1/17

Store:

* obiekt, który zarządza state w aplikacji
* tworzony za pomocą metody createStore() obiektu Redux
* metoda createStore() przyjmuje jako parametr tzw. reducer – funkcję przyjmującą jako argument state i zwracająca nowy state

Zadanie:

const reducer = (state = 5) => {

  return state;

}

// Redux methods are available from a Redux object

// For example: Redux.createStore()

// Define the store here:

const store = Redux.createStore(reducer)

2/17 **Get State from the Redux Store**

* obiekt store ma dostęp do różnych metod
* np. Można wyciągnąć obecny stan przechowywany w store za pomocą metody getState()

const store = Redux.createStore(

  (state = 5) => state

);

// Change code below this line

const currentState = store.getState()

3/17 **Define a Redux Action**

* aktualizacja stanu jest jednym z głównych zadań Reduxa
* aktualizacja ta odbywa się za pomocą akcji (actions)
* akcje to obiekty JS zawierające informacje o akcji jaka zaistniała
* store otrzymuje te obiekty akcji, a na ich podstawie aktualizuje stan
* czasem akcja może zawierać jakieś opcjonalne informacje (payload)
* koniecznie trzeba w akcji zamieścić właściwość o nazwie typ (type)
* o akcjach należy myśleć jak o posłańcach, którzy przenoszą informacje do store o zdarzeniach następujących w aplikacji, następnie store aktualizuje stan na podstawie tych informacji

// Define an action here:

const action = {

  type: 'LOGIN'

}

4/17 **Define an Action Creator**

* kolejny krokiem po stworzeniu akcji jest jej wysłanie do store, żeby mógł on zaktualizować stan
* żeby to osiągnąć, w Reduxie tworzy się action creators - są to zwykłe funkcje, które zwracają akcje

const action = {

  type: 'LOGIN'

}

// Define an action creator here:

const actionCreator=()=>{

  return action

}

5/17 **Dispatch an Action Event**

* metoda dispatch służy do przesyłania akcji do store
* wywołanie store.dispatch() i przekazanie wartości zwróconej przez action creator wysyła akcję z powrotem do store
* poniższe zapisy są równoznaczne, obie zwracają akcje o typie LOGIN:

store.dispatch(actionCreator());

store.dispatch({ type: 'LOGIN' });

const store = Redux.createStore(

  (state = {login: false}) => state

);

const loginAction = () => {

  return {

    type: 'LOGIN'

  }

};

// Dispatch the action here:

store.dispatch(loginAction())

6/17 **Handle an Action in the Store**

* po utworzeniu i wysłaniu (dispatch) akcji store musi wiedzieć w jaki sposób odpowiedzieć na akcję – to jest zadaniem funkcji reducer
* reducer jest odpowiedzialny za modyfikację stanu w odpowiedzi na przesłaną akcję
* reducer za argumenty przyjmuje stan oraz akcję (state, action) i zawsze zwraca nowy stan
* jest to jedyny cel reducera - nie pojawiają się żadne efekty uboczne (np. zapytania do API), nie modyfikuje on stanu lecz zwraca nowy