

**TÀI LIỆU HƯỚNG DẪN SỬ DỤNG SOURCE CHOCAYXANH-REACTAPP.**

# **I/ Tìm hiểu công nghệ trong source ReactApp.**

## **0/ Tài liệu tham khảo:**

Link hướng dẫn học react native cơ bản tiếng việt: <https://www.youtube.com/watch?v=JzfleA-ZpMg&list=PLWBrqglnjNl03UsDDN5ogQP_SHmK-qB4q&index=1>

Tài liệu react native chính thức từ Facebook: <https://facebook.github.io/react-native/docs/getting-started>

React native hello world: <https://facebook.github.io/react-native/docs/tutorial>

## **1/ React native:**

### a/ React native là gì?

+ React Native là một framework do công ty công nghệ nổi tiếng Facebook phát triển cho phép build ứng dụng di động trên cả 2 nền tảng ios và android mà chỉ cần 1 source duy nhất thay vì phải sử dụng nhiều loại ngôn ngữ native cho từng nền tảng di động khác nhau.

### b/ Tại sao lại sử dụng React Native?

+ React native tuy là ứng dụng Hybrid nhưng vẫn đảm bảo hiệu năng ơ mức ổn định và Sử dụng react native giúp giảm chi phí và thời gian khi mà không cần phải viết nhiều loại ngôn ngữ native cho từng nền tảng di động.

### c/ Các thành phần của React Native?

+ Component: là khái niệm cơ bản trong các ứng dụng React. Các ứng dụng react-native quản lý ứng dụng theo các component. Một component đại diện cho 1 thành phần của ứng dụng. Trong 1 component cha có thể chứa các component con.

Lợi ích đáng kể nhất của component là tính tái sử dụng và mở rộng giúp dễ dàng quản lý source.

+ State: Là thuộc tính private của một component. State chỉ có giá trị bên trong component và không thể truy cập state từ bên ngoài component.

Link tham khảo: <https://www.youtube.com/watch?v=v6P80ouFJqg&list=PLWBrqglnjNl03UsDDN5ogQP_SHmK-qB4q&index=4>

+ Props: là thuộc tính public của class component trong react. Props có thể truyền từ component này sang component khác. Component con khi thao tác với dữ liệu props truyền từ component cha chỉ nên đọc dữ liệu không nên làm theo đổi giá trị props.

Link tham khảo: <https://www.youtube.com/watch?v=PNNlxM0mhOU&list=PLWBrqglnjNl03UsDDN5ogQP_SHmK-qB4q&index=3>

+ Event handler: một số cách bind event trong ứng dụng react

1/ bind event trong hàm dựng constructor

2/ bind event bằng cú pháp arrow function trong es6

Link tham khảo: <https://www.freecodecamp.org/news/the-best-way-to-bind-event-handlers-in-react-282db2cf1530/>

### d/ Link tài liệu tham khảo về các thành phần của react-native: <https://viblo.asia/p/component-state-va-props-trong-react-native-Eb85oM34Z2G>

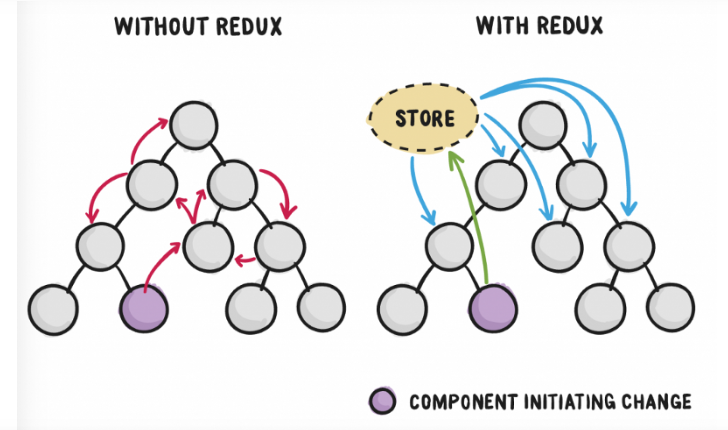
## **2/ Redux:**

### a/ Redux là gì?

Redux là một library hỗ trợ quản lý và chia sẻ state trong các ứng dụng js. Redux thường đc sử dụng chung với React nên cũng đc gọi là react-redux.

+ Tại sao nên sử dụng Redux?

Khi ứng dụng react chứa quá nhiều các component thì việc quản lý và chia sẻ giá trị state trở nên rất khó khăn. Redux sinh ra nhằm giải quyết bài toán quản lý state một cách khoa học và hiệu quả hơn.



### b/ Các thành phần và workflow của Redux?

1. Action:

Là một pure object gồm 2 thuộc tính là “type”: loại action được thực hiện và “payload”: giá trị tham số truyền đi. Action được gửi bằng cách sử dụng store.dispatch() method.

1. Reducer:

+ Định nghĩa cách state theo đổi theo action được dispatch.

+ Reducer nhận 2 tham số vào: 1 state cũ và action được dispatch sau đó trả ra một state mới, ko làm thay đổi state cũ.

1. Store:

Nơi lưu giữ các state toàn cục của ứng dụng trong redux. mỗi component đều có thể access bất kỳ state nào mà chúng muốn từ chính store này. Store có 3 phương thức chính.

+ getState(): Giúp lấy ra state hiện tại.

+ dispatch(action): Thực hiện gọi 1 action

+ subscrible(listener): lắng nghe xem có thay đổi gì ko rồi cập nhật ra View.

1. Workflow:

Luồng hoạt động của Redux:

Dispatch(action) -> reducer xử lý cập nhập state theo action -> state mới cập nhập trong store -> nếu state đổi render lại View.

1. Ví dụ: các bạn down source example này về. <https://github.com/ochutgio/counter-app>

### c/ Tài liệu tham khảo:

<https://viblo.asia/p/redux-la-gi-va-tai-sao-can-dung-redux-RnB5pGJJlPG>

<https://insights.innovatube.com/redux-th%E1%BA%ADt-l%C3%A0-%C4%91%C6%A1n-gi%E1%BA%A3n-ph%E1%BA%A7n-1-76a3fa2c31ab>

## **3/ Typescript:**

### a/ Typescript là gì:

TypeScript là một dự án mã nguồn mở được phát triển bởi Microsoft, nó có thể được coi là một phiên bản nâng cao của Javascript bởi việc bổ sung tùy chọn kiểu tĩnh và lớp hướng đối tượng mà điều này không có ở Javascript. TypeScript có thể sử dụng để phát triển các ứng dụng chạy ở client-side (Angular2, Reactjs, React Native, VueJS..) và server-side (NodeJS).

### a/ Tại sao nên sử dụng Typescript ?

1/ Dễ phát triển dự án lớn: Với việc sử dụng các kỹ thuật mới nhất và lập trình hướng đối tượng nên TypeScript giúp chúng ta phát triển các dự án lớn một cách dễ dàng.

2/ TypeScript là Javscript: Bản chất của TypeScript là biên dịch tạo ra các đoạn mã javascript nên bạn có thê chạy bất kì ở đâu miễn ở đó có hỗ trợ biên dịch Javascript. Ngoài ra bạn có thể sử dụng trộn lẫn cú pháp của Javascript vào bên trong TypeScript, điều này giúp các lập trình viên tiếp cận TypeScript dễ dàng hơn.

### c/ Tài liệu tham khảo:

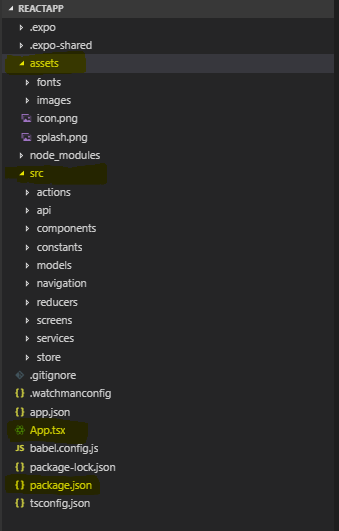
<https://viblo.asia/p/tim-hieu-typescript-va-kien-thuc-co-ban-PmeRQpnyGoB>

# **II/ Hướng dẫn sử dụng source ReactApp.**

## **0/ Source:**

Các bạn clone source về theo đường dẫn: https://ochutgio@bitbucket.org/ochutgio/chocayxanh-reactapp.git

## **1/ Cấu trúc source:**



+ thư mục assets chứa các tài nguyên tĩnh như hình ảnh, font chữ, ....

+ thư mục src chứa tất cả các code chính.

+ file app.tsx file khởi chạy ban đầu của ứng dụng

+ file package.json chứa các dependency của ứng dụng

## **2/ Ý nghĩa các thư mục.**

+ src/actions: chứa định nghĩa các action **(redux)**

+ src/api: chứa các hàm gọi api.

+ src/components/: chứa các component.

+ src/constants: Chứa các khai báo hằng số, biến tĩnh.

+ src/models: Chứa các định nghĩa class model

+ src/navigation: Chứa định nghĩa switch navigation.

+ src/screens: Chứa các màn hình chính của ứng dụng.

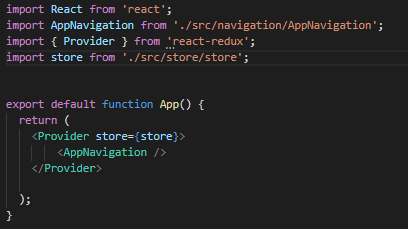
+ src/reducers: Chứa các reducers **(redux)**

+ src/services: chứa các hàm xử lý nghiệp vụ, tính toán.

+ src/store: chứa định nghĩa store **(redux)**

## **3/ Redux workflow:**

+ Thực ra react & redux là 2 thư viện được sử dụng hoàn toàn độc lập với nhau. Chức năng chính của redux xoay quanh store ( nơi lưu trữ state của toàn bộ ứng dụng). Ta không thể chỉnh sửa giá trị state một cách trực tiếp mà phải thông qua reducers & cách duy nhất là gửi một action (dispatch actions).



Hình 1: App.tsx:

Component Provider của react-redux cung cấp store của redux cho các component con của nó. Vì vậy ở đây ta cần đặt component AppNavigation bên trong component Provider để toàn bộ component của ứng dụng có thể truy cập được store.



Hình 2: src/components/places.tsx

Hàm connect() sẽ giúp component kết nối tới store của redux. Hàm mapStateToProps() nhằm mục đích lấy đúng giá trị state cần lấy từ store của redux và gán giá trị vào props của component. Hàm mapDispatchToProps sẽ truyền hàm dispatch từ store của redux sang props của component tức là ta có gọi hàm dispatch trong component, ví dụ: this.props.dispatch(addPlace(place).



Hình 3: src/components/places.tsx

Khi hàm This.props.add() được gọi reducer nhận được action và cập nhập giá trị state.places.places trong store của redux. Lúc này this.props.places sẽ nhận được giá trị state.places.places từ store của redux.

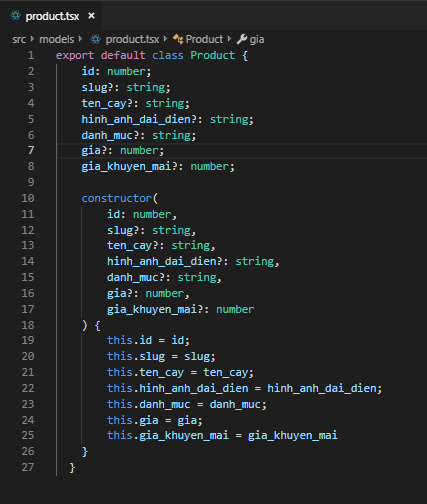
Luồng redux sẽ như sau: action => reducer => store => view(component)

## **4/ Các bước thực hiện 1 chức năng:**

Dưới đây sẽ là các bước thực hiện 1 chức năng demo là lấy danh sách sản phẩm từ server về. Các bước được thực hiện như sau:

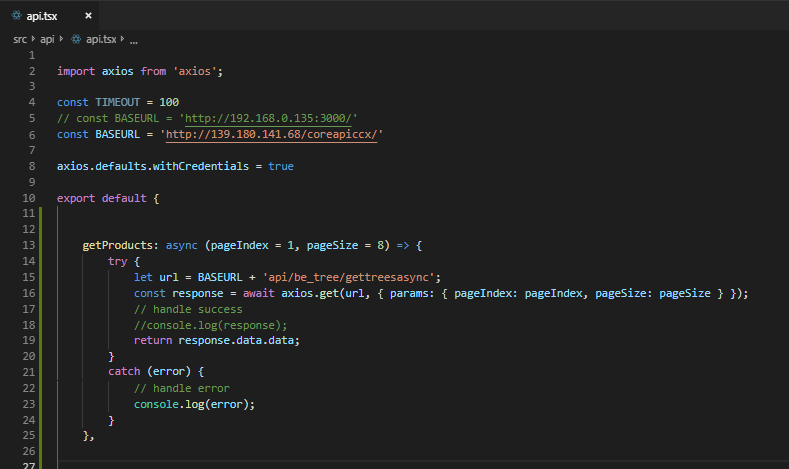
1. Viết api và định nghĩa model class Product.

+ Định nghĩa model class Product.tsx



Hình 4: src/models/product.tsx

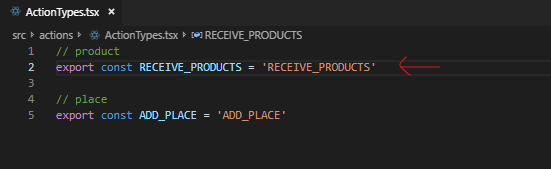
+ Viết api lấy danh sách sản phẩm.



Hình 5: src/api/api.tsx

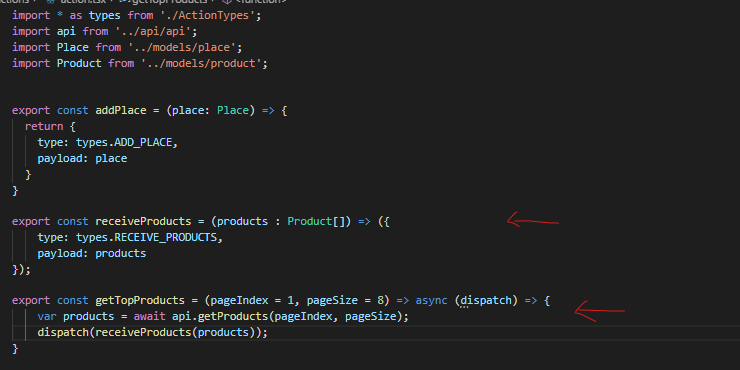
+ Axios là một thư viện hỗ trợ giao tiếp api theo chuẩn restful api dựa trên giao thức http.

1. Định nghĩa actionType trong file src/actions/actionTypes.tsx



Hình 6: src/actions/actiontypes.tsx

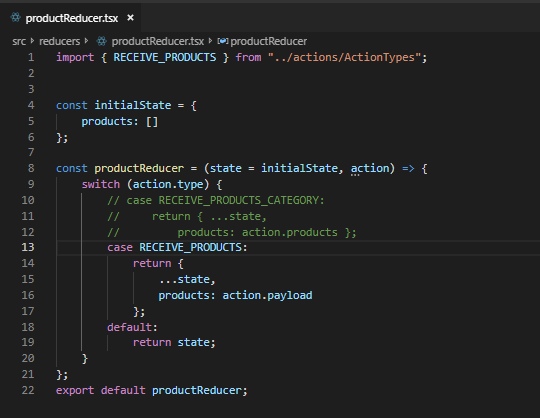
1. Định nghĩa action trong file src/actions/action.tsx



Hình 7: src/actions/action.tsx

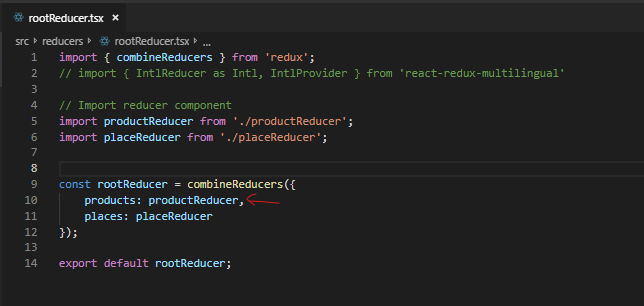
1. Viết reducer xử lý cho action.

+ tạo một productReducer theo đường dẫn src/reducers/productReducer.tsx



Hình 8: src/reducers/productReducer.tsx

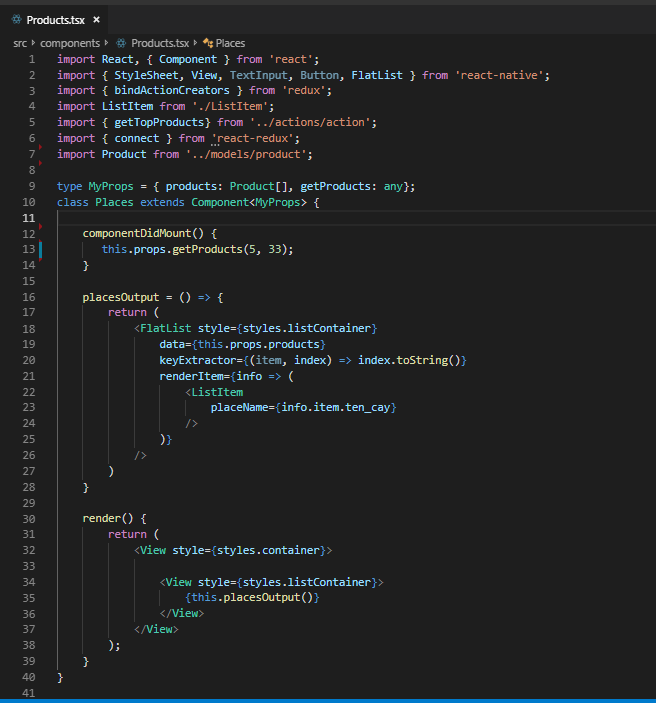
+ combine productReducer vào rootReducer



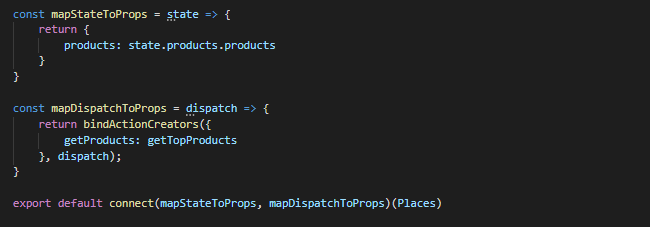
Hình 9: src/reducers/rootReducer.tsx

1. Xây dựng component và viết xử lý component

+ Tạo mới component src/components/products.tsx







Hình : src/components/products