Intro To Redux

What is Redux?

Redux is a predictable state container for JavaScript apps. Redux is mostly used for application state management. In a nutshell, Redux maintains the status of the entire application in a single unchanged state tree (object), which cannot be changed directly. When anything changes, a new object is created (using actions and reducers).

Why Redux is used?

- **Always Predictable:** The state in redux is always predictable as a state and are immutable and can't be changed. If the same state and actions are passed to redux always the same output comes because reducers are pure functions.
- **Easily Maintainable:** Redux is strict towards organising its code so it is very easy for someone who has knowledge of redux to understand the application. Hence this is why redux is easy to maintain.

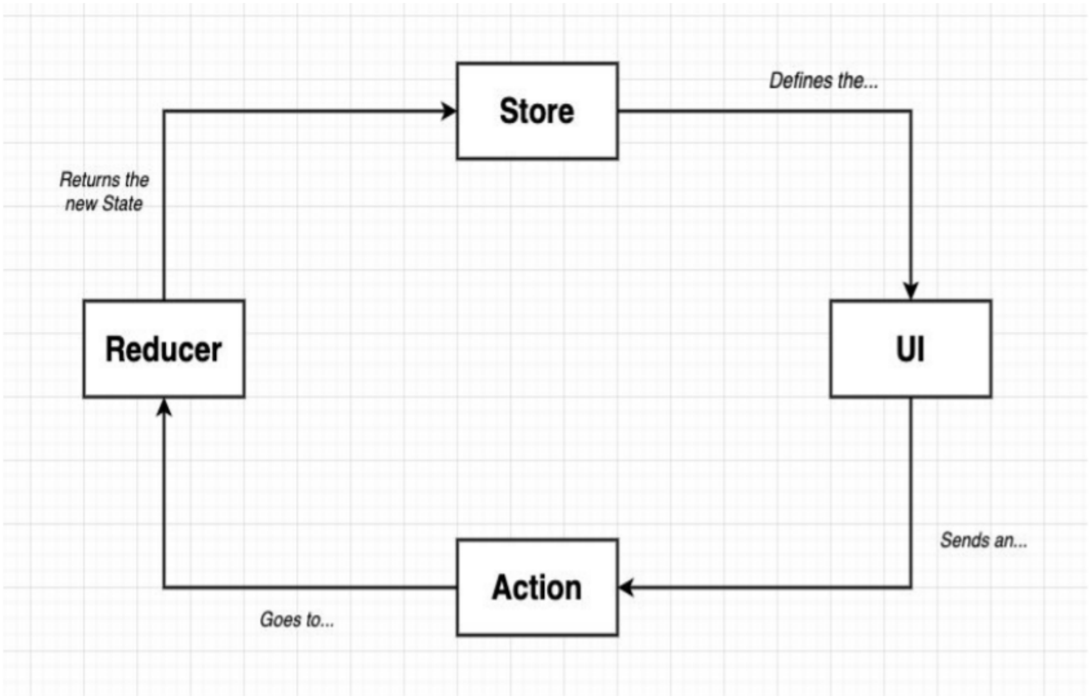
When to use redux?

Redux allows you to manage the state of your application in **one place** and keep changes in your application more capable, scalable and recognizable. This will make it easier to argue about the changes taking place in your application. But all these benefits come with trade-offs and hurdles. It may seem to one that it adds boilerplate plate code, making simple things a little heavier; But it depends on the architectural decisions.

A simple answer to this question is that when you need Redux, you will realize it yourself. If you're still confused as to whether you need it, you can not. This usually happens when your app has grown to a level where it has become difficult to maintain the app state and you are looking to make it easy and simple.

How Redux Works?

The way in which Redux works is very **simple**. There is a **central store** that contains the overall status of the application. Each component can access the stored state without sending credentials from one component to another. The container components do not communicate directly with each other by passing callbacks and props from parent to child. A rough flow is proposed by Redux: The callbacks create and dispatch actions based on the request sent by UI. Reducers process actions hence executing the new state.



Reducers

Reducer is a function that determines the change in the state of an application. It uses the obtained action to determine this change. Redux relies heavily on Reducer functions that perform an action to execute the previous state and the next state.

How does the reducer function work?
State changes are based on user interaction or network request. If the state of the application is maintained by Redux, changes will occur within the **reducer function**. The reducer function uses the initial state of an application called **action** to determine how the new state or final state will look alike.

Summarising It

Let's summarise what we have learnt in this module:

- Learned about what is redux
- Why Redux is used and when
- How Redux works
- A small brief about reducers

Some References:

● Principles of Redux
https://redux.js.org/introduction/three-principles
● When to use Redux
https://blog.logrocket.com/when-and-when-not-to-use-redux-41807f29a7fb/