REACTJS 03

FDP 5.0

Nội dung

- Form
- CSS & SASS

FORM

- Cũng giống như trong HTML, React sử dụng **form** để cho phép người dùng tương tác với trang web.
- Cấu trúc 1 form cũng tương tự với HTML

```
class MyForm extends React.Component {
  render() {
    return (
      <form>
        <h1>Hello</h1>
        <p>Enter your name:</p>
        <input
          type="text"
        />
      </form>
    );
  }
}
ReactDOM.render(<MyForm />, document.getElementById('root'));
```

FORM

- Sử dụng state để xử lý dữ liệu trong các control
- Đặt event **onChange** trong mỗi control và gọi hàm xử lý để cập nhật giá trị cho control đó.
- Cần phải khởi tạo state cho control trước khi sử dụng
- Cập nhật dữ liệu của từng control thông qua cú pháp **event.target.value**

FORM

Xử lý form

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = { username: '' };
  }
  myChangeHandler = (event) => {
    this.setState({username: event.target.value});
  }
  render() {
    return (
      <form>
      <h1>Hello {this.state.username}</h1>
      <p>Enter your name:</p>
      <input
        type='text'
        onChange={this.myChangeHandler}
      />
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

FORM

- Để gửi 1 form đi có thể sử dụng event **onSubmit** đặt trong thẻ form
- Lưu ý dử dụng **event.preventDefault()** để ngăn trang tải lại

FORM

Submit form

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = { username: '' };
  }
  mySubmitHandler = (event) => {
    event.preventDefault();
    alert("You are submitting " + this.state.username);
  }
  myChangeHandler = (event) => {
    this.setState({username: event.target.value});
  }
```

```
  render() {
    return (
      <form onSubmit={this.mySubmitHandler}>
      <h1>Hello {this.state.username}</h1>
      <p>Enter your name, and submit:</p>
      <input
        type='text'
        onChange={this.myChangeHandler}
      />
      <input
        type='submit'
      />
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

FORM

- Để kiểm soát nhiều control cần thêm thuộc tính name vào cho mỗi control
- Khi khởi tạo state cho mỗi control cần thiết lập tên giống với giá trị của thuộc tính name
- Để lấy state name mỗi control khi onChange sử dụng cú pháp **event.target.name**
- Sử dụng cặp ngoặc **[]** để bọc bên ngoài của state name như sau **[event.target.name]**

FORM

Xử lý nhiều control đầu vào

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      username: '',
      age: null,
    };
  }
  myChangeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
      <h1>Hello {this.state.username} {this.state.age}</h1>
      <p>Enter your name:</p>
      <input
        type='text'
        name='username'
        onChange={this.myChangeHandler}
      />
      <p>Enter your age:</p>
      <input
        type='text'
        name='age'
        onChange={this.myChangeHandler}
      />
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

- Bất kỳ dữ liệu nào trước khi được gửi đi cũng cần phải kiểm tra tính hợp lệ của dữ liệu
- Có thể kiểm tra tính hợp lệ của dữ liệu trong quá trình thay đổi dữ liệu trên control hoặc khi submit

FORM

Xử lý nhiều control đầu vào – Validate onchange

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      username: '',
      age: null,
    };
  }
  myChangeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    if (nam === "age") {
      if (!Number(val)) {
        alert("Your age must be a number");
      }
    }
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
      <h1>Hello {this.state.username} {this.state.age}</h1>
      <p>Enter your name:</p>
      <input
        type='text'
        name='username'
        onChange={this.myChangeHandler}
      />
      <p>Enter your age:</p>
      <input
        type='text'
        name='age'
        onChange={this.myChangeHandler}
      />
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

FORM

Xử lý nhiều control đầu vào – Validate submit

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      username: '',
      age: null,
    };
  }
  mySubmitHandler = (event) => {
    event.preventDefault();
    let age = this.state.age;
    if (!Number(age)) {
      alert("Your age must be a number");
    }
  }
  myChangeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form onSubmit={this.mySubmitHandler}>
      <h1>Hello {this.state.username} {this.state.age}</h1>
      <p>Enter your name:</p>
      <input
        type='text'
        name='username'
        onChange={this.myChangeHandler}
      />
      <p>Enter your age:</p>
      <input
        type='text'
        name='age'
        onChange={this.myChangeHandler}
      />
      <br/>
      <br/>
      <input type='submit' />
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

FORM

- Thiết lập các thông báo phía bên dưới mỗi control khi khi không hợp lệ
- Sử dụng state để hứng lỗi

FORM

Thiết lập thông báo lỗi

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      username: '',
      age: null,
      errormessage: ''
    };
  }
  myChangeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    let err = '';
    if (nam === "age") {
      if (val !="" && !Number(val)) {
        err = <strong>Your age must be a number</strong>;
      }
    }
    this.setState({errormessage: err});
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
      <h1>Hello {this.state.username} {this.state.age}</h1>
      <p>Enter your name:</p>
      <input
        type='text'
        name='username'
        onChange={this.myChangeHandler}
      />
      <p>Enter your age:</p>
      <input
        type='text'
        name='age'
        onChange={this.myChangeHandler}
      />
      {this.state.errormessage}
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

FORM

- Để lấy giá trị của option trong select cần thiết lập giá trị cho các option
- Sử dụng state để cập nhật giá trị nhận được trong thẻ select

FORM

Lấy giá trị option trong Select

```
class MyForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      mycar: 'Volvo'
    };
  }
  render() {
    return (
      <form>
      <select value={this.state.mycar}>
        <option value="Ford">Ford</option>
        <option value="Volvo">Volvo</option>
        <option value="Fiat">Fiat</option>
      </select>
      </form>
    );
  }
}

ReactDOM.render(<MyForm />, document.getElementById('root'));
```

CSS

- Có nhiều cách để thiết lập CSS cho ReactJS.
- Trong bài này sẽ hướng dẫn 3 cách sử dụng cho 1 ứng dụng ReactJS cơ bản
  - **Inline style CSS**
  - **CSS Stylesheet**
  - **CSS Module**

CSS

- Trong ReactJS CSS được xem là object
- Để thêm css vào 1 tag cần sử dụng thuộc tính style
- Đặt thuộc tính css bên trong 2 cặp ngoặc nhọn **{{}}**
- Sử dụng cấu trúc camelCased (cách viết biến kiểu lạc đà) để chuyển đổi cho thuộc tính css có dấu –
- Ngoài ra có thể đặt biến cho object css sao đó gán vào thuộc tính style

CSS

Inline CSS

```
class MyHeader extends React.Component {
  render() {
    return (
      <div>
      <h1 style={{backgroundColor: "lightblue"}}>Hello Style!</h1>
      <p>Add a little style!</p>
      </div>
    );
  }
}
```

```
class MyHeader extends React.Component {
  render() {
    const mystyle = {
      color: "white",
      backgroundColor: "DodgerBlue",
      padding: "10px",
      fontFamily: "Arial"
    };
    return (
      <div>
      <h1 style={mystyle}>Hello Style!</h1>
      <p>Add a little style!</p>
      </div>
    );
  }
}
```

CSS

- Đây là cách quen thuộc nhất trong 3 cách được nêu ở trên
- Tạo ra 1 file css như thông thường và viết mã HTML như css bình thường.
- Import vào component cần sử dụng thông qua cú pháp **import filestyle.css;**

CSS

CSS Stylesheet

```
body {
  background-color: #282c34;
  color: white;
  padding: 40px;
  font-family: Arial;
  text-align: center;
}
```

```
import React from 'react';
import ReactDOM from 'react-dom';
import './App.css';

class MyHeader extends React.Component {
  render() {
    return (
      <div>
      <h1>Hello Style!</h1>
      <p>Add a little style!.</p>
      </div>
    );
  }
}

ReactDOM.render(<MyHeader />, document.getElementById('root'));
```

CSS

- Để tránh xung đột về đặt tên class dẫn đến trùng css ở các component. React đã đưa ra giải pháp mã hóa tên class ở mỗi component riêng lẽ để tránh trùng lập css.
- Bằng cách thêm hậu tố **.module.css** khi tạo file css là đã tạo được 1 file CSS Module
- Cách viết CSS tương tự như cách thứ 2

CSS

CSS Module

```
body {
  background-color: #282c34;
  color: white;
  padding: 40px;
  font-family: Arial;
  text-align: center;
}
```

```
import React from 'react';
import ReactDOM from 'react-dom';
import styles from './mystyle.module.css';

class Car extends React.Component {
  render() {
    return <h1 className={styles.bigblue}>Hello Car!</h1>;
  }
}

export default Car;
```

SASS/SCSS

- Sass được gọi là tiền biên dịch CSS (pre-processor)
- Các cú pháp của Sass sẽ được xử lý ở server sau đó sẽ gửi mã CSS vào trình duyệt
- Tìm hiểu thêm về Sass tại https://www.w3schools.com/sass/default.asp hoặc https://sass-lang.com/

SASS/SCSS

- Để sử dụng Sass trong project ReactJS, trước tiên cần cài đặt 1 package có tên là **node-sass**
- Package này chịu trách nhiệm đọc và biên dịch mã Sass thành CSS thông thường.
- Tạo 1 file có đuôi là .scss hoặc .sass để sử dụng các cú pháp của SASS

SASS/ SCSS

```
$myColor: red;

h1 {
  color: $myColor;
}
```

```
import React from 'react';
import ReactDOM from 'react-dom';
import './mysass.scss';

class MyHeader extends React.Component {
  render() {
    return (
      <div>
      <h1>Hello Style!</h1>
      <p>Add a little style!.</p>
      </div>
    );
  }
}

ReactDOM.render(<MyHeader />, document.getElementById('root'));
```

Thank you and
happy learning !!!