תרגול react-redux

action אסינכרוני, איך לסדר את ה- component , state
"חכמות" ו"טיפשות"

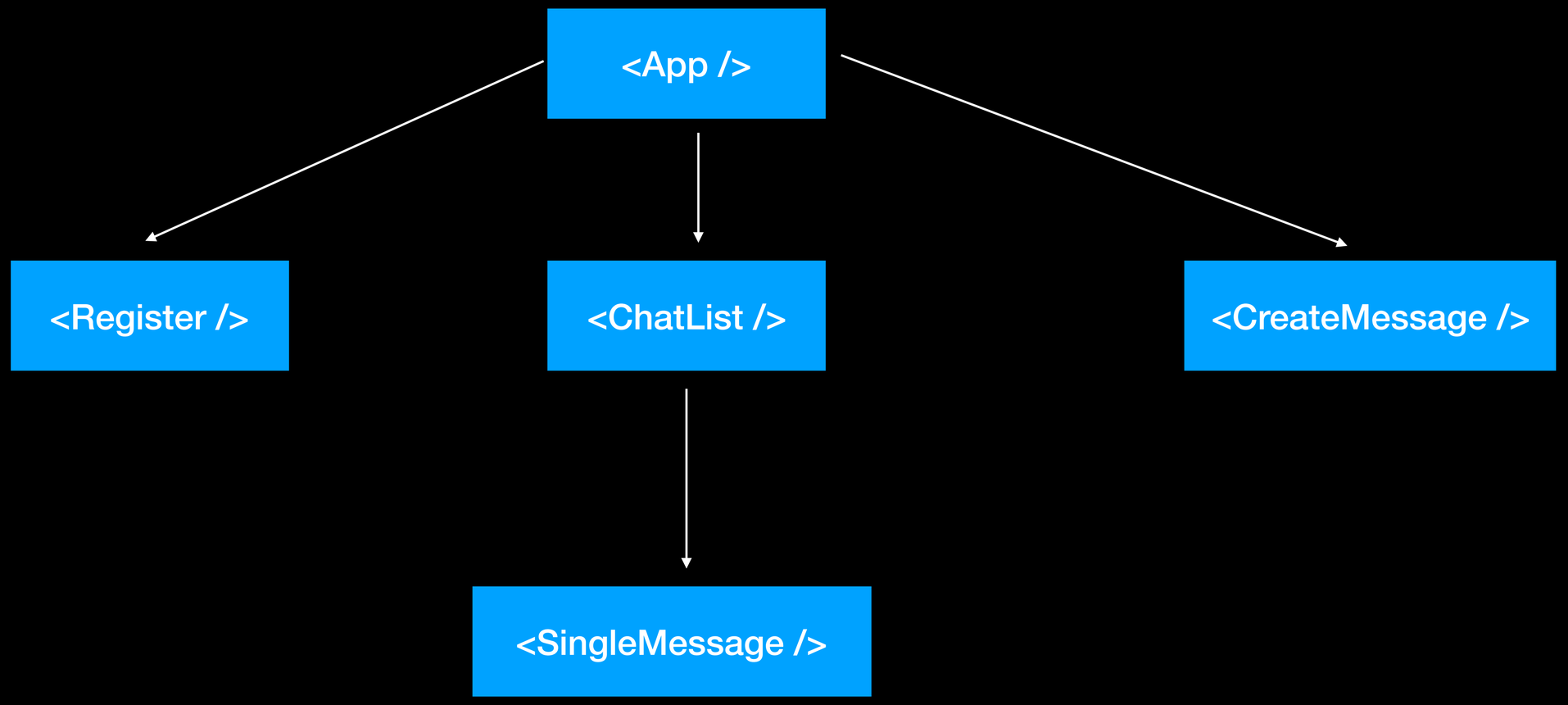
מטרת התרגול

- נוחות בעבודה עם שרת
- אתחול ה- state משרת
- סידור ה- components לחכמות וטיפשות
- איך מסדרים נכון את ה- state

שאלה:

מה אנחנו ניצור בתרגול היום?

עץ ה- components



שרת ה- chat

- השרת יושב בכתובת:
  - https://academeez-chat.herokuapp.com
- השרת הוא שרת REST שמחזיר מידע בפורמט של JSON
- השרת מחובר לבסיס נתונים שמכיל שתי טבלאות
  - users - /api/users/
  - messages - /api/messages/

שרת ה- chat

- השרת יושב בכתובת:
  - https://academeez-chat.herokuapp.com
- השרת הוא שרת REST שמחזיר מידע בפורמט של JSON
- השרת מחובר לבסיס נתונים שמכיל שתי טבלאות
  - users - /api/users/
  - messages - /api/messages/

users - /api/users/

קריאת כל המשתמשים:

Request type: GET
Request path: /api/users/

הוספת משתמש חדש:

Request type: POST
Request path: /api/users/
Request body:
{firstName: ‘Yariv’, lastName: ‘katz’}

messages - /api/messages/

קריאת כל ההודעות:

Request type: GET
Request path: /api/messages/

הוספת משתמש חדש:

Request type: POST
Request path: /api/messages/
Request body:
{message: ‘…’, userId: 20}

איך ה- state צריך להיראות

```
state = {
    users: {
        users: {
            1: {firstName: ‘yariv’, lastName: ‘katz’, …},
            2: {firstName: ‘foo’, lastName: ‘bar’, …},
             ….
        },
        selectedRight: 1,
        selectedLeft: 2
    },
    messages: {
        messages: {
            1: {message: ‘hello’, userId: 1, …}
            2: {message: ‘world’, userId: 2, …}
        }
    }

}
```

הערות נוספות

- שימו לב component שמחובר ל- redux הוא פחות reusability לאורך אפליקציות נוספות ועל כן יש להקפיד על חלוקה נכונה של containers and dumb components
- את העיצוב אנחנו נעשה בעזרת Bootstrap
- תחברו את Redux ל - Redux dev tools
- העבודה עם השרת תצריך מאיתנו להתקין ל-redux את ה - middleware שנקרא redux-thunk
- את הקריאות לשרת שימו בתוך singleton services

עצרו את הוידאו ונסו לבצע את
התרגיל בעצמכם.
הפתרון בהמשך...

class UserService

- מכיל את התקשורת עם שרת ה- REST ל- API של ה- users
- Singleton
- מכיל method ליצור משתמש חדש
- מכיל method לקחת את כל המשתמשים מהשרת

<Register />

- מכילה טופס הרשמה שבו המשתמש מזין את השם הפרטי ושם המשפחה שלו.
- את ניהול הטופס נבצע באמצעות Formik
- במילוי הטופס צריך לשלוח בקשה לשרת ליצור משתמש חדש
- dumb component - לא מחובר ל- Redux
- מקבל מהאבא פונקציית callback וקורא לה עם המשתמש שיצרנו.

users.actions.js

- קובץ המכיל actions שקשורים ל- users
- מכיל שלושה actions:
  - setUsers - יאתחל את רשימת כל ה- users
  - setSelected - יאתחל את המשתמש הנבחר בצד ימין ובצד שמאל
  - fetchUsers - action אסינכרוני שמביא מהשרת את כל המשתמשים

users.reducer.js

- state התחלתי יכיל את התוכן הבא:
  - users: {}
  - selectedRight: 0
  - selectedLeft: 0

class MessageService

- מכיל את התקשורת עם שרת ה- REST ל- API של ה- messages
- Singleton
- מכיל method ליצור הודעה חדשה
- מכיל method לקחת את כל ההודעות מהשרת

<CreateMessage />

- מכיל את הטופס ליצירת הודעה חדשה
- dumb component
- מקבל cb שאליו הוא ישלח את ההודעה החדשה שיצרנו

messages.actions.js

- קובץ המכיל actions שקשורים ל- messages
- מכיל שלושה actions:
  - setMessages - יאתחל את רשימת כל ה- messages
  - fetchMessages - action אסינכרוני שמביא מהשרת את כל המשתמשים

messages.reducer.js

- state התחלתי יכיל את התוכן הבא:
  - messages: {}

store

- נשתמש ב- combineReducers כדי ליצור reducer מאוחד משני ה- reducers שכבר יצרנו
- נתקין Redux dev tools
- נתקין redux-thunk
- נעטוף את ה- <App /> ב- <Provider store={store} >

<ChatList />

- יכיל את רשימת כל ההודעות שהמשתמש השני שלח
- dumb component
- יקבל את ההודעות וגם את המשתמש שמחפשים את ההודעות שלו

<SingleMessage />

- מציג הודעה בודדת
- מקבל את ההודעה ואת המשתמש שכתב אותה

<App />

- ה- App אצלנו מהווה את ה- component החכם היחיד והוא היחיד שמחובר ל- state ול-actions
- צריך לדאוג לאכלס את ה- selectedLeft, selectedRight לפי הרישום של המשתמש
- צריך ליזום הבאת ההודעות מהשרת כל פעם שהודעה חדשה נוצרת
- צריך לדאוג להעביר את המשתמש וההודעות ל- ChatList

סיכום

- ראינו איך מאכלסים את ה- state מתוך מידע שמגיע מהשרת
- סידרנו את ה- state כאובייקט במקום כמערך
- נתנו דגש ל - containers ול dumb components
- עבדנו עם async actions