**Tạo ứng dụng ghi chú với ReactJS và Redux**

**Trong bài viết này mình sẽ hướng dẫn xây dựng một ứng dụng ghi chú đơn giản sử dụng ReactJS và Redux. Đây cũng là một project nhỏ giúp làm quen với Redux.**

Bạn cần chuẩn bị một vài kiến thức về Redux trước khi bắt tay vào xây dựng ứng dụng bằng cách tham khảo bài viết về [tích hợp Redux vào ReactJS](https://freetuts.net/tich-hop-redux-vao-reactjs-2636.html).

Ở đây ta sẽ xây dựng ứng dụng ghi chú đơn giản với chức năng bao gồm thêm, sửa, xóa ghi chú, có cấu trúc thư mục src như sau:

src/

....const/

index.js

.....actions/

index.js

....reducers/

noteReducer.js

index.js

----components/

AddNote.css

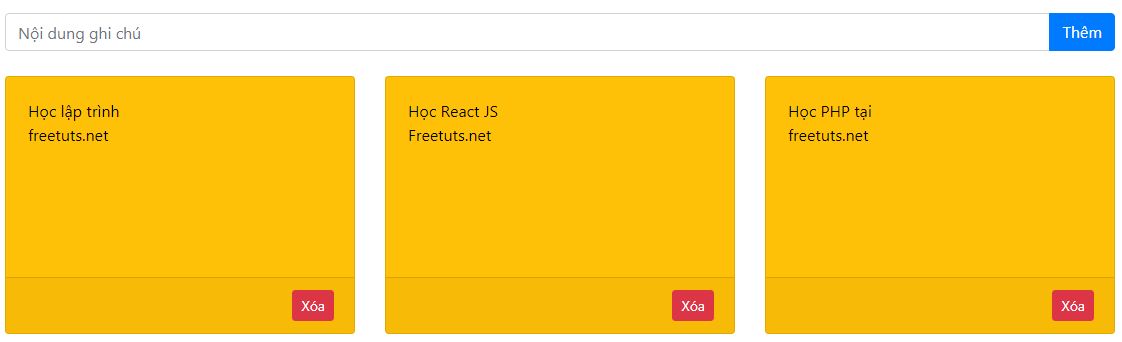
AddNote.js

ShowNote.js

....App.js

Ngoài ra mình có sử dụng thư viện Bootstrap 4 để xây dựng giao diện, hãy tham khảo thêm về Bootstrap 4 để có thể hiểu rõ hơn nhé.

*Bài viết này được đăng tại [free tuts .net]*

  
Giao diện của ứng dụng ghi chú

**Mục lục**

* [1. Cài đặt thư viện cần thiết](https://freetuts.net/tao-ung-dung-ghi-chu-voi-reactjs-va-redux-2654.html#goto-h2-0)
* [2. Triển khai Redux](https://freetuts.net/tao-ung-dung-ghi-chu-voi-reactjs-va-redux-2654.html#goto-h2-1)
* [3. Xây dựng giao diện ứng dụng ghi chú](https://freetuts.net/tao-ung-dung-ghi-chu-voi-reactjs-va-redux-2654.html#goto-h2-2)

**1. Cài đặt thư viện cần thiết**

Trước tiên cần phải cài đặt redux cho dự án của mình bằng npm với câu lệnh:

|  |  |
| --- | --- |
| 1 | npm install redux react-redux --save |

Để có thể xây dựng giao diện dễ dàng hơn ta sẽ sử dụng Bootstrap 4, tiến hành thêm thư viện bootstrap vào bên trong thẻ <head> của file public/index.html.

|  |  |
| --- | --- |
| 1 | <link rel="stylesheet" href="<https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css>"> |

Bước tiếp theo phải phân tích xem project cần những chức năng gì, để từ đó xây dựng cấu trúc redux một cách hợp lí hơn. Đối với project này bao gồm 3 chức năng chính đó là thêm, sửa, xóa ghi chú. Đồng nghĩa với đó là sẽ có 3 actions, 1 reducers và 3 const,..chúng ta sẽ triển khai ở bên dưới.

**2. Triển khai Redux**

Chúng ta sẽ triển khai mô hình Redux với 3 thư mục đó là *actions, reducers, const*,..Với từng thư mục sẽ có chức năng riêng.

Thư mục const sẽ chứa các hằng số hỗ trợ trong quá trình triển khai dự án. Ta sẽ khai báo các hằng số ở trong file const/index.js.

|  |  |
| --- | --- |
| 1  2  3  4 | // const/index.js  export const ADD\_NEW\_NOTE = "ADD\_NEW\_NOTE";  export const REMOVE\_NOTE = "REMOVE\_NOTE";  export const EDIT\_NOTE = "EDIT\_NOTE"; |

Thư mục actions sẽ bao gồm các hàm chứa hành động để thực hiện *dispatch* tới *reducers*. Ở đây chúng ta sẽ triển khai các hàm ở file actions/index.js.

import { ADD\_NEW\_NOTE, REMOVE\_NOTE, EDIT\_NOTE } from "../const/index";

//action thêm note

export const actAddNote = (content) => {

return {

type: ADD\_NEW\_NOTE,

content,

};

};

//Xóa note

export const actRemoveNote = (id) => {

return {

type: REMOVE\_NOTE,

id,

};

};

//Sửa note

export const actEditNote = (id, content) => {

return {

type: EDIT\_NOTE,

id,

content,

};

};

Có 3 hàm được export ở file actions/index.js đó là actAddNote, actRemoveNote, actEditNote dùng để tạo action cho chức năng thêm, sửa, xóa note.

Thư mục reducers sẽ chứa các hàm chứa các *reducers* trong Redux, có 2 files đó là

* noteReducers.js : reducers hỗ trợ thực thi hành động với ghi chú như thêm, sửa, xóa. Ta có thể tạo nhiều *reducers* như này, và gộp nó vào bằng phương thức combineReducers trong redux.
* index.js : trong trường hợp có nhiều *reducers* thì gộp tất cả *reducers* ở đây.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | //reducesr/noteReducers.js    //Import các const  import { ADD\_NEW\_NOTE, REMOVE\_NOTE, EDIT\_NOTE } from "../const/index";    const noteReducers = (state = [], action) => {    switch (action.type) {      case ADD\_NEW\_NOTE:        const generateID = new Date().getTime();        state = [...state, { id: generateID, content: action.content }];        return state;        case EDIT\_NOTE:        const indexOfEditNote = state.findIndex((note) => note.id === action.id);        if (indexOfEditNote !== -1)          state[indexOfEditNote].content = action.content;          return state;        case REMOVE\_NOTE:        const idRemove = action.id;        state = state.filter(note => {          if (note.id === idRemove)            return false          return true        })        return state;      default:        return state;    }  };    export default noteReducers |
| 1  2  3  4  5  6  7 | //reducers/index.js  import {combineReducers} from 'redux'  import noteReducers from './noteReducer'    export default combineReducers({      note: noteReducers  }) |

Để các component khác có thể tương tác với **Store** ta cần bọc root component bởi Provider. Ở file src/index.js sẽ chỉnh sửa lại như sau.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | import React from "react";  import ReactDOM from "react-dom";  import "./index.css";  import App from "./App";  import \* as serviceWorker from "./serviceWorker";    import { Provider } from "react-redux";  import { createStore } from "redux";    //Gọi reducers  import reducers from "./reducers/index";      //Tạo store từ reducers  const store = createStore(reducers);    ReactDOM.render(    <React.StrictMode>      <Provider store={store}>        <App />      </Provider>    </React.StrictMode>,    document.getElementById("root")  );    serviceWorker.unregister(); |

Vậy là các bước cài đặt redux xong, tiếp theo sẽ là đi xây dựng các component.

**3. Xây dựng giao diện ứng dụng ghi chú**

Bây giờ sẽ đi xây dựng giao diện và kết nối với store redux trong mỗi component. Ví dụ này sẽ có 2 components đó thêm và hiển thị ghi chú (nằm trong thư mục src/components).

Ở đây 2 component AddNote và ShowNote sẽ được render ở trong file src/App.js

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | //file: src/App.js  import React from "react";  import "./App.css";  //Kết nối vơi redux  import { connect } from "react-redux";  import ShowNote from './components/ShowNote'  import AddNote from './components/AddNote'  function App(props) {    return (      <div className="row" style={{ marginTop: "3%" }}>        <AddNote />        {props.note.map((note, index) => {            // Render ra lần lượt các ghi chú.            return <ShowNote noteData = {note} key={note.id}/>        })}      </div>    );  }    // Lấy state từ store bằng mapStateToProps  // Lúc này state nhận được sẽ gán vào props  const mapStateToProps = (state, ownProps) => {    // Gán state nhận về từ store    // thành props có tên note (props.note)    return {      note: state.note,    };  };    export default connect(mapStateToProps, null)(App); |

Component AddNote sẽ có chức năng thêm ghi chú mới, file AddNote.js sẽ có nội dung như sau:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63 | //file AddNote.js    import React, { useState, useRef } from "react";  import { connect } from "react-redux";  //Import actions vào đây  import { actAddNote } from "../actions/index";    function AddNote(props) {    const [content, setContent] = useState();      //Refs: Giúp chúng ta tương tác với DOM thật      const noteInput = useRef(null)      //Khi click vào nút thêm sẽ gọi hàm này,    const handleAdd = () => {        // Dispatch action.      // Props này được tạo bởi hàm      // mapDispatchToProps bên dưới      props.addNote(content)        //Gán giá trị cho input thành rỗng      noteInput.current.value = ''        //Cập nhật lại state content      setContent('')    };      return (      <div className="col-md-12" style={{ marginBottom: 15 }}>        <div className="input-group mb-8">          <input            type="text"            className="form-control"            placeholder="Nội dung ghi chú"            value =  {content}            onChange={(e) => {              setContent(e.target.value)            }}            ref={noteInput}          />          <div className="input-group-append">            <button className="btn btn-primary" onClick={handleAdd}>              Thêm            </button>          </div>        </div>      </div>    );  }    //Chuyển dispatch thành props.  //Ở đây nếu mình muốn dispatch action actAddNote  //thì chỉ cần gọi props.addNote(content)  const mapDispatchToProps = (dispatch) => {    return {      addNote: (content) => {        dispatch(actAddNote(content));      },    };  };  export default connect(null, mapDispatchToProps)(AddNote); |

Component ShowNote sẽ nhận vào một props là dataNote chứ các thuộc tính của từng ghi chú. Component này sẽ có chức năng sửa và xóa các ghi chú.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60 | //file: src/components/ShowNote.js  import React, { useState } from "react";  import "./ShowNote.css";  import { connect } from "react-redux";  import { actEditNote, actRemoveNote } from "../actions/index";    function ShowNote(props) {    const [noteContent, setNoteContent] = useState(props.noteData.content)      //Lấy ID của ghi chú    const noteID = props.noteData.id        // Được gọi mỗi khi thay đổi giá trị    // của ghi chú.    const handleChange = (e) => {      setNoteContent(e.target.value)      props.editNote(noteID, e.target.value)    }      //Được gọi khi click vào    //xóa ghi chú.    const handleRemoveNote = () => {      props.removeNote(noteID)    }    return (      <div className="col-md-4" style={{marginTop: 10}}>        <div className="card bg-warning">          <div className="card-body" style={{ height: 200 }}>            <textarea value= {noteContent} onChange = {handleChange}></textarea>          </div>          <div className="card-footer">            <button className="btn btn-danger btn-sm float-right" onClick={handleRemoveNote}>Xóa</button>          </div>        </div>      </div>    );  }    // chuyển dispatch thành props  const mapDispatchToProps = (dispatch) => {    return {      editNote: (id, content) => {        dispatch(actEditNote(id, content));      },      removeNote: id => {        dispatch(actRemoveNote(id));      }    };  };    // chuyển state từ store thành props  // của component  const mapStateToProps = (state, ownProps) => {    return {      note: state.note,    };  };    export default connect(mapStateToProps, mapDispatchToProps)(ShowNote); |

Để giao diện của component ShowNote đẹp hơn 1 chút thì ta sẽ thêm một vài đoạn css ở file ShowNote.css

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | /\* src/components/ShowNote.css \*/  textarea {      background-color: rgba(0, 0, 0, 0);      border-width: 0;      overflow: hidden;      resize: none;  }  textarea:focus{      outline: none;  } |

Đây cũng là buớc cuối cùng để xây dựng ứng dụng ghi chú, để khởi chạy dự án ta chỉ cần mở terminal và gõ dòng lệnh:

|  |  |
| --- | --- |
| 1 | npm start |

Trên đây là ví dụ về khởi tạo ứng dụng ghi chú sử dụng ReactJS và Redux, mong rằng ví dụ này sẽ giúp bạn có thể nắm chắc hơn về cách triển khai redux trong một dự án thực tế.