

# WHAT IS REDUX?



codewithgauri

swipe >>

# INTRODUCTION :

Redux is a predictable state container designed to help you write JavaScript apps that behave consistently across client, server, and native environments and are easy to test.



In short, Redux is a way to manage the “state” or we can say it is a cache or storage that can be accessed by all components with a structured way. It has to be accessed through a “Reducer” and “Actions”



# What is state management in Redux?

State management is essentially a way to facilitate communication and sharing of data across components. It creates a tangible data structure to represent the state of your app that you can read from and write to. That way, you can see otherwise invisible states while you're working with them.

As the management gets messy as the app gets complex. This is why you need a state management tool like Redux that makes it easier to maintain these states. Let's get a good overview of Redux concepts before considering its benefits.



# Actions in Redux :

Simply put, actions are events. They are the only way you can send data from your application to your Redux store. The data can be from user interactions, API calls, or even form submissions.

Actions are sent using the `store.dispatch()` method. Actions are plain JavaScript objects, and they must have a `type` property to indicate the type of action to be carried out.

Here's an example of an action that can be carried out during login in an app:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. It contains a JavaScript object representing a Redux action.

```
{  
  type: "LOGIN",  
  payload: {  
    username: "foo",  
    password: "bar"  
  }  
}
```



# Reducers in Redux:

Reducers are pure functions that take the current state of an application, perform an action, and return a new state. These states are stored as objects, and they specify how the state of an application changes in response to an action sent to the store

Here is an example of how reducers work in Redux:

```
const LoginComponent = (state = initialState, action) => {
  switch (action.type) {

    // This reducer handles any action with type "LOGIN"
    case "LOGIN":
      return state.map(user => {
        if (user.username !== action.username) {
          return user;
        }

        if (user.password === action.password) {
          return {
            ...user,
            login_status: "LOGGED IN"
          }
        }
      });
    default:
      return state;
  }
};
```

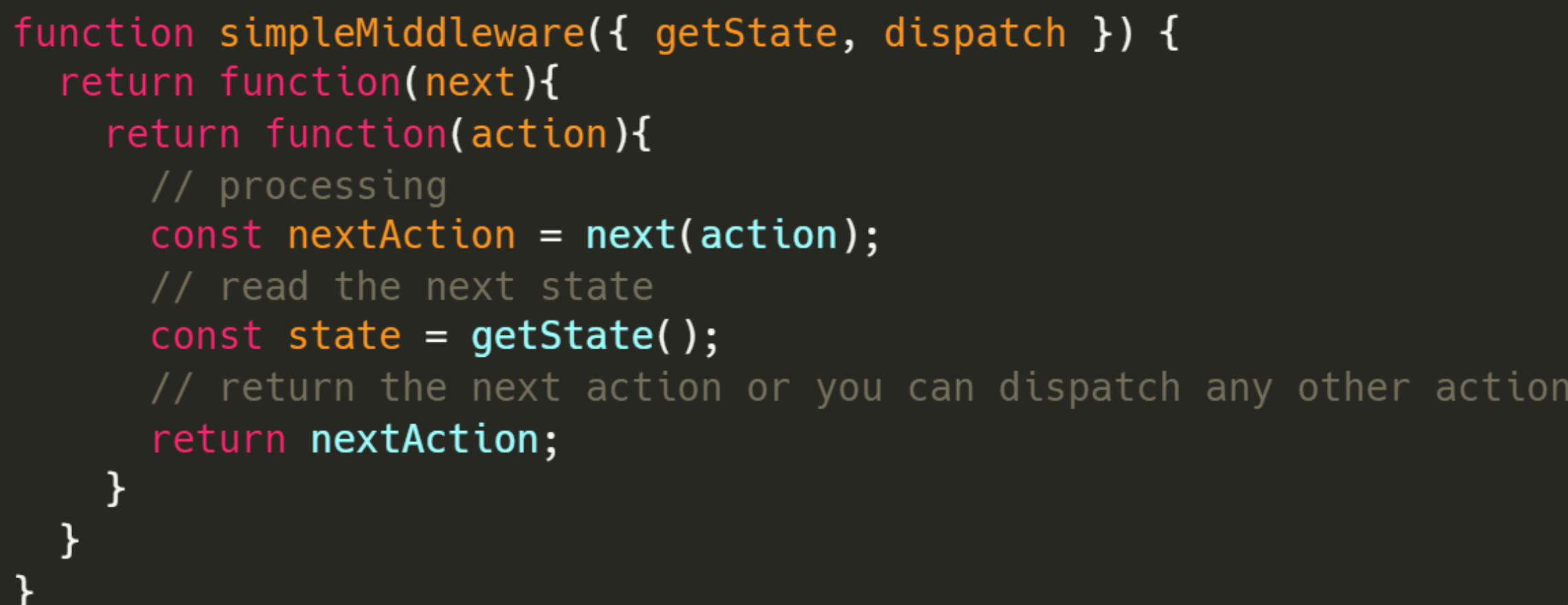


# Redux middleware:

Redux allows developers to intercept all actions dispatched from components before they are passed to the reducer function. This interception is done via middlewares.

Building on the example Login component discussed in the last section, we might want to sanitize the user's input before it reaches our store for further processing. This can be achieved via Redux middleware.

Here's what a simple middleware looks like:



```
function simpleMiddleware({ getState, dispatch }) {  
  return function(next){  
    return function(action){  
      // processing  
      const nextAction = next(action);  
      // read the next state  
      const state = getState();  
      // return the next action or you can dispatch any other action  
      return nextAction;  
    }  
  }  
}
```





# Why use Redux?

When using Redux with React, states will no longer need to be lifted up. This makes it easier for you to trace which action causes any change.

As you can see in the example above, the component does not need to provide any state or method for its children components to share data among themselves. Everything is handled by Redux. This greatly simplifies the app and makes it easier to maintain.

This is the primary reason why you should use Redux, but it's not the only benefit. Take a look at the list below for a summary of what you stand to gain by using Redux for state management.



And for amazing stuff you can follow me



# Gaurav Pandey

LinkedIn :Gaurav Pandey

Twitter : @gauravcode

#### References:

Why use Redux? A tutorial with examples - LogRocket Blog

What Is Redux?. You might have heard the word Redux... | by Rany ElHousieny | The Startup | Medium