Data model | Structured | Unstructured |
| Predictability | Synchronous | Asynchronous |
| Form validation | Functions | Directives |
| Mutability | Immutable | Mutable |
| Scalability | Low-level API access | Abstraction on top of APIs |

**Similarity between Reactive Forms and Template-driven Forms**

There are some building blocks which are shared by both reactive and template-driven forms:

**FormControl:** It is used to track the value and validation status of an individual form control.

**FormGroup:** It is used to track the same values and status for a collection of form controls.

**FormArray:** It is used to track the same values and status for an array of form controls.

**ControlValueAccessor:** It is used to create a bridge between Angular FormControl instances and native DOM elements.