**Tìm hiểu về Component API trong ReactJS**

Trong bài viết này chúng ta sẽ cùng nhau đi **tìm hiểu về các API bên trong các React Component**, đây là những API được sử dụng khá nhiều trong quá trình làm việc với các component bao gồm *setState(), forceUpdate(), ReactDOM.findDOMNode(),...*

**Mục lục**

* [Set State API](https://freetuts.net/component-api-trong-reactjs-2379.html#goto-h2-0)
* [Force Update](https://freetuts.net/component-api-trong-reactjs-2379.html#goto-h2-1)
* [Find Dom Node](https://freetuts.net/component-api-trong-reactjs-2379.html#goto-h2-2)
* [Bind function](https://freetuts.net/component-api-trong-reactjs-2379.html#goto-h2-3)

**Set State API**

Hàm setSate() này cho phép chúng ta cập nhật giá trị của state, tham số truyền vào của API này sẽ là giá trị của state bạn muốn update:

|  |  |
| --- | --- |
| 1  2  3 | this.setSate({     website: 'freetuts.net'  }) |

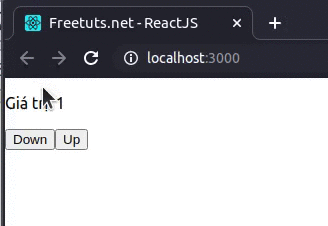
hoặc là một function callback bao gồm các tham số lần lượt là state trước đó, và props có trong component.

|  |  |
| --- | --- |
| 1  2  3 | this.setState((prevState, props) => {       return newState;  }) |

Trong ví dụ bên dưới mình sẽ tiến hành update state bằng 2 cách trên.

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

|  |  |
| --- | --- |
| 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 | import React from "react";    class App extends React.Component {    constructor(props) {      super(props);      this.state = {        index: 1      };    }    countDown() {      this.setState({        index: this.state.index - 1      });    }    countUp(){      this.setState((prevState, props) => {        return {          index: prevState.index + 1        }      });    }    render() {      return (        <div>          <p>Giá trị: {this.state.index}</p>          <button onClick={() => this.countDown()}>Down</button>          <button onClick={() => this.countUp()}>Up</button>        </div>      );    }  }    export default App; |



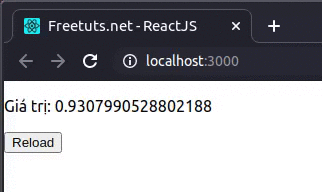
**Force Update**

Component chỉ thực hiện re-render khi các state thay đổi, trong trường hợp cần phải re-render mà không cần bất cứ sự thay đổi nào của state thì chúng ta chỉ cần gọi hàm forceUpdate()

this.forceUpdate()

Bên dưới mình có ví dụ về hàm forceUpdate(), với mỗi lần component được re-render sẽ random ra một số khác nhau:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | import React from "react";    class App extends React.Component {    constructor(props) {      super(props);      this.state = {        index: 1      };    }    render() {      return (        <div>          <p>Giá trị: {Math.random()}</p>          <button onClick={() => this.forceUpdate()}>Reload</button>        </div>      );    }  }    export default App; |



**Find Dom Node**

Hàm này cho phép chúng ta làm việc với DOM trong React, nó không được khuyến khích bởi sẽ làm ảnh hưởng đến các Virtual DOM gây ra lỗi không mong muốn. Bạn cần phải gọi đối tượng ReactDOM trong thư viện react-dom bằng cách:

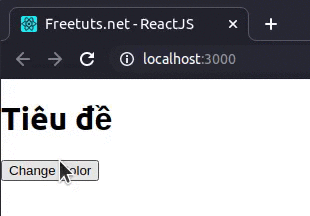
|  |  |
| --- | --- |
| 1 | import ReactDOM from 'react-dom'; |

Để sử dụng hàm findDOMNode() chúng ta có cú pháp như sau:

|  |  |
| --- | --- |
| 1 | ReactDOM.findDOMNode(component) |

Giả sử ví dụ bên dưới mình muốn sửa đổi màu chữ của thẻ h1 có id là title thành màu đỏ.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | import React from "react";  import ReactDOM from "react-dom";    class App extends React.Component {    changeColor() {      var title = document.getElementById("title");      ReactDOM.findDOMNode(title).style.color = "red";    }    render() {      return (        <div>          <h1 id="title">Tiêu đề</h1>          <button onClick={() => this.changeColor()}>Change Color</button>        </div>      );    }  }    export default App; |



**Bind function**

Đây là phần mà rất nhiều bạn gặp lỗi liên quan đến cú pháp ES6 khi thực hiện gọi hàm trong class component, giả sử mình muốn khi người dùng click vào nút **Change** sẽ gọi hàm changeTitle().

Khi bạn gọi hàm bằng cách này sẽ gây ra lỗi không tồn tại biến this trong hàm changeTitle().

|  |  |
| --- | --- |
| 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 | class App extends React.Component {    constructor() {      super();      this.state = {        title: "Freetuts.net"      };    }    changeTitle() {      //Sẽ không nhận được giá trị của this.      //Từ đó không thể set state được.      this.setState({        title: "Freetuts.net New"      });    }      render() {      //Khi bạn gọi hàm như này sẽ gây ra lỗi.      //Vì không chỉ định giá trị của biến this cho hàm changeTitle.      return (        <div>          <h1 id="title">{this.state.title}</h1>          <button onClick={this.changeColor}>Change Color</button>        </div>      );    }  } |

Chúng ta cần phải chỉ định biến this cho hàm vừa gọi bằng cách dùng cú pháp bind trong ES6:

|  |  |
| --- | --- |
| 1 | <button onClick={this.changeColor.bind(this)}>Change Color</button> |

hoặc viết như sau :

|  |  |
| --- | --- |
| 1 | <button onClick={() => this.changeColor()}>Change Color</button> |

Trên đây chúng ta đã cùng nhau đi tìm hiểu về Props 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.