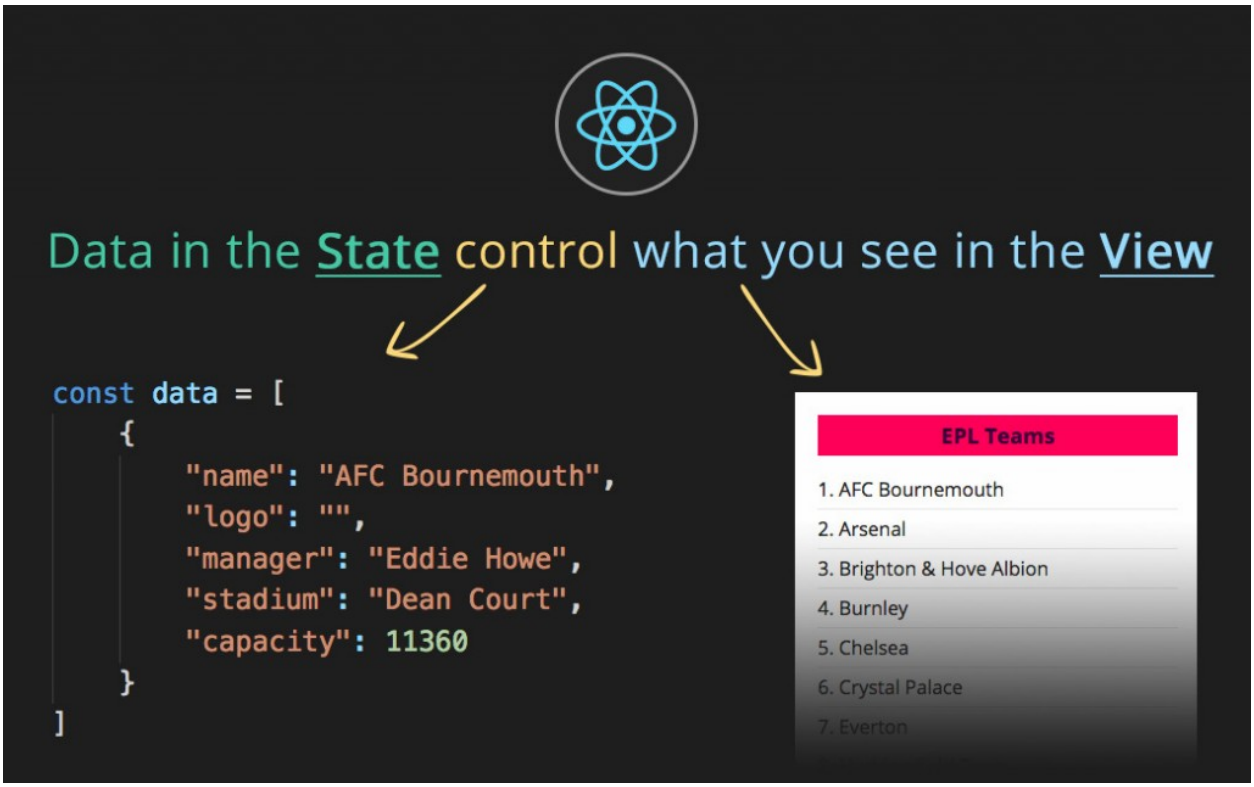
State Management and props

Date	@February 14, 2023 → February 20, 2023
Assign	
Status	
↗ ✅ Tasks	

State in react



> image from https://ihatetomatoes.net/react-state-management-tutorial-do-you-really-need-redux-or-mobx/

It refers to variables that contain information, we can look at it like the component’s or the application’s memory, in our case we can think of it as a variable that we use to modify what we see on screen based on the user’s interaction.

Importance of State management.

As stated before, the state is what contains the current “state” of the app, is it using light or dark theme, the information the user saved in a form, is the button disabled? all those thing and more are part of what we save in the state, and managing that information is important, knowing when to use local or global state and the different ways to share that state defines how we interact with a components lifecycle and how each component interacts with others.

Why do we need State Management?

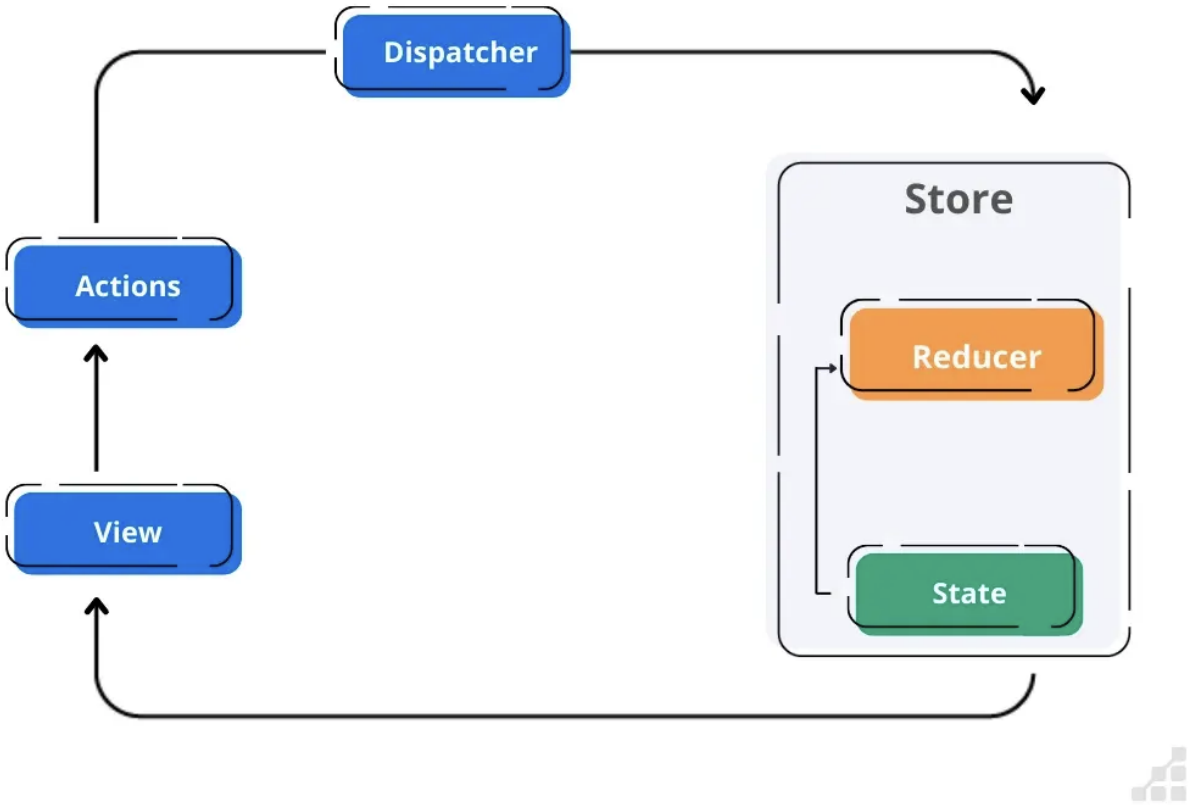
We have many different things we can handle using state, and because of that there are many ways to handle state… we can have local state in a component, for example the open/closed state on a burger menu component can be local since we don't need to share that information with the rest of the app, but the user information will probably be used in many pages or components where we need to access stuff like permissions information or contact information, that should be global.

So, when and where do we use one or the other? the answer is the same as to many other questions: it depends, you, as a developer should ask yourself or the PM the questions that help you make an informed decision.

- Where am I going to use this information?
- Can this be fetched from an API?
- Is this info going to change over time?

Redux vs Context API

Redux

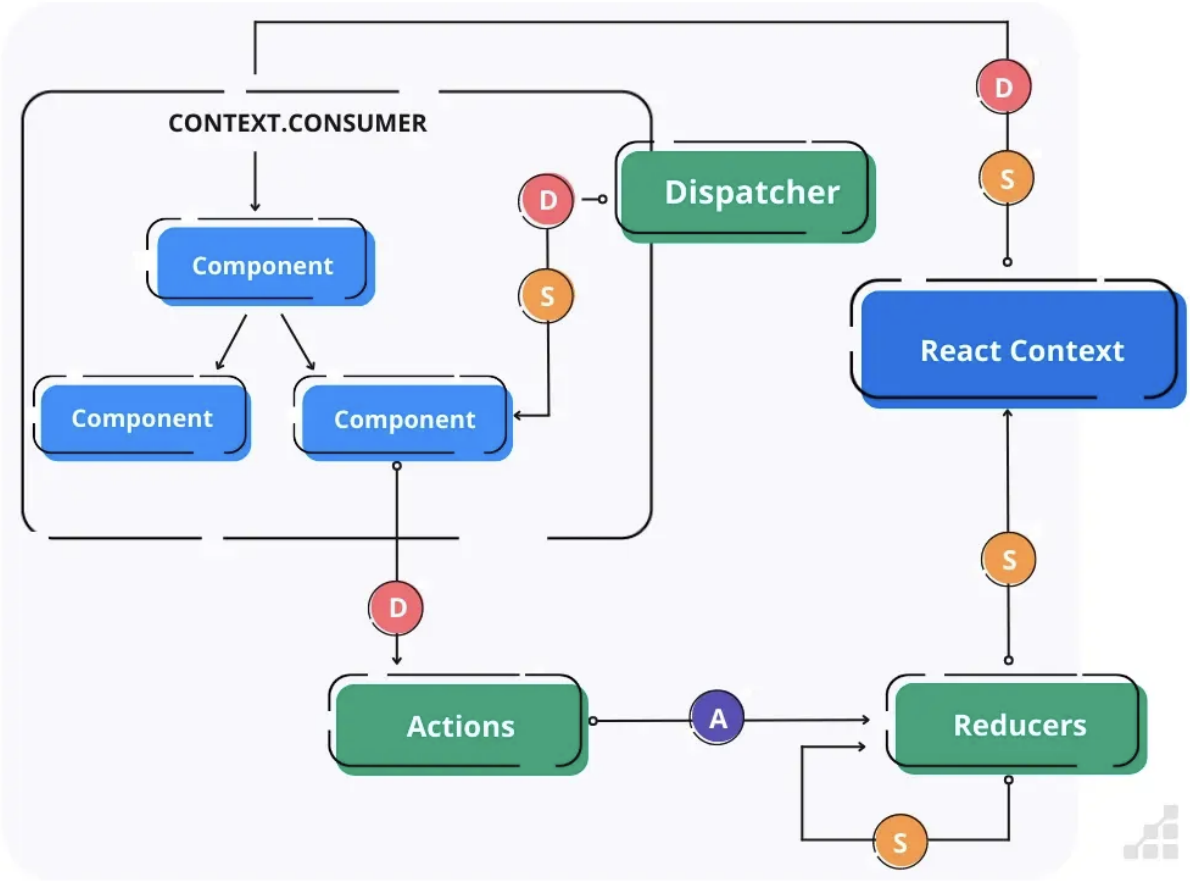


> image from https://www.scalablepath.com/react/context-api-vs-redux

Redux is a library that focuses on managing and updating the state, it uses Actions, reducers and dispatchers in order to maintain a coherent and easy to maintain structure where the developer can have freedom to write the code and ease of debugging.

the way it works is pretty straightforward: the view dispatches actions that represent a change in the state, the store then handles that changes with the reducer and informs the view.

Context API



> Image from https://www.scalablepath.com/react/context-api-vs-redux

it is a dependency injection mechanism, which means its main function is to be used to pass information from the state to different components on the tree, the way to use it is: you create a context using createContext, and it will provide you with a provider and a consumer.