**บทที่ 8 รู้จักและใช้งาน React Lifecycle**

**ตัวอย่างการตรวจสอบ React Lifecycle**

import React, {Component} from 'react';

class App extends Component {

// 1 รับค่า default props

// 2 กำหนดค่าเริ่มต้นให้กับ state

state = {

name: "ค่าเริ่มต้นให้ state"

};

// 3 ก่อน render

componentWillMount() {

console.log("3 ก่อน render");

}

// 4 ขณะ render

render() {

console.log('4 ขณะ render', this.props);

return (

<div>

{this.state.name}

</div>

)

}

// 5 หลังจาก render เสร็จแล้ว

componentDidMount() {

console.log("5 หลังจาก render แล้ว");

}

}

export default App;

**Lifecycle เมื่ออัพเดทค่า state ในคอมโปเน้นท์**

import React, {Component} from 'react';

class App extends Component {

state = {

name: 'Jeerawuth'

}

componentWillMount() {

console.log('Component Will Mount');

}

render() {

console.log('Component Render');

return (

<div onClick={this.userClick}>

Hello!, Life Cycle!!! {this.state.name}

</div>

)

}

componentDidMount() {

console.log('Component Did Mount!!!!');

}

componentWillUpdate() {

console.log('Component Will Update??');

}

componentDidUpdate() {

console.log('Component Did Update!!!!');

}

userClick = () => {

this.setState({

name: 'Watsarun'

})

}

}

export default App;

**ตรวจสอบค่า props กับ state ก่อน render()**

shouldComponentUpdate(nextProps, nextState) {

return true;

}

import React, {Component} from 'react';

class App extends Component {

state = {

name: 'Jeerawuth',

status: true

}

componentWillMount() {

console.log('Component Will Mount');

}

render() {

console.log('Component Render');

return (

<div onClick={this.userClick}>

Hello!, Life Cycle!!! {this.state.name}

</div>

)

}

componentDidMount() {

console.log('Component Did Mount!!!!');

}

componentWillUpdate() {

console.log('Component Will Update??');

}

componentDidUpdate() {

console.log('Component Did Update!!!!');

}

shouldComponentUpdate(nextProps, nextState) {

if (nextState.status === false) {

return false;

}

return true;

}

userClick = () => {

this.setState({

name: 'Jeerawuth',

status: false

})

}

}

export default App;

**Lifecycle เมื่อได้รับค่า props มาใหม่**

import React, {Component} from 'react';

class Child extends Component {

state = {

name: 'Jeerawuth',

status: true

}

componentWillMount() {

console.log('Component Will Mount');

}

render() {

console.log('Component Render');

return (

<div>

<button onClick={this.userClick}>

State Change!!!

</button>

<button onClick={this.props.counterFn}>

Props Change!!!

</button>

</div>

)

}

componentDidMount() {

console.log('Component Did Mount!!!!');

}

componentWillUpdate() {

console.log('Component Will Update??');

}

componentDidUpdate() {

console.log('Component Did Update!!!!');

}

componentWillReceiveProps() {

console.log('I have got new props!!');

}

userClick = () => {

this.setState({

name: 'Jeerawuth',

status: false

})

}

}

export default Child;

import React, {Component} from 'react';

import Child from './Child';

class App extends Component {

state = {

counter: 1

}

plusCounter() {

this.setState({

counter: this.counter + 1

})

return this.state.counter;

}

render() {

return (

<div>

<Child counterFn={this.plusCounter.bind(this)} />

</div>

)

}

}

export default App;

**เคลียร์เนื้อหาก่อนออกจากคอมโปเน้นท์**

import React from 'react';

import { BrowserRouter, Route, Link } from 'react-router-dom';

import 'bootstrap/dist/css/bootstrap.min.css';

import Child from './Child';

const App = () => {

return (

<BrowserRouter>

<div>

<ul>

<li><Link to="/">Home</Link></li>

<li><Link to="/child">Child</Link></li>

</ul>

<Route path="/child" component={Child} />

</div>

</BrowserRouter>

);

}

export default App;

import React, {Component} from 'react';

class Child extends Component {

state = {

name: 'Jeerawuth',

status: true

}

componentWillMount() {

console.log('Component Will Mount');

}

render() {

console.log('Component Render');

return (

<div>

Hello This is the Child!!!

<button onClick={this.userClick}>Change State</button>

</div>

)

}

componentDidMount() {

console.log('Component Did Mount!!!!');

}

componentWillUpdate() {

console.log('Component Will Update??');

}

componentDidUpdate() {

console.log('Component Did Update!!!!');

}

componentWillReceiveProps() {

console.log('I have got new props!!');

}

componentWillUnmount() {

console.log('Unmount LifeCycle Component!!!!!');

}

userClick = () => {

this.setState({

name: 'Jeerawuth',

status: false

})

}

}

export default Child;

**ป้องกันการ render ซ้ำหากค่าใน state ไม่เปลี่ยน**

**วิธีที่ 1 ตรวจสอบค่า state กับ nextState**

import React, {Component} from 'react';

class App extends Component {

state = {

status: true

}

shouldComponentUpdate(nextProps, nextState) {

if (this.state.status === nextState.status) {

return false

} else {

return true;

}

}

render() {

console.log('Current State is: ' + this.state.status);

return(

<div>

<button onClick={this.clickMeHandler}>Change State</button>

<hr />

{

this.showHide()

}

</div>

)

}

checkCondition = () => {

return this.state.status

}

showHide = () => {

return (

this.checkCondition()?

<div>Show</div>:

<div>Hide</div>

)

}

clickMeHandler = () => {

this.setState({

status: false

});

}

}

export default App;

**วิธีที่ 2 ใช้คลาส PureComponent**

import React, {PureComponent} from 'react';

class App extends PureComponent {

state = {

status: true

}

render() {

console.log('Render for component!!!');

return(

<div>

<button onClick={this.clickMeHandler}>Click me to change</button>

<hr />

{

this.showHide()

}

</div>

)

}

checkCondition = () => {

return this.state.status

}

showHide = () => {

return (

this.checkCondition()?

<div>Show</div>:

<div>Hide</div>

)

}

clickMeHandler = () => {

this.setState({

status: false

});

}

}

export default App;