При взаємодії з сервером маємо справу з асинхронними запитами. Відмітимо, що при взаємодії з сервером весь процес складається з двох частин: запит на сервер та відповідь від сервера (одержання даних або помилка). Тому необхідно описати 3 дії (actions)

**FETCH\_DOGS\_REQUEST**,  
 **FETCH\_DOGS\_SUCCESS**,  
 **FETCH\_DOGS\_FAILURE**

|  |  |
| --- | --- |
|  |  |
| Створюємо констани стану | **const** FETCH\_STATUS = {  **IS\_PASSIVE**: **"IS\_PASSIVE"**,  **IS\_FETCHING**: **"IS\_FETCHING"**,  **FETCHED\_DATA**: **"FETCHED\_DATA"**,  **FETCHING\_ERROR**: **"FETCHING\_ERROR"**, };  **export default** FETCH\_STATUS |
| Створюємо конструктори дій | **import** {**createActions**} **from 'redux-actions' const** {fetchDogsRequest,fetchDogsSuccess,fetchDogsFailure}=**createActions**(  {  **FETCH\_DOGS\_REQUEST**:**undefined**,  **FETCH\_DOGS\_SUCCESS**:**undefined**,  **FETCH\_DOGS\_FAILURE**:**undefined** } ) **export** {fetchDogsRequest,fetchDogsSuccess, fetchDogsFailure} |
| Створюємо редюсери | **import** {fetchDogsRequest,fetchDogsSuccess, fetchDogsFailure} **from '../actions'**  **const** initialState = {  **fetchingState**: FETCH\_STATUS.**IS\_PASSIVE**,  **dogs**: [{**title**:**"first"**,**src**:**"--- шлях до першої собачки --- в "**}],  **error**: **null** };  **const** reducer = **handleActions**(  {  [fetchDogsRequest]: state => ({  ...state,  **fetchingState**: FETCH\_STATUS.**IS\_FETCHING** }),  [fetchDogsSuccess]: (state,action) => ({  ...state,  **fetchingState**: FETCH\_STATUS.**FETCHED\_DATA**,  **dogs**: [...state.**dogs**, action.**payload**]  }),  [fetchDogsFailure]: (state,action) => ({  ...state,  **fetchingState**: FETCH\_STATUS.**FETCHING\_ERROR**,  **error**: action.**payload**.**error** })   },  initialState );  **export default** reducer |
| Створюємо middleware | **import** {FETCH\_STATUS} **from "../constants" import** {fetchDogsRequest,fetchDogsSuccess, fetchDogsFailure} **from '../actions'**  **const** *dogFetchingMiddleware* = store => next => action => {  **const** result = next(action);   **if** (action.**type** === fetchDogsRequest().**type**) {  fetch(**'https://dog.ceo/api/breeds/image/random'**,{**method**:**"GET"**})  .then(response=>response.json())  .then(dog => {  store.dispatch(fetchDogsSuccess({**title**:dog.**message**, **src**:dog.**message**}))  })  .catch(error => {  store.dispatch(fetchDogsFailure({error}))  })  }  **return** result; } **export default** *dogFetchingMiddleware* |
| Створюємо сховище | **import** *dogFetchingMiddleware* **from '../middlewares' import** {createStore,*applyMiddleware*} **from "redux"**; **import** reducer **from "../reducers"**;  **let** initState={**dogs**:[]} **const** store = createStore(reducer, initState,  *applyMiddleware*(*dogFetchingMiddleware*)); //🡨 Підключаємо middleware  **export default** store; |
| Створюємо компоненти | **------------------- Відображення одного собаки--------------------------**  **import** React **from 'react'**;  **const** *Dog* = ({dog}) => {  **return** (  <**div**>  <**p**>{dog.**title**}</**p**>  <**img src=**{dog.**src**} **alt="Собака"**/>  </**div**>  ); };  **export default** *Dog*;  **------------------- Компонент відображення масиву собак -----------------**  **import** React, {PureComponent} **from 'react'**; **import** {**Provider**, **connect**} **from 'react-redux' import** {FETCH\_TYPES,FETCH\_STATUS} **from "../constants" import** *Dog* **from './Dog'  class** DogsViewer **extends** PureComponent {  render() {  **const** {fetchingState, error, dogs,fetchDogsRequest} = **this**.**props**;  **if** (fetchingState==FETCH\_STATUS.**IS\_FETCHING**)  **return** <**p**>Завантаження даних ...</**p**>;  **if** (error != **null**)  alert(**"Помилка при зчитуванні даних"**)  **return** <**div**>  {dogs.**length**? dogs.map((dog,i) => <**Dog key={i} dog=**{dog} />):  **"Немає собак"** }  <**button onClick=**{()=>fetchDogsRequest()}>Додати собаку</**button**>  </**div**>;  } } **export default** DogsViewer |
| Підключаємо компонент до сховища | **import** React **from 'react' import** {**Provider**, **connect**} **from 'react-redux'**  **import** DogsViewer **from './components/DogsViewer' import** {fetchDogsRequest,fetchDogsSuccess, fetchDogsFailure} **from './actions'**  **import** store **from './store'  const** *mapStateToProps* = state => ({  **error**: state.**error**,  **dogs**: state.**dogs**,  **fetchingState**: state.**fetchingState**, });  **const** mapDispatchToProps= {  fetchDogsRequest *//* fetchDogsRequest *:()=>dispatch(fetchDogsRequest())* };  **let** ConnectedDogsViewer = **connect**(  *mapStateToProps*,  mapDispatchToProps )(DogsViewer);  **export default** () => (  <**Provider store=**{store}>  <**ConnectedDogsViewer**/>  </**Provider**> ) |
| Використовуємо компонент | **import** React, { Component } **from 'react'**; **import** logo **from './logo.svg'**; **import './App.css'**; **import** *DogsViewer* **from './DogsViewerApp' class** App **extends** Component {  render() {  **return** (  <**div className="App"**>  <**DogsViewer**/>  </**div**>  );  } }  **export default** App; |