MEDAC
FUNDACIÓN

Práctica Evaluable 6

Flux, Redux y Mobx

Índice

1. Flux

Flux es un patrón de arquitectura que se comenzó a emplear en Facebook junto con la librería React y se basa en la implementación de capas de JavaScript en aplicaciones de una sola página.

La idea principal detrás de la implementación es resolver los problemas de escalabilidad detrás del modelo MVC (modelo-vista-controlador). Al empezar a desarrollar aplicaciones muy grandes con varios componentes y diferentes manejos de datos, aumentaba la complejidad para gestionar y controlar como los datos eran recibidos por los diferentes componentes y cómo se relacionaban con otros.

Para resolver estos problemas se creó este patrón (fig. 1), basado en una store, dispatchers, vistas y acciones/creadores de acciones. Este patrón, además, encaja muy bien con el sistema de declarativo de React.

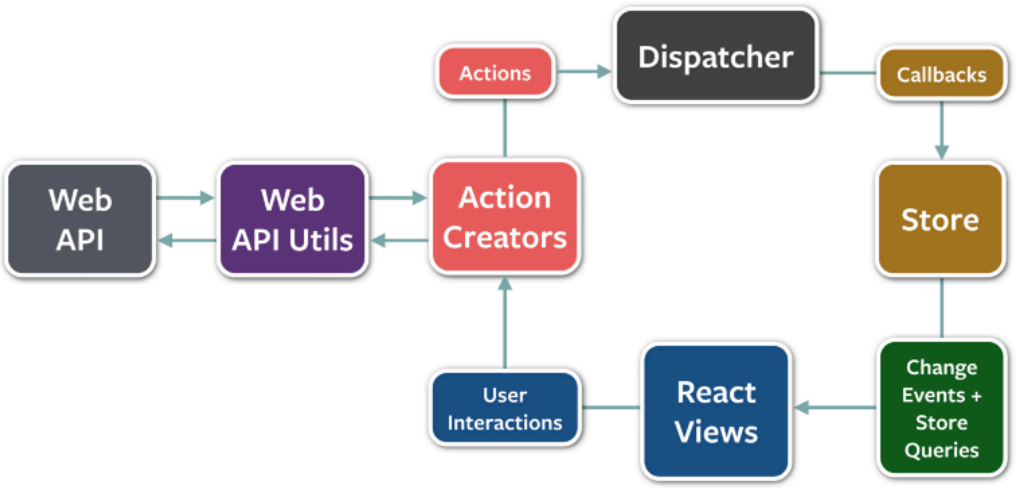

Fig. 1.- Ejemplo de funcionamiento de patrón flux.

Los distintos componentes listados anteriormente en el patrón y su función son:

-Store: Encargado de almacenar y controlar los diferentes estados, tanto en el objeto como en la interfaz.

-Dispatcher: Es el encargado de enviar las acciones que se generan y pueden cambiar estados a los componentes store que necesiten recibirlo.

-vistas: Es el componente encargado de la interfaz de usuario, como interactúa con la aplicación y muestra los cambios de estado.

-Acciones/creadores de acciones: Las acciones son objetos que contienen la información necesaria para realizar los cambios de estado y los creadores de acciones son la encapsulación de ese código en funciones que producen ese cambio de acción.

2. Redux

Redux es una librería que surge como una evolución de Flux pero reduciendo su complejidad al tomar ideas del lenguaje Elm. Redux se basa en tres principios: sólo hay una store, es de lectura y los cambios de estado se realizan a través de funciones, por lo que el esquema sería el usuario realiza una acción, gestionada por el reducer, esta es recibida por la store, modificando el estado y se altera en la vista (fig. 2).

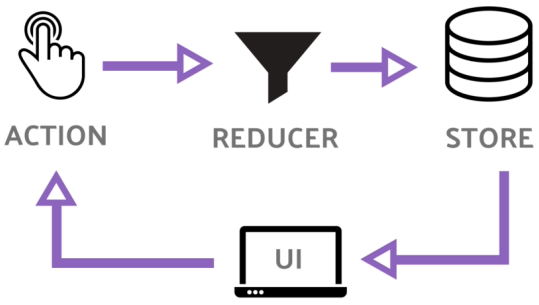

Fig. 2.- Patrón de comportamiento Redux.

Los principales componentes serían:

-acciones/creadores de acciones: son las interacciones que el usuario hace en la interfaz, enviando la información necesaria al reducer.

-reducer: objetos encargados de alterar el estado en función de las acciones realizadas por el usuario.

-store: objeto encargado del control del estado global de la aplicación, acceso a la información de los diferentes estados, registro de listeners y permitir la modificación a través de los reducers.

3. Mobx

Mobx es una librería para la gestión de estados a través de la implementación de programación funcional reactiva que prevé las estados inconsistentes a través de la automatización de las derivaciones. Puede emplearse de manera aislada pero es usada principalmente en frameworks de front-end, como React, Vue o Angular.

Los principales conceptos de Mobx son observables, acciones y reacciones y como se relacionan entre ellos (fig. 3).

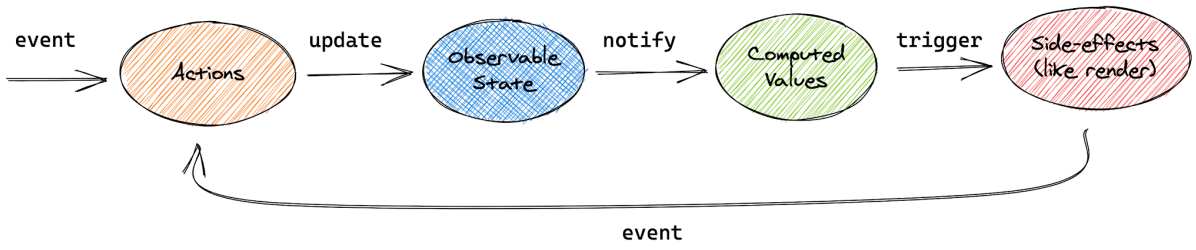

Fig. 3.- Flujo de trabajo en Mobx.

Los principales componentes son;

-observables: representan el estado reactivo de la aplicación, compuestos de los posibles estados que pueden tener la aplicación, conocidos como core-state y derived-state.

-acciones: Interacciones del usuario que modifican los observables.

-reacciones: Indican como las acciones modifican los observables.

Bibliografía

https://www.freecodecamp.org/news/an-introduction-to-the-flux-architectural-pattern-674ea74775c9/

https://medium.com/@w3bh4ck/the-flux-architecture-pattern-for-frontend-development-1f2dae32b789

https://leanpub.com/react-redux/read

https://es.redux.js.org/

https://pub.dev/packages/mobx

https://medium.com/@yogeshwaranramesh/introduction-to-mobx-2dd276e10e07