**State management library such as Redux.**

**Two Principle of React used in the Redux:**

1. **Unidirectional data flow:**

In a React application, data flows from parent components to child components via props.

2. **Applications can be structured around Container and Pure components.**

This is not a React characteristic, but rather a stablished best practice.

Pure components are concerned with how things look, rendering DOM markup.

Container components are responsible for providing the data and behavior to other components. They usually don’t have any DOM markup of their own and instead render presentational components

To begin creating React+Redux applications, there are three pieces that you need to understand: The **store**, **actions** and **reducer** functions:

**Store:**

Redux provides a store to hold the entire application state in a single, centralized JavaScript object. The store can be directly accessed by any container components in the application:

The store exposes three methods you can call from your container components:

• **getState**: Used by container components on any hierarchy level to read the current state from the store.

• **subscribe**: React Container components can subscribe to the store to be notified of state changes and re-render themselves - as well as their children.

• **dispatch**: Used to dispatch an action to the store. Dispatched actions are used to trigger state changes.

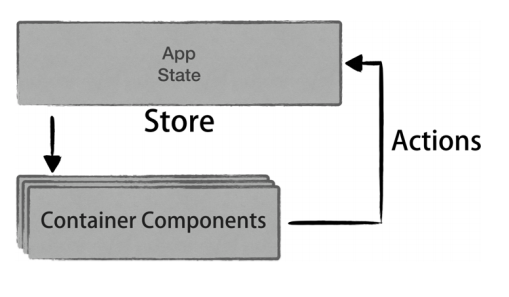
It is important to stress that a React+Redux applications always contain only one store, and all your application state lives in a single object in this single store

**Actions**

The store is read-only – no part of the application can change the state inside it (or, to be precise, the only part of the application that can update the store **is the store itself**.).

Only the store can update its state, but it provides a mechanism by which any other part of the application can indicate that the state needs to change: **dispatching actions**

Dispatching an action is like sending a message to the store saying that something happened and that the store should update itself in response.



**Inside the Store: Reducer functions:**

The third piece in Redux are the reducer functions. They are used internally by the store and are responsible for mutating the store state when actions are dispatched.

#################33

Q) if we will define store.subscribe and store.dispatch method together in the same file and subscribe declaration is after dispatch then it don’t get call but if dispatch is called from different file and subscribe is in different file then subscribe method is called or we have to defined it above dispatch call method..