React
שיעור חמישי

React

React Redux

- React Redux מאפשרת לנו לנהל state גלובלי באפליקציה שלנו. זאת אומרת, שבמקום להתחיל להוריש states לאורך כל האפליקציה, נוכל לשים state מסויים שיש לנו צורך בו בכל האפליקציה ואז נוכל לגשת אליו מכל קומפוננט, ללא צורך בהורשות.
- בשביל זה צריך להוריד שתי ספריות : redux
```
t\react_course> npm i redux
```
ו react-redux
```
\react_course> npm i react-redux
```
שתפקידה הוא לחבר את האפליקציית ריאקט שלנו לרידקס.
- איך זה עובד? נניח באפליקציה שלנו קיים כפתור שבלחיצה עליו נרצה לעדכן ערך מסוים בכל העמודים בקומפוננט – נפעיל סוג של פונקציה המעדכנת את הנתון הגלובלי. בשביל זה צריך כמפורט בשקופית הבאה.

איך זה עובד?

- **Store** - "חנות" המאכסנת את הנתונים הגלובליים.
Reducer –מכיל את ה state, מעדכן אותו למצב החדש בהתאם למה שנשלח אליו.
Action – מורה לReducer את הפעולה שיש לבצע על ה state. ה reducer מבצע את הפעולה בפועל אבל איזו פעולה ואופן הביצוע – על זה אחראי ה Action.
Dispatch - שליחת הפעולה ל Action בשביל ביצוע הפעולה ב Reducer. – זו הדרך היחידה לעדכן את ה state – קריאה ל dispatch עם שליחת הפעולה. ולמה לא ניתן פשוט לשנות את ה state ולהפעיל Action במקום שצריך? כי אם לא נשלח נוכל לשנות את ה state מכל מקום באפליקציה שלנו ולא נדע מאיפה הפעולה באה. כתוצאה מכן redux יודע מתי ואיך השתנה ה state וזה מקל בפתרון הבאגים.
Middleware – נבין מה מטרתו לפי העניין הבא: נניח שה Action שולח ל Reducer מה לעשות, לא בצורת פרמטר אלא בצורת פונקציה. במקרה זה ה Reducer לא ידע איך לפעול כי לא יכול לתרגם את הפונקציה לפעולה שצריך לעשות. בשביל זה יש את ה Middleware שתפקידו כשלב ביניים בין שליחת ה Actionל הפעלתו ב Reducer– היא מזהה האם נשלח פונקציה ואם כן, מפעילה ומתרגמת את הפונקציה לפעולה אותה שולחת ל Reducer כ Payload.

קבצים

- לאחר התקנות ניתן לכתוב redux☺ נוסיף בפרויקט שלנו (בתיקיית ה src) תיקייה redux המכילה את התיקיות והקבצים הנצרכים כמפורט:
- קובץ store.js – יוצרת את ה"חנות" של ה reducers. בשביל שימוש ה states בקומפוננטות.
- תיקיית reducers – תכיל את כל ה reducers.
- תיקיית actions – תכיל את הפעולות השונות.
- (לא חייב בתוך תיקיית ה redux) קומפוננט בה נשתמש ב store, reducer, actions.

כתיבת reducer

- Reducer זה פונקציה המחזירה state חדש בהתאם לפעולה שקיבלה לעשות עליו.
  פונקציה זו מקבלת state ו action, לפי סוג ה action משנה את ה state.
  לדוגמא:

```
const inititalState = { name: "User 1", age: 23 };
const userReducer = (state = inititalState, action) => {
  switch (action.type) {
    case "CHANGEUSERNAME": {
      return { ...state, name: action.payload };
    }
  }
  return state;
};
export default userReducer;
```

- בודקים אם סוג ה action הוא CHANGEUSERNAME ובהתאם לזה – נשנה את שם ה user ב state לפי ה payload שהפונקציה קיבלה.

כתיבת reducer - המשך

- ניתן להוסיף כמה cases שצריך בהתאם לדרישות, ה payload שנשלח לפונקציה יתאים לסוג אותו הפונקציה צריכה לבצע.

```
switch (action.type) {
  case "CHANGEUSERNAME": {
    return { ...state, name: action.payload };
  }
  case "CHANGEAGE": {
    return { ...state, age: action.payload };
  }
}
```

- פונקציה זו צריכה שיהיה לה export – לייצא אותה.

```
const inititalState = { name: "User 1", age: 23 };
const addUserReducer = (state = inititalState, action) => {
  switch (action.type) {
    case "CHANGEUSERNAME": {
      return { ...state, name: action.payload };
    }
    case "CHANGEAGE": {
      return { ...state, age: action.payload };
    }
  }
  return state;
};
export default addUserReducer;
```

combineReducers

- רוב האפליקציות ככולם צריכות לשמור יותר מstate אחד. יצירת מספר states באותו reducer הוא לא נוח וקשה לתחזוקה. בשביל זה נוצר combineReducers: לנהל מספר reducers נפרדים לכל state.
- פונקציית העזר combineReducer עוזרת להעביר ל store יחיד את כל ה reducers בפרויקט ע"י שנדאג להצהיר על כל אחד מהם בפונקציה זו.
- פונקציה זו combineReducxer היא חובה, אחרת ה store לא יוכל להכיר ולהפעיל את ה reducers.
- בד"כ נשים אותה בקובץ נפרד בתיקיית ה reducers לדוגמא:

מספר Reducers

```
import { combineReducers } from "redux";
import userReducer from "./userReducer";
import productReducer from "./productReducer";

const allReducers = combineReducers({
  userReducer: userReducer,
  productReducer: productReducer,
});

export default allReducers;
```

בודד Reducer

```
import { combineReducers } from "redux";
import userReducer from "./userReducer";

const allReducers = combineReducers({
  userReducer: userReducer,
});

export default allReducers;
```

React

כתיבת Action

- ה Action הוא פונקציה המקבלת משתנה – בו משתמשים ב reducer לשנות את ה state. ומחזירה את סוג הפעולה , לפיה reducer יודע איזו פעולה לבצע, ואת ה payload –אותו ה reducer מקבל ומבצע את פעולת שינוי ה state עליו.

```
export const changeUsersName = (name) => {
    return {
      type: "CHANGEUSERNAME",
      payload: name,
    };
  };
```

כתיבת store

- תפקיד ה store – להוסיף את ה reducer ל state. בתוך ה reducer מוגדרים הstates שה reducer צריך כדי לעדכן ולשנות. ה store מכונה "חנות" - מאכסן את כל ה reducer רק כך נוכל להכיר ולהשתמש ב states ב components.

```
import { createStore } from "redux";
import allReducers from "./reducers";

const store = createStore(
  allReducers,
);
store.getState();
export default store;
```

הפעלת הקופוננט

- ל react redux יש פונקציה מובנית בשם useSelector() המאפשרת לנו לקבל את ה states מה store בכל מקום שנרצה באפליקציה שלנו.
- בשביל זה יש לעטוף הפעלת כל קומפוננט שצריכה שרות זה בתגית Provider המגיעה מ react-redux שאומר ל react שאנחנו רוצים להיות מסוגלים להשתמש ב redux בכל אחד מקומפוננטות אלו. בתוכו נשים את אובייקט ה store – נותן גישה לreducers המאוכסנים ב store בקומפוננטות אלו.

import

```
import {Provider} from 'react-redux';
import store from './redux/store';
```

return

```
<Provider store={store}>
  <UserDetails></UserDetails>
</Provider>
```

הפעלת ה state בקומפוננט

- כעת ניתן להשתמש ב states בכל מקום בו נרצה בקומפוננטות העטופות.
- נייבא את הספריה useSelector() מ react-redux

```
import { useSelector } from "react-redux";
```

- קבלת ה state למשתנה באמצעות useSelector():

```
const user = useSelector((state) => state.userReducer);
```

- הצגת תוכן ה state:

```
<h1>Name: {user.name}</h1>
<h1>Age: {user.age}</h1>
```

הפעלת useDispatch

- נייבא useDispatch מ react-redux

```
import { useSelector, useDispatch } from "react-redux";
```

- יצירת אובייקט מ useDispatch

```
const dispatch = useDispatch();
```

- הפעלת ה action באמצעות ה dispatch באופן הבא במיקום המתאים:

```
onSubmit={(e) => {
    e.preventDefault();
    dispatch(changeUsersName(name));
    dispatch(changeUsersAge(age));
}}
```

שיעורי בית:

- יש ליצור שני states גלובליים (שני reducers) אחד המכיל אובייקטים של ספרים ואחד של לקוחות. (מערכי אובייקטים)
- באובייקטי הספרים יהיה id שהוא מכיל את id היוזר. (לכל ספר מספר של Id של יוזר מסוים.)
- יצירת actions עם אפשרות שינוי ספרים (שיהיה ניתן לשנות את id הלקוח באובייקטים.
- יצירת קומפוננט עם האפשרויות הבאות: הצגת רשימת הספרים.
רשות: אפשרות הצגת ספרים לפי id מסוים של היוזר. (לפי המספר שנשים ב input – id של היוזר, יוצגו כל הספרים של לקוח זה.

בהצלחה!!

React