React.js

Lecture 5

Agenda

- Recap last lecture points
- Redux Thunk
- Context
- Redux vs Context
- Apply context instead of redux
- Open discussion and questions

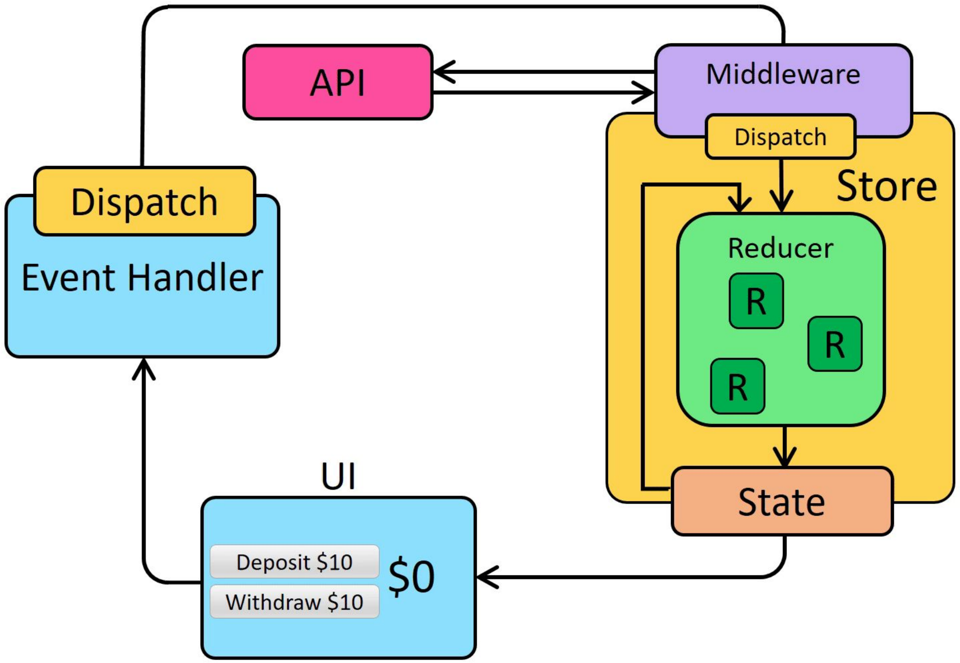
Redux thunk

For more info check [this link]

Redux Thunk is middleware that allows you to return functions, rather than just actions, within Redux. This allows for delayed actions, including working with promises.

```
npm install redux-thunk
```

Redux with middleware



Redux with middleware: Store file

```
import { createStore, applyMiddleware, compose } from "redux";
import { composeWithDevTools } from "redux-devtools-extension";
import reducers from "./combineReducers";
import thunk from "redux-thunk";

const store = createStore(reducer,
composeWithDevTools(applyMiddleware(thunk)));

export default store;
```

Redux with middleware: Store file

```
export const getMovies = () => (dispatch) => {
//nameless functions
    // Return promise with success and failure
   return axiosInstance.get("/movie/popular").then(
     (res) => dispatch({ type: 'GET_MOVIES', payload :
res.data.results }),
     (err) => console.log("err")
   );
};
```

Redux with middleware: Store file

```
export const actionName = () => async (dispatch) => {
  try {
    const response = await axios.get(`api`);
    dispatch({
      type: type,
      Payload: response.data,
    });
  } catch (err) {
    console.log(err);
  }
};
```

Context

Context provides a way to pass data through the component tree without having to pass props down manually at every level.

When to Use Context

Context is designed to share data that can be considered “global” for a tree of React components, such as the current authenticated user, theme, or preferred language.

Context VS Redux

- If you are using Redux only to avoid passing props down to deeply nested components, then you could replace Redux with the Context API. It is exactly intended for this use case.

- On the other hand, if you are using Redux for everything else (handling your application's logic outside of your components, centralizing your application's state, using Redux DevTools to track when, why, and how your application's state changed, or using plugins such as Redux Form, Redux Saga, etc…), then there is absolutely no reason for you to abandon Redux.

  The Context API doesn't provide any of this.

Context : Create context

Create context file config :

```
import React from "react";

const contextFeature = React.createContext();

export const contextProvider= contextFeature.Provider;

export default contextFeature ;
```

Context : Use context

- **Wrap the components that needs to have the provider values with context provider:**

```
const [value, setValue] = useState(initial value);
<ContextProvider  value={{ value, setValue}}>
    <clild />
</ContextProvider>
```

- **In child component you can use and set context value using useContext hook**

```
const {value , setValue} = useContext(contextFeature)
```

Check the following for class components with context : Link

Resources

React links :

- https://www.youtube.com/watch?v=Dorf8i6lCuk
- https://scrimba.com/learn/learnreact
- https://www.youtube.com/watch?v=w7ejDZ8SWv8&t=2559s
- https://www.youtube.com/watch?v=0riHps91AzE
- Chat with firebase https://www.youtube.com/watch?v=zQyrwxMPm88
- Medhat Dawoud Channel : https://www.youtube.com/channel/UCuwTHYdMavwEPsZ6OAkXfig
- Mosh channel : https://www.youtube.com/channel/UCWv7vMbMWH4-V0ZXdmDpPBA
- Formik in deep : https://www.youtube.com/watch?v=a94FOvaBomQ&list=PLC3y8-rFHvwiPmFbtzEWjESkqBVDbdgGu
- React router V6 : https://www.youtube.com/watch?v=zEQiNFAwDGo&t=84s

Thank you

Lecture 5
LAP

Task : Movies Using Redux

Change the movies API to use Redux thunk to get list of movies.

Task :Movies App

Change the app language Using the Context to save the latest Value and call the API again when the language changed with the language query.