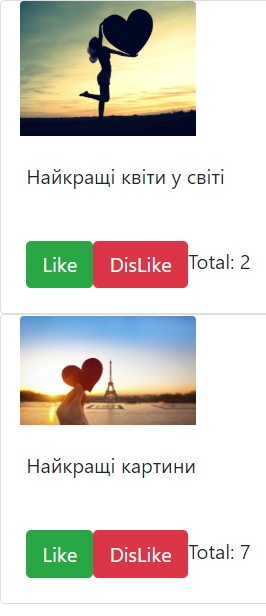
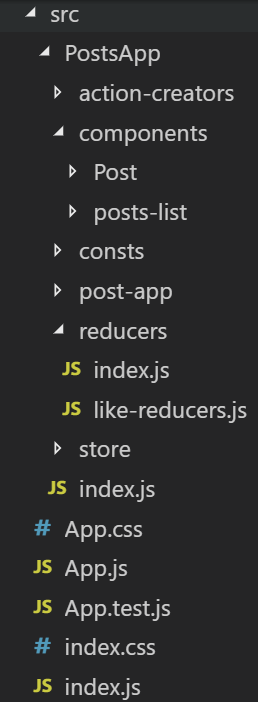
Приклад. Додаємо лайки та дислайки

|  |  |
| --- | --- |
| Загальна форма | Приклад |
| Створення типів дій  (для зручності типи подій можна представити у вигляді властивостей об’єкта-константи)  {  властивість\_дія : назва\_дії,  . . . . . . . .  } | const LikeTypes={  ADD\_LIKE:"ADD\_LIKE",  ADD\_DISLIKE:"ADD\_DISLIKE",  }  export default LikeTypes |
| Генератори дій  Дія – це об’єкт, у якому обов’язковим є поле типу type (вказує на тип події), а також можуть бути інші властивості, які несуть у собі необхідні дані (що необхідні для виконання даної дії).  Генератор дії – це функція, яка створює і повертає об’єкт дії.  {  type : тип події ,  payload: необхідні дані  }  Для спрощення використання подій прийнято створювати спеціальні функції-генератори. | import LikeTypes from '../consts'  const addLikeActionCreator =(postId)=>(  {  type:LikeTypes.ADD\_LIKE,  payload:{  postId  }  }  )  const addDisLikeActionCreator =(postId)=>(  {  type:LikeTypes.ADD\_DISLIKE,  payload:{  postId  }  }  )  export {addLikeActionCreator, addDisLikeActionCreator} |
| Reducers (перетворювачі сховища) -- це функції, які на вхід приймають поточний стан і дію, та повертає новий стан)  (previousState, action) =>  newState  Для кожної події можна створювати окрему функцію-reducer або ж використовувати оператор вибору switch (аналізуючи тип події) | import LikeTypes from '../consts'  const likeReducers = (prevState, action) => {  const { posts } = prevState  switch (action.type) {  case LikeTypes.ADD\_LIKE: {  console.log(action);  const { postId } = action.payload  console.log(postId);  const newOb={  posts: {  ...posts,  [postId]: {  ...posts[postId],  likesCount: posts[postId].likesCount + 1  }  }  }  console.log('new: ')  console.log(newOb)  return {  posts: {  ...posts,  [postId]: {  ...posts[postId],  likesCount: posts[postId].likesCount + 1  }  }  }  }  case LikeTypes.ADD\_DISLIKE: {  const {postId} = action.payload  // const {postId}=action.payload  return {  posts: {  ...posts,  [postId]: {  ...posts[postId],  likesCount: posts[postId].likesCount - 1  }  }  }  }  default:  return prevState;  }  }  export default likeReducers |
| Компоненти | import React from 'react';  const Post = ({id,imgSrc,text,likesCount,addLike, addDisLike}) => {  return (  <div className='card'>  <div className='col col-sm-2'>  <img src={imgSrc} class="card-img-top"/>  </div>  <div className="card-body">  {text}  </div>  <div className="card-body ">  <button class="btn btn-success" onClick={()=>addLike(id)}>  Like  </button>  <button class="btn btn-danger" onClick={()=>addDisLike(id)}>  DisLike  </button>  Total: {likesCount}  </div>  </div>  );  }    export default Post;  -------------------------------------------------------------------------------------------------------------  import React from 'react';  import Post from '../Post'  const PostsList = ({posts,addLike, addDisLike}) => {  return (  <>  {Object.keys(posts).map(id=>(  <Post key={id} {... posts[id]} addLike={addLike} addDisLike={addDisLike}/>  ))}  </>  );  }    export default PostsList;  ------------------------------------------------------------------------------------------------------------- |
|  | import React,{Component} from 'react';  import {addLikeActionCreator, addDisLikeActionCreator} from '../action-creators'  import PostsList from '../components/posts-list'  class PostApp extends Component {  addLike=(id)=>{  const {store} =this.props  store.dispatch( addLikeActionCreator(id) )  }  addDisLike=(id)=>{  const {store} =this.props  store.dispatch( addDisLikeActionCreator(id) )  }  render() {  const {store} =this.props  const {posts}= store.getState()  return (  <PostsList posts={posts}  addLike={this.addLike}  addDisLike={this.addDisLike}/>  );  }  }    export default PostApp  --------------------------------------------------------------- |
| Сховище (store) – місце, де зберігається стан додатку  Створення сховища  **const** store=createStore(  *reducer*,  initialStore);  **Store Methods**   * [getState()](https://redux.js.org/api/store#getState) -повертає поточний стан * [dispatch(action)](https://redux.js.org/api/store#dispatch) – ініціює виконання дії * [subscribe(listener)](https://redux.js.org/api/store#subscribe) – підписує функцію на події сховища * [replaceReducer(nextReducer)](https://redux.js.org/api/store#replaceReducer)- зміна перетворювача сховища (reduser) | import { createStore } from "redux";  import likesReducers from "../reducers";  let initialStore= {  posts:{  1:{  id:1,  imgSrc:'https://psy-practice.com/upload/iblock/143/love\_test.jpg',  text:'Найкращі квіти у світі',  likesCount:2  },  2:{  id:2,  imgSrc:'https://mirpozitiva.ru/uploads/posts/2017-04/medium/1492620921\_06.jpg',  text:'Найкращі картини',  likesCount:7  }  }  }  export default createStore(likesReducers,initialStore); |
|  | import PostApp from './post-app'  import store from './store'  import ReactDOM from 'react-dom'  import React from 'react';  //-------- Функція, яку будемо підписувати на події сховища ---------  const renderPostApp = () => {  ReactDOM.render(  <PostApp store={store}/>,  document.getElementById("root")  );  };  //-------- Підписуємо функцію на події сховища  store.subscribe(renderPostApp);  export default renderPostApp; |
| Використання додатку | import React from 'react';  import logo from './logo.svg';  import './App.css';  import renderPostApp from './PostsApp'  import store from './PostsApp/store'  function App() {  return (  <div className="App">  {renderPostApp()}  </div>  );  }  export default App; |