**What is NGRX?**

Reactive State for Angular. It’s a state management for Angular applications, inspired by Redux.

**What is Application State? What is Redux?**

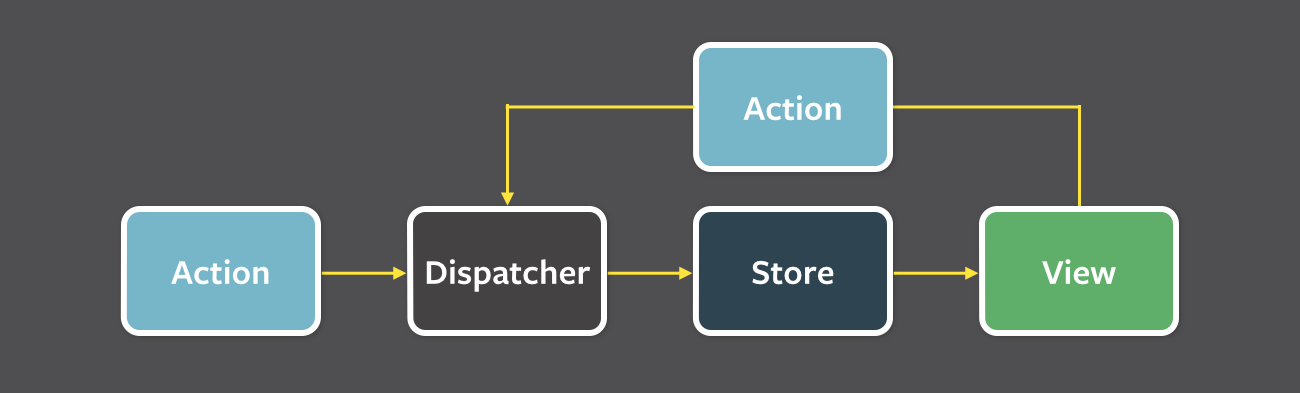
Application State is the current state of your Application per user.

I can compare with old school session that get created in server before this API madness.

Redux is a way to build Application State which based on the Flux Architecture.

**What is Flux?**

Flux is the application architecture that Facebook uses for building client-side web applications.



Data in a Flux application flows in a single direction. That all about flux I know and care.

**Note:**

Unlike Flux, Redux does not have the concept of a Dispatcher. This is because it relies on pure functions instead of event emitters.

A brief history of Redux

Three fundamental principles of Redux:

* Single source of truth:

The state of your whole application is stored in an object tree within a single store.

* State is read-only:

The only way to change the state is to emit an action, an object describing what happened.

* Changes are made with pure functions:

To specify how the state tree is transformed by actions, you write pure reducers.

**Deep Dive into NGRX**

NgRx is a framework for building reactive applications in Angular. It use RxJS to implement the Redux pattern for angular.

**Packages**

* [Store](https://ngrx.io/guide/store) - RxJS powered state management for Angular apps, inspired by Redux.
* [Store Devtools](https://ngrx.io/guide/store-devtools) - Instrumentation for @ngrx/store enabling time-travel debugging.
* [Effects](https://ngrx.io/guide/effects) - Side effect model for @ngrx/store.
* [Router Store](https://ngrx.io/guide/router-store) - Bindings to connect the Angular Router to @ngrx/store.
* [Entity](https://ngrx.io/guide/entity) - Entity State adapter for managing record collections.
* [Schematics](https://ngrx.io/guide/schematics) - Scaffolding library for Angular applications using NgRx libraries.

**Store**

It’s an javascript object which contain the application state.

**Actions**

Actions express *unique events* that happen throughout your application.

It can be user event (like click event), network requests (like api calls) or maybe direct interaction with device API (like geolocation).

**Reducers**

Reducers in NgRx are responsible for handling transitions from one state to the next state in your application. Reducer functions handle these transitions by determining which actions to handle based on the type.

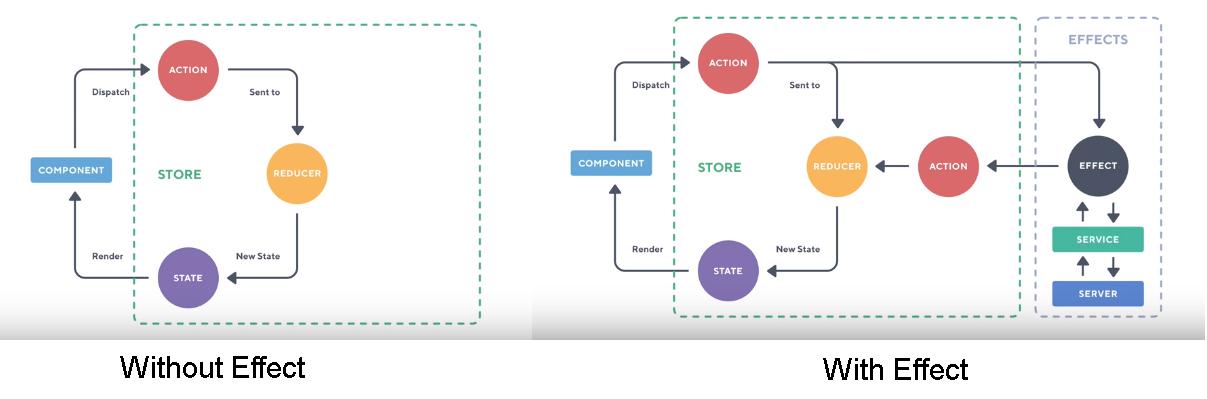
Reducer functions are pure functions in that they produce the same output for a given input.

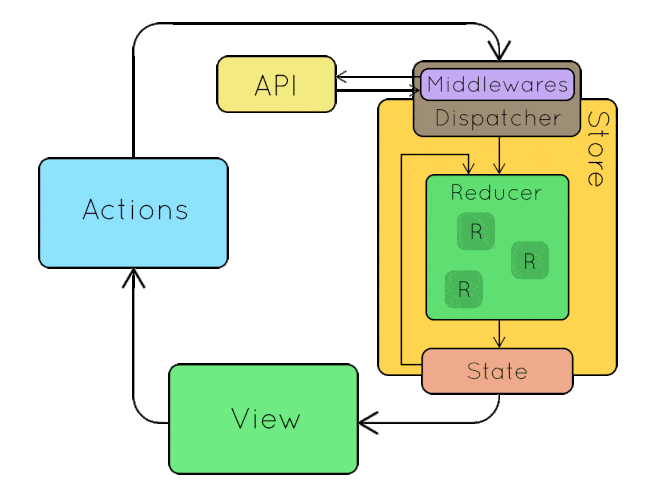
**Selectors**

Selectors are pure functions used for obtaining slices of store state.

**Effects**

RxJS powered side effect model for Store.





**NGRX 101**

Create a new angular project

ng new myngrxapp --style=scss --routing