**Xây dựng Hook trong React JS (React Custom Hook)**

**Bài viết này sẽ hướng dẫn cách để tự tay đi xây dựng một hooks - custom hooks trong React JS.**

React Hooks là một tính năng mới trong React 16.8. Cho phép sử dụng state và các tính năng khác trong React mà không cần viết một class component. Ngoài sử dụng các hooks mặc định như *useState*, *useEffect*,...chúng ta còn có thể tự tạo một hooks tùy theo chức năng của nó.

**Mục lục**

* [1. Custom Hooks là gì?](https://freetuts.net/xay-dung-hook-trong-react-js-2725.html#goto-h2-0)
* [2. Khi nào dùng Custom Hooks](https://freetuts.net/xay-dung-hook-trong-react-js-2725.html#goto-h2-1)
* [3. Tự xây dựng một custom hook](https://freetuts.net/xay-dung-hook-trong-react-js-2725.html#goto-h2-2)

## 1. Custom Hooks là gì?



**Custom Hooks**là những hooks mà do lập trình viên tự định nghĩa với mục đích thực hiện một chức năng nào đó, nó thường được sử dụng để chia sẻ logic giữa các components.

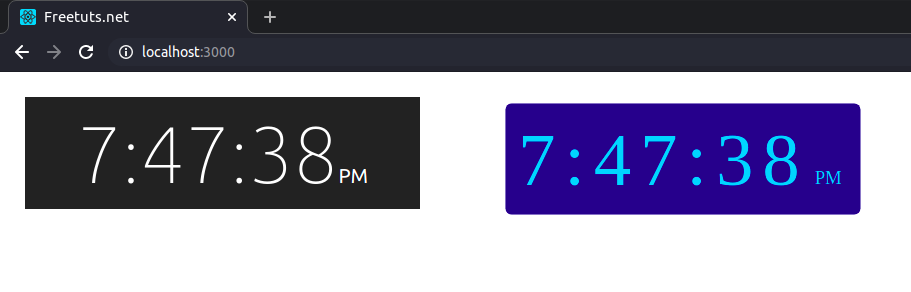
React cũng định nghĩa cho chúng ta các hooks như useState, useEffect, useContext,... cho phép chúng ta làm việc dễ dàng hơn. Để tự định nghĩa một hooks cho riêng mình, chúng ta chỉ cần xây dựng 1 hàm nhận và trả về các giá trị.

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

Khi đặt tên một custom hooks phải có từ khóa use ở đầu, ví dụ như: useClick(), useClock(), useQuery(),...

## 2. Khi nào dùng Custom Hooks

**Custom Hooks**rất hay được sử dụng trong quá trình triển khai ứng dụng. Trong trường hợp chúng ta muốn tách biệt các phần xử lý logic riêng ra khỏi UI, hay chia sẻ logic giữa các component. Giả sử trong trường hợp muốn xây dựng 2 components đều có 1 chức năng là hiển thị ngày giờ hiện tại nhưng khác nhau ở giao diện như bên dưới.



Thông thường, chúng ta sẽ viết tất cả các logic xử lý giờ ở trong component như này.

|  |  |
| --- | --- |
| 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 | import React, { useState } from "react";  import "./Clock1.css";  function Clock1() {    const [time, setTime] = useState("");    const [ampm, setampm] = useState("");    const updateTime = function () {      let dateInfo = new Date();      let hour = 0;      /\* Lấy giá trị của giờ \*/      if (dateInfo.getHours() === 0) {        hour = 12;      } else if (dateInfo.getHours() > 12) {        hour = dateInfo.getHours() - 12;      } else {        hour = dateInfo.getHours();      }      /\* Lấy giá trị của phút \*/      let minutes =        dateInfo.getMinutes() < 10          ? parseInt("0" + dateInfo.getMinutes())          : dateInfo.getMinutes();        /\* Lấy gía trị của giây \*/      let seconds =        dateInfo.getSeconds() < 10          ? "0" + dateInfo.getSeconds()          : dateInfo.getSeconds();        /\* Định dạng ngày \*/      let ampm = dateInfo.getHours() >= 12 ? "PM" : "AM";        /\* Cập nhật state \*/      setampm(ampm)      setTime(`${hour}:${minutes}:${seconds}`);    };    setInterval(function () {      updateTime();    }, 1000);    return (      <div className="time">        <span className="hms">{time}</span>        <span className="ampm">{ampm}</span>      </div>    );  }    export default Clock1; |

Rồi lại lặp lại logic này ở **Clock2**, có thể nhận thấy cách viết này không tối ưu một chút nào. Giả sử, dự án có 10 cái đồng hồ thì cần phải lặp lại logic này ở mỗi components. Điều này là không cần thiết và tốn thời gian. Giải pháp ở đây là sử dụng custom hooks.

## 3. Tự xây dựng một custom hook

Vậy làm sao để tự mình xây dựng một custom hooks, rất đơn giản chỉ cần tách phần xử lý logic ra một function.

Chúng ta sẽ có một custom hooks thay cho cách viết dài dòng ở ví dụ trên. Trong thư mục src tạo một thư mục có tên hooks, thư mục này sẽ chứa tất cả các custom hooks. Đây là một hooks có tên useClock() có nhiệm vụ trả về thời gian hiên tại.

|  |  |
| --- | --- |
| 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 | // src/hooks/useClock.js  import { useState } from "react";      export default function useClock() {      const [time, setTime] = useState("");      const [ampm, setampm] = useState("");        // Function cập nhật thời gian.      const updateTime = function () {        let dateInfo = new Date();        let hour = 0;        /\* Lấy giá trị của giờ \*/        if (dateInfo.getHours() === 0) {          hour = 12;        } else if (dateInfo.getHours() > 12) {          hour = dateInfo.getHours() - 12;        } else {          hour = dateInfo.getHours();        }        /\* Lấy giá trị của phút \*/        let minutes =          dateInfo.getMinutes() < 10            ? parseInt("0" + dateInfo.getMinutes())            : dateInfo.getMinutes();          /\* Lấy gía trị của giây \*/        let seconds =          dateInfo.getSeconds() < 10            ? "0" + dateInfo.getSeconds()            : dateInfo.getSeconds();          /\* Định dạng ngày \*/        let ampm = dateInfo.getHours() >= 12 ? "PM" : "AM";          /\* Cập nhật state \*/        setampm(ampm)        setTime(`${hour}:${minutes}:${seconds}`);      };        setInterval(function () {        updateTime();      }, 1000);        return [time, ampm]  } |

Khi muốn sử dụng hooks này chỉ cần import nó vào component và gọi như các hooks thông thường. React dự vào tên để xem đâu là một hooks bởi vậy nên đặt tên đúng định dạng là use + nameHooks.

// src/components/Clock2.js

import React from 'react'

import "./Clock2.css";

//Import Hooks

import useClock from '../hooks/useClock'

function Clock2() {

//Gọi custom hook để sử dụng

const [time, ampm] = useClock()

return (

<div id="MyClockDisplay" className="clock">

{time}

<span>{ampm}</span>

</div>

)

}

export default Clock2

Vậy là chúng ta có thể sử dụng logic trong nhiều component khác nhau mà không cần lặp lại các đọan logic phức tạp, chỉ cần import và gọi là có thể sử dụng. Với cộng đồng sử dụng rộng lớn thì ngoài các hooks có sẵn trong React, còn có các custom hooks do cộng đồng đóng góp. Có thể nói ReactJS là một thư viện khá mạnh và hữu ích cho việc lập trình front-end.

Trên đây chúng ta đã cùng nhau xây dựng một Hook trong React. Đây là kiến thức rất cơ bản về nó nhưng cũng hết sức quan trọng trong quá trình làm việc với ReactJS sau này. Mong rằng bài viết sẽ giúp ích cho bạn !