**Kỹ thuật Lifting State Up trong ReactJS**

Trong bài viết này chúng ta sẽ cùng nhau đi tìm hiểu về Lifting State Up trong ReactJS, trong quá trình làm việc với React, các component phải re-render và thay đổi rất nhiều lần. Trong một vài trường hợp, bạn muốn khi 1 component con được thay đổi, đồng thời bạn muốn component cha cũng sẽ bắt được hành động đó. Theo tài liệu chính thức của ReactJS thì Lifting State Up được định nghĩa là rằng :

several components need to reflect the same changing data. We recommend lifting the shared state up to their closest common ancestor

chúng ta có thể hiểu rằng

một vài component phải cùng nhận được sự thay đổi. Chúng tôi khuyên bạn nên chia sẻ các state cho cha, mẹ, ông bà dòng họ của chúng :v

Đọc đến đây chắc hẳn bạn cũng sẽ hiểu một phần, kỹ thuật lifting state up là cách mà có thể chia sẻ dữ liệu cho các component khác. Có nghĩa là khi bạn thay đổi dữ liêu của 1 component con thì bạn sẽ gửi dữ liệu cho component cha biết.

**Mục lục**

* [1. Lifting State Up trong React](https://freetuts.net/ky-thuat-lifting-state-up-trong-reactjs-2426.html#goto-h2-0)
* [2. Ví dụ cụ thể](https://freetuts.net/ky-thuat-lifting-state-up-trong-reactjs-2426.html#goto-h2-1)

## 1. Lifting State Up trong React

Trong phần này mình sẽ chỉ ra các bước thực hiện **Lifting State Up**, các bước thức hiện rất đơn giản và chúng ta sẽ có ví dụ cụ thẻ ở phần tiếp theo.

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

Trước tiên, chúng ta sẽ truyền cho component con một props có giá trị là một function. Function này sẽ được gọi khi component con trả về dữ liệu:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | function ComponentCha(props) {    //Hàm này sẽ được gọi khi nhận được dữ liệu    const receiveData = function (data) {      console.log('Data nhận được', data )    }    //Gọi compoennt con và truyền vào một props    //có giá trị là một hàm    return (      <ComponentCon onReceiveData = {receiveData}/>    )  } |

Tiếp theo, ở componentCon chúng ta sẽ tiến hành gửi lại cho componentCha bằng cách truyền vào props onReceiveData giá trị cần gửi về.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | function ComponentCon (props) {    return(      <div>          <button onClick={() => {                // Chúng ta sẽ gọi props có tên là              // receiveData đã được truyền từ compoentCha.              // Và truyền vào đó giá trị cần gửi                props.onReceiveData('data gửi đi')            }}>Gửi lại cho componentCha</button>      </div>    )  } |

Vậy là chúng ta đã gửi thành công dữ liệu từ componentCon sang componenetCha. Lúc này componentCha có thể sử dụng dữ liệu đó. Đây là các bước cơ bản để sử dụng Lifting State Up. Chúng ta sẽ đi vào ví dụ cụ thể bên dưới.

## 2. Ví dụ cụ thể

Ở đây chúng ta sẽ đi vào ví dụ cụ thể về cách sử dụng kỹ thuật Lifting State Up. Mình sẽ đi xây dựng một ứng dụng cho phép chuyển đổi giữa USD và VNĐ. Chúng ta sẽ làm việc với file App.js

Trước tiên, chúng ta sẽ đi xây dựng component cha có tên Caculator, và hàm handleChange chịu trách nhận data từ component con.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | export default class App extends Component {    constructor(props) {      super(props);      this.state = {        usd: 0,        vnd: 0,      };    }    handleChange = (data) => {      this.setState(data);    };      render() {      return (        <div style={{margin: "3%"}}>          <USDtoVND onHandleChange={this.handleChange} value={this.state.usd} />          <VNDtoUSD onHandleChange={this.handleChange} value={this.state.vnd} />          <hr />          <code>freetuts.net</code>        </div>      );    }  } |

Mình sẽ truyền cho 2 component con có tên USDtoVND và VNDtoUSD các props như:

* **onHandleChange:** giúp cho component con có thể gửi dữ liệu về component cha.
* **value:**giá trị của input.

Tiếp theo, chúng ta sẽ khởi tạo các component con có tên USDtoVND và VNDtoUSD:

|  |  |
| --- | --- |
| 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 | const USDtoVND = function (props) {    const convert = function (usd) {      return usd \* 23632;    };    return (      <div>        <span>USD </span>        <input          onChange={(e) => {            const usd = e.target.value;            const vnd = convert(usd);            props.onHandleChange({              usd,              vnd,            });          }}          value={props.value}        />      </div>    );  };  const VNDtoUSD = function (props) {    const convert = function (vnd) {      return vnd / 23632;    };    return (      <div>        <span>VND </span>        <input          onChange={(e) => {            const vnd = e.target.value;            const usd = convert(vnd);            props.onHandleChange({              usd,              vnd,            });          }}          value={props.value}        />      </div>    );  }; |

các component này có nhiệm vụ chuyển từ đổi tiền và trả về cho component cha bằng prop onHandleChange().

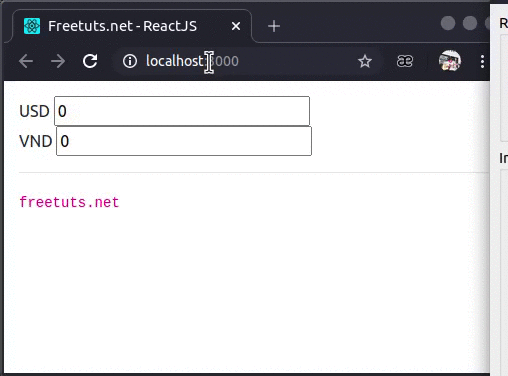
Cuối cùng, chúng ta sẽ có file App.js đầy đủ như sau:

|  |  |
| --- | --- |
| 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  64  65  66  67  68 | import React, { Component } from "react";    const USDtoVND = function (props) {    const convert = function (usd) {      return usd \* 23632;    };    return (      <div>        <span>USD </span>        <input          onChange={(e) => {            const usd = e.target.value;            const vnd = convert(usd);            props.onHandleChange({              usd,              vnd,            });          }}          value={props.value}        />      </div>    );  };  const VNDtoUSD = function (props) {    const convert = function (vnd) {      return vnd / 23632;    };    return (      <div>        <span>VND </span>        <input          onChange={(e) => {            const vnd = e.target.value;            const usd = convert(vnd);            props.onHandleChange({              usd,              vnd,            });          }}          value={props.value}        />      </div>    );  };    export default class App extends Component {    constructor(props) {      super(props);      this.state = {        usd: 0,        vnd: 0,      };    }    handleChange = (data) => {      this.setState(data);    };      render() {      return (        <div style={{margin: "3%"}}>          <USDtoVND onHandleChange={this.handleChange} value={this.state.usd} />          <VNDtoUSD onHandleChange={this.handleChange} value={this.state.vnd} />          <hr />          <code>freetuts.net</code>        </div>      );    }  } |

Khởi chạy dự án bằng cách mở terminal và gõ dòng lệnh:

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

Truy cập đường dẫn **http://localhost:3000** và chúng ta sẽ thấy kết quả:



Kỹ thuật Lifting State Up chỉ có thể sử dụng khi bạn truyền từ component cha sang con và ngược lại. Tức là bạn chỉ có thể truyền dữ liệu từ 1 component quan hệ cha con. Trong trường hợp, bạn muốn chia sẻ dữ liệu state sang component khác, chúng ta sẽ có khái niệm về Redux, cho phép chúng ta tạo ra local state, giúp việc chia sẻ dữ liệu giữa các component được dễ dàng hơn.

Trên đây chúng ta đã cùng nhau đi tìm hiểu về kỹ thuật Lifting State Up trong ReactJS 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.