ReactJS

# Giới thiệu

## Giới thiệu

React Js là một thư viện viết bằng javascript, dùng để xây dựng giao diện người dùng (UI). React được sử dụng rộng rãi và có hệ sinh thái đa dạng phong phú. UI tất nhiên là quan trọng, nhưng không phải là tất cả. Để phát triển ứng dụng hoàn chỉnh, một mình React Js là không làm được tất cả, bạn sẽ cần thêm:

* Server side language: để xử lý logic và lưu trữ dữ liệu trên server.
* HTML/CSS nếu bạn làm ứng dụng web.
* Flux/Redux?: là một kiến trúc giúp bạn tổ chức code rõ ràng và sạch sẽ.
* Objective C: nếu bạn sử dụng React để xây dựng app cho iOS

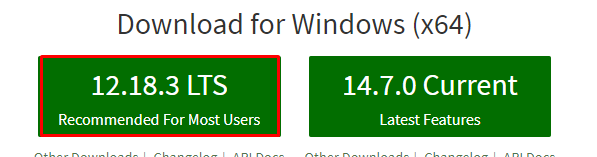
## Cài đặt môi trường

### Các công cụ

* **React react detector**: Giúp ta xác định được trang web đang sử dụng React



* **NodeJS**: Cài đặt NodeJS



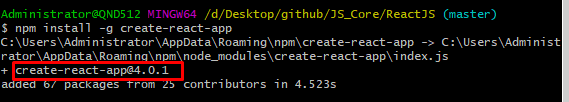
* **Yarn**: Là lựa chọn thay thế cho npm. (Có thể sử dụng thêm)
* **IDE:** Công cụ viết code (Subline Text, Visual Studio Code)

### Khởi tạo Project

* Document React: <https://reactjs.org/docs/getting-started.html>

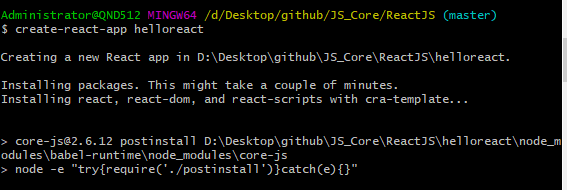
**Bước 1**: Cài đặt create react app

Cài đặt bằng câu lệnh: npm install -g create-react-app



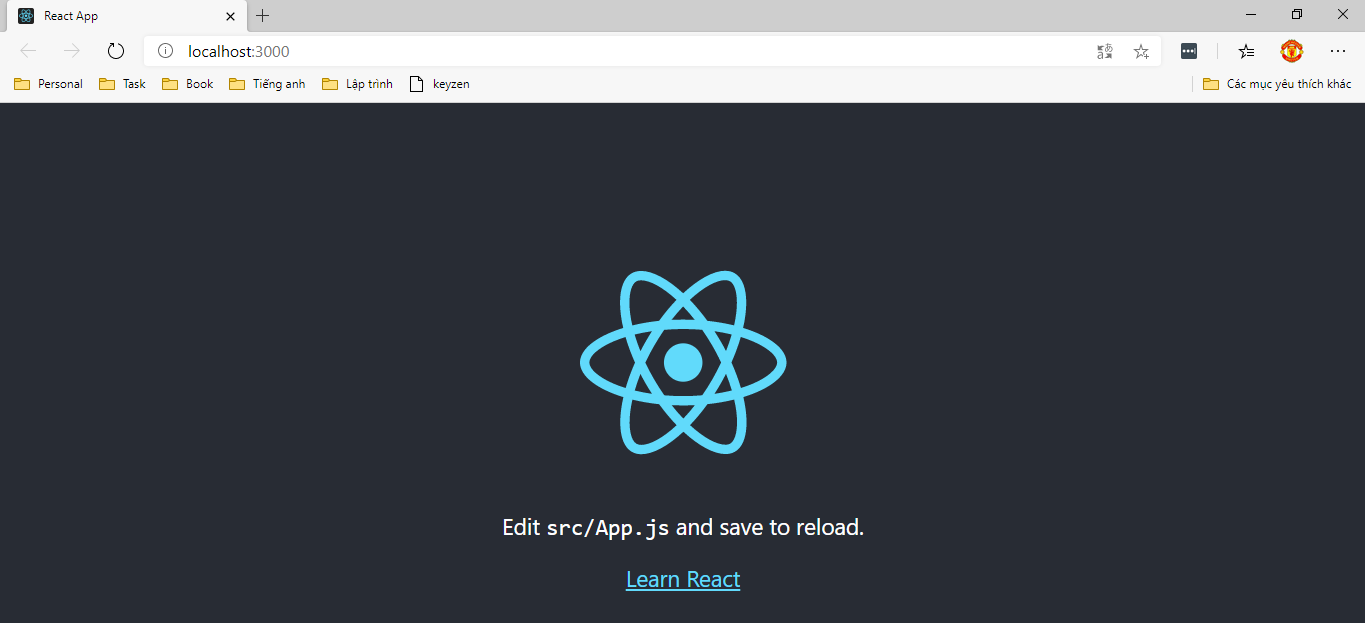
**Bước 2:** Tạo mới một project

* Tạo một thư mục sau đó vào thư mục đó bật cmd lên mà nhập lệnh tạo project
* Câu lệnh tạo: create-react-app ten\_project



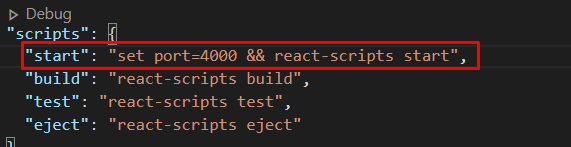
**Bước 3:** Chạy ứng dụng

* Vào trong project vừa tạo: cd helloreact
* Khởi chạy ứng dụng : npm start



**Bước 4:** Đổi port React

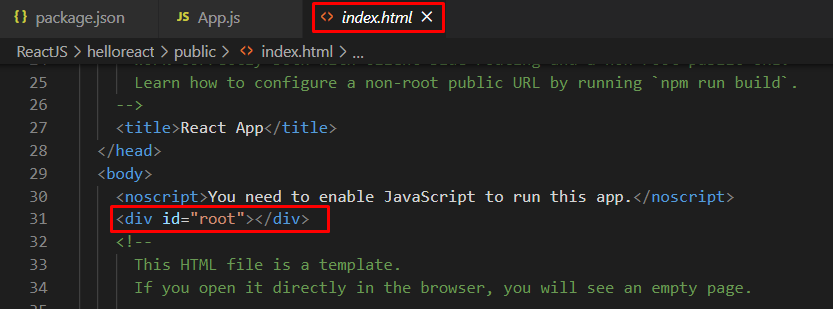
* Đổi port tại file package.json trong project
* Câu lệnh set port: “start” : ”set port=port\_muon\_thay\_doi && react-scripts start”

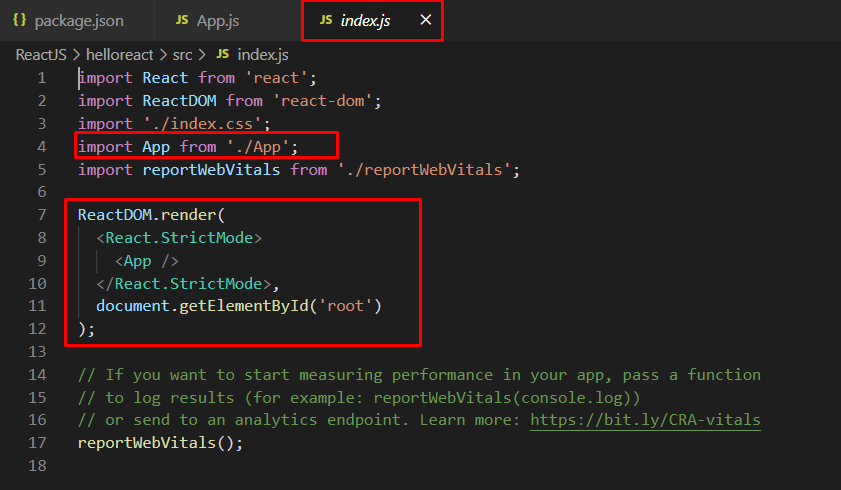


## Cấu trúc thư mục

* **Package.json**: Nơi khai báo thông tin của project
* **Node\_modules:** Nơi chứa các thư viện của ReactJS
* **Public/index.html:** Là file chạy đầu tiên. Trong index.html có:  Đây là nơi react được đổ vào.
* **Src:** Là nơi làm việc của lập trình viên. Ta có thể thay đổi và mở rộng code ở thư mục này.

## Đường đi của các file

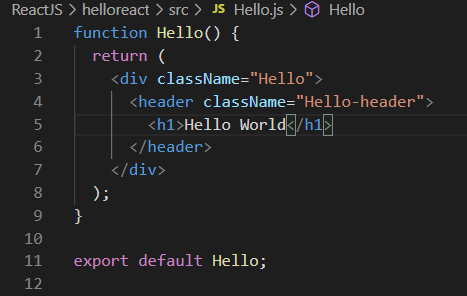




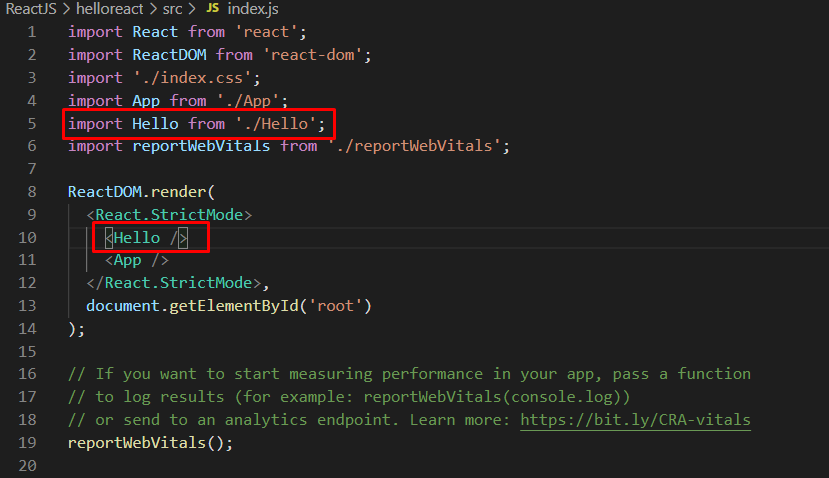
* Trong file index.js có phần render App vào component có Id là root trong index.html bằng câu lệnh document.getElementById(‘root)
* Và file App được import từ ./App và chính là file App.js trong thư mục src

## Hello World

**Bước 1**: Tạo Hello.js trong thư mục src tương tự như App.js



**Bước 2**: Import component Hello vừa mới tạo vào file index.js



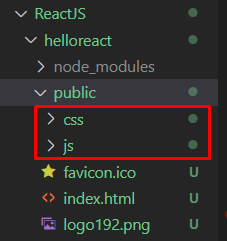
* Trong React.StrictMode sẽ là thứ tự các component mà chúng ta muốn sắp xếp

# Component

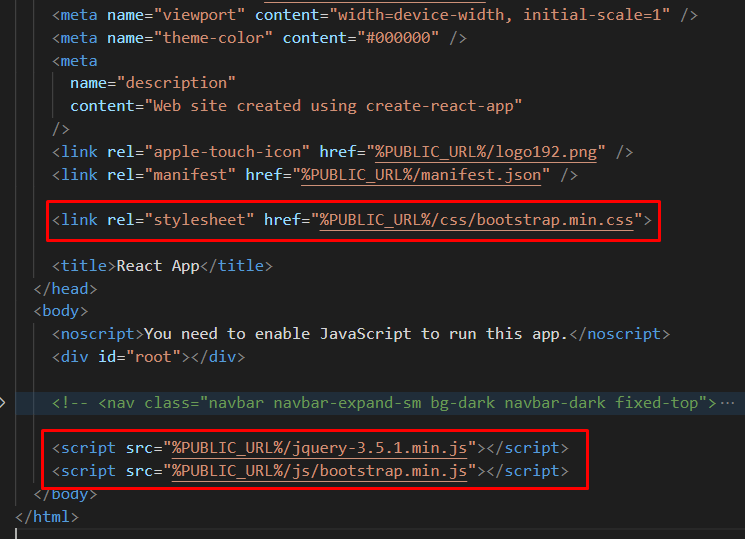
## Nhúng boostrap và các thư viện bên ngoài vào React

**Bước 1**: Download boostrap và Jquery

**Bước 2**: Giải nén và copy các thư mục css, js, và jquery vào thư mục public trong project

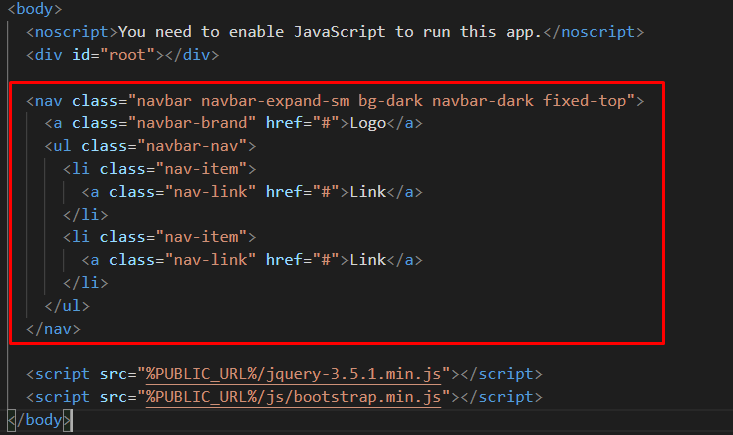


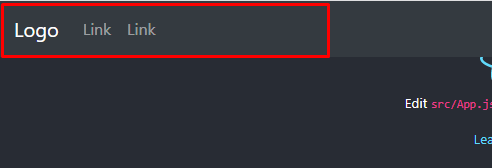
**Bước 3**: Import vào file index.html trong thư mục public



* Với %PUBLIC\_URL% là đường dẫn tới thư mục public trong project

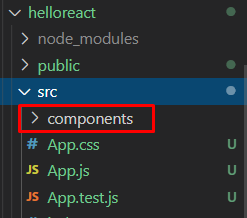
**Bước 4**: Kiểm tra xem project đã nhận boostrap chưa



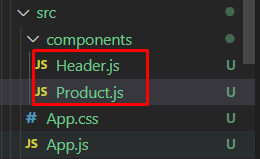


## Tạo một components

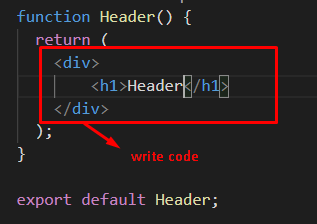
**Bước 1**: Tạo folder chứa các components



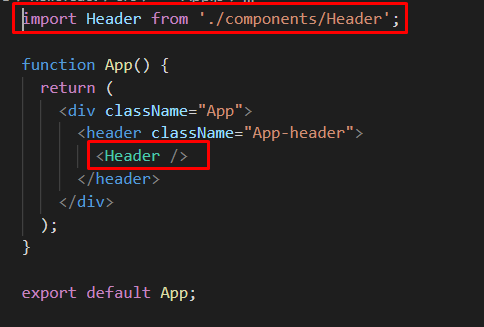
**Bước 2**: Tạo component trong folder vừa tạo



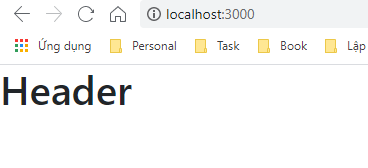
Với Header.js



Lồng Header.js vào App.js: Trong App.js



KQ:

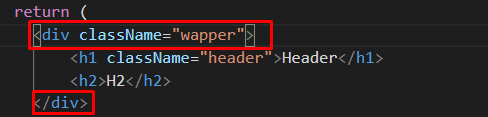


Chú ý:

* Trong component thì class sẽ được viết thành className

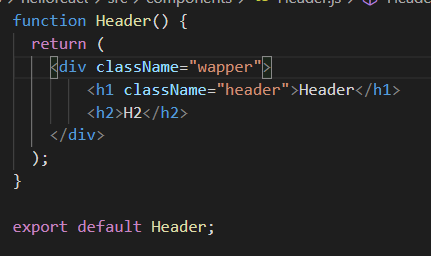


* Trong một component thì luôn luôn phải có một thẻ ngoài cùng bao bọc toàn bộ các thẻ con khác

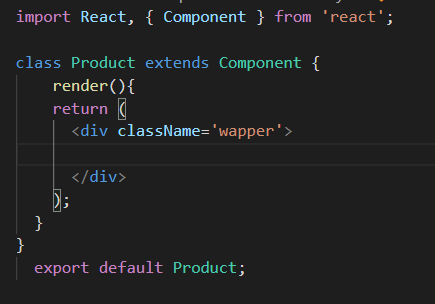


## Class component và Function component

* Function component



* Class component (ES6)

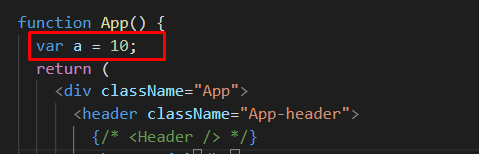


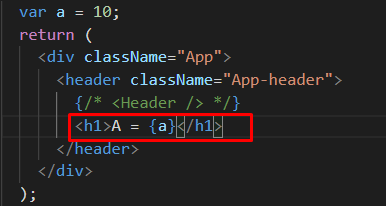
|  |  |
| --- | --- |
| Class component | Function component |
|  |  |
|  |  |
|  |  |

## JSX

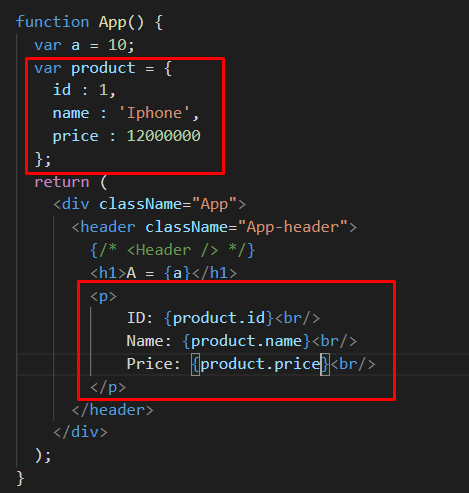
* JSX (viết tắt của cụm từ JavaScript XML) là một loại cú pháp mở rộng dành cho ngôn ngữ JavaScript viết theo kiểu XML. JSX cung cấp cú pháp ngọt (syntactic sugar) để thay cho câu lệnh React.createElement() trong React.
* Chuyển đổi từ HTLM sang JSX: <https://magic.reactjs.net/htmltojsx.htm>

### Biến và hiển thị biến trong JSX

* Khai báo biến: 
* Hiển thị biến ra ngoài màn hình:



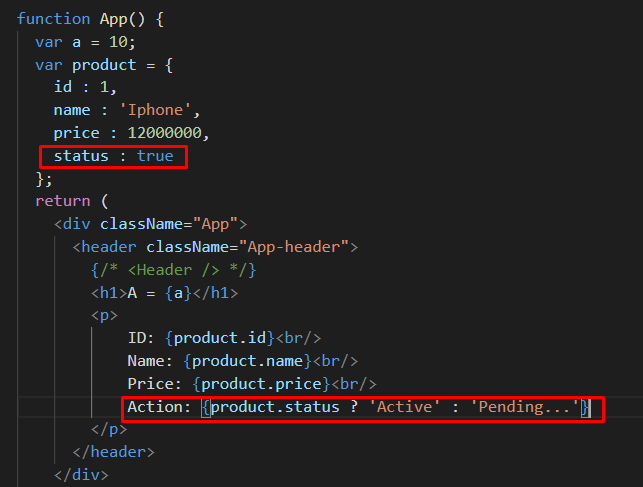
* Với biến là một project



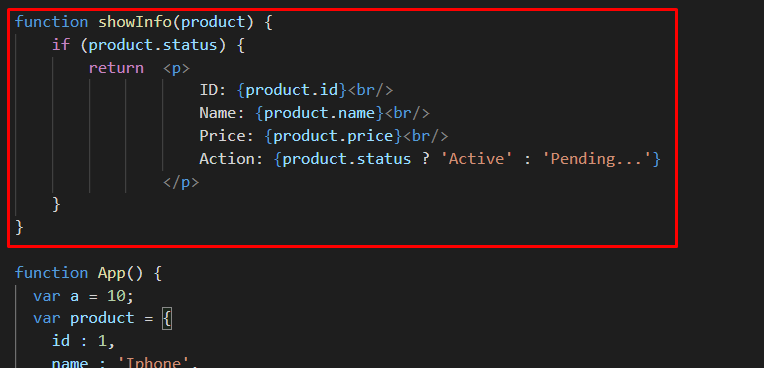
### Điều kiện

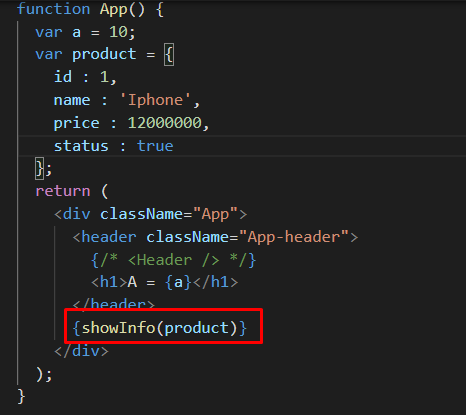
#### Toán tử 3 ngôi

* Cú pháp: {Cau\_lenh ? ’Dk\_dung’ : ‘Dk\_sai’}



#### Gọi thông qua thuộc tính



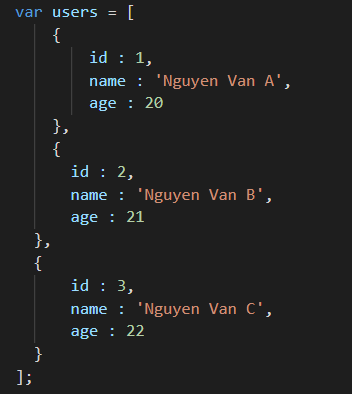


### Vòng lặp

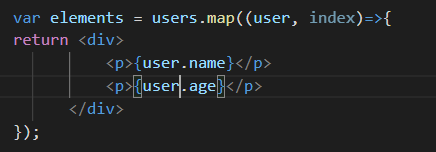
* Dùng hàm map() để sử dụng vòng lặp

VD:

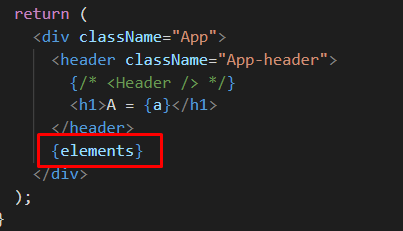
1. Ta có một mảng sau:



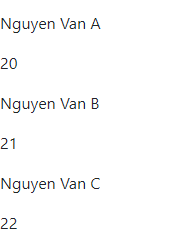
1. Khai báo vòng lặp với hàm map()



1. Gọi vòng lặp ra ngoài màn hình



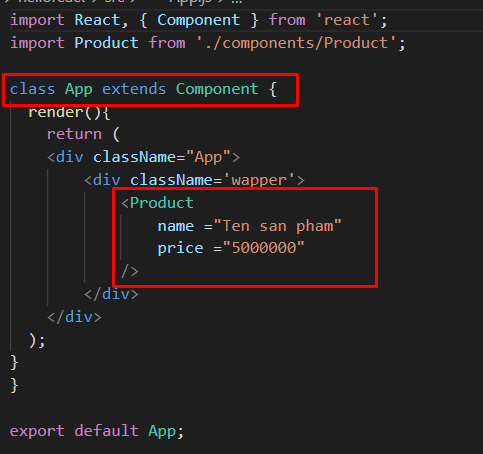
KQ:



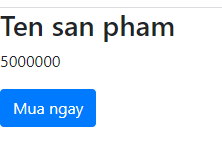
## Props

* Chỉ dùng được khi React JS ở dạng ES6 tức là các file phải ở dạng class
* Là thuộc tính của một components
* Truyền dữ liệu từ cha -> con theo dạng key=”value” (Không đặt key=”children”)
* Vlaue nhận vào có kiểu dữ liệu là kiểu chuỗi. Bỏ trong dấu { } để truyền đúng kiểu dữ liệu
* Nhận dữ liệu thông qua từ khóa: this.props.key
* Nhận nội dung bên trong thẻ: this.props.children

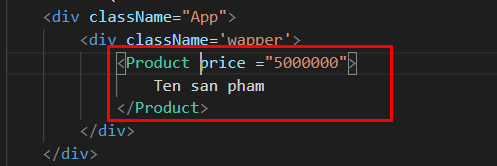
VD:

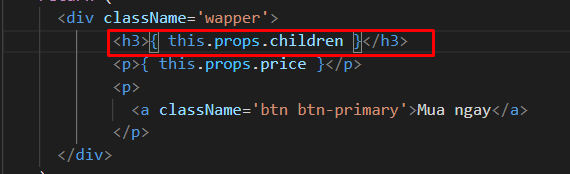


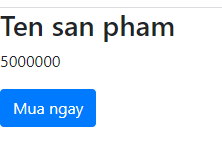




VD2:



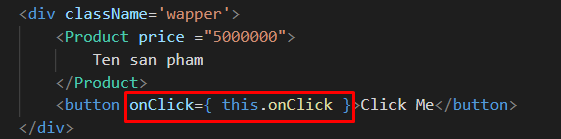


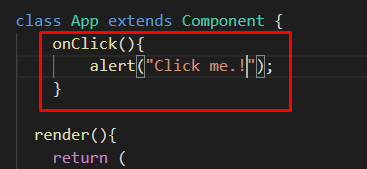


### Bắt sự kiện

* Sử dụng ReactJS được viết dưới dạng ES6

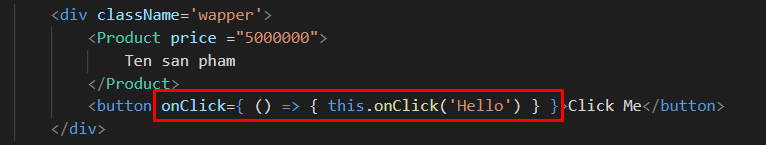
**Cách 1**: Bắt sự kiện không truyền tham số

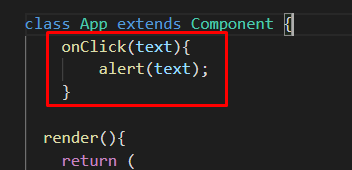




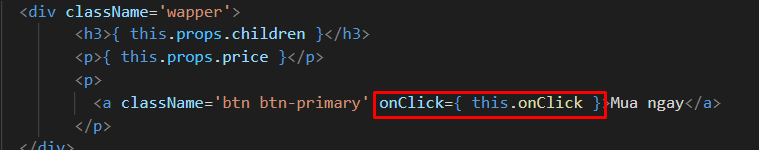
**Cách 2**: Sự kiện có tham số

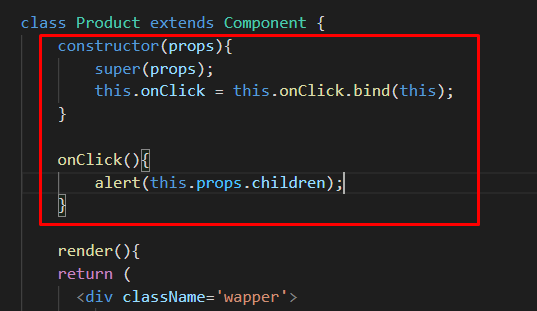
* Sử dụng () => {su\_Kien}



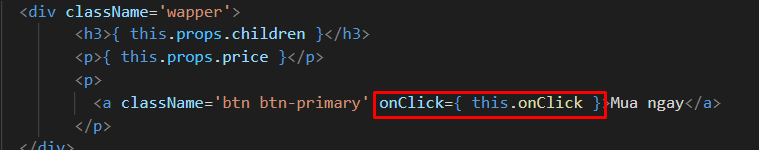


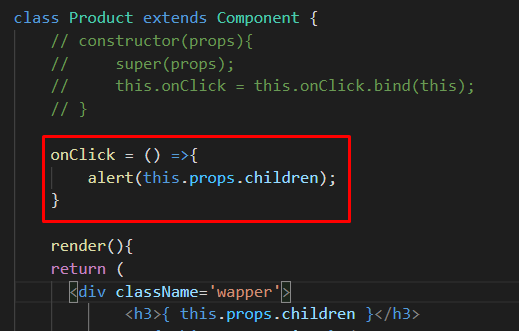
**Cách 3**: Sử dụng Props





**Cách 4**: Sử dụng props không khai báo contructor



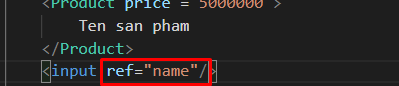


* Có thể tham khảo một số Events khác tại : <https://reactjs.org/docs/events.html#gatsby-focus-wrapper>

## Refs

* Lấy giá trị thông qua ref
* Cú pháp lấy dữ liệu: this.refs.key.value
* Các bước thực hiện:

1. Đặt tên key cho input



1. Bắt sự kiện cho button

