Assignment 3 – Props & State

Hands on Task

1. Implement class and functional components that print:
   1. Class component should print Your First Name
   2. Functional component should print Your Last Name
   3. Have children and print that out as well.

**import React from "react";**

**class ClassComponent extends React.Component{**

**render(){**

**return <p > First Name: {this.props.fname} {this.props.children} </p>**

**}**

**}**

**export default ClassComponent;**

**function FunctionalComponent (props) {**

**return <p> Last Name :{props.lname} {props.children}</p>**

**}**

**export default FunctionalComponent;**

**import React from "react";**

**import ClassComponent from "./classComponent";**

**import FunctionalComponent from "./functionComponent";**

**export default function App() {**

**return ( <div className = "App">**

**<ClassComponent fname = "Yogesh">-Printed First-Name from Class Component as children</ClassComponent>**

**<FunctionalComponent lname= "Ghimiray"> -Printed Last-Name from Functional Component as children </FunctionalComponent>**

**</div> );**

**}**

1. Implement increment counter.

**import React from "react";**

**import ClassComponent from "./ClassComponent";**

**import FunctionalComponent from "./FunctionComponent";**

**class App extends React.Component {**

**state = {counter:0};**

**increamentCounter = ()=>{**

**this.setState({counter: this.state.counter +1});**

**};**

**render(){**

**return (**

**<div className = "App">**

**<ClassComponent fname = "Yogesh">-Printed First-Name from Class Component as children</ClassComponent>**

**<FunctionalComponent lname= "Ghimiray"> -Printed Last-Name from Functional Component as children </FunctionalComponent>**

**<button onClick = {this.increamentCounter} > Counter Increament {this.state.counter} </button>**

**</div> );**

**}**

**}**

**export default App;**

1. Implement the students component.

**import React from "react";**

**class Student extends React.Component{**

**render(){**

**return (**

**<div>**

**<hr/>**

**{this.props.children}**

**<p> Name : {this.props.name} </p>**

**<p> Age : {this.props.age} </p>**

**</div>**

**)**

**};**

**}**

**export default Student;**

**import React from "react";**

**import Student from "./Student";**

**class App extends React.Component {**

**state = {students:[**

**{name:"John", age :10},**

**{name: "Mary", age: 11}**

**]};**

**render(){**

**return (**

**<div className = "App">**

**<Student name={this.state.students[0].name} age={this.state.students[0].age} > First Student </Student>**

**<Student name={this.state.students[1].name} age={this.state.students[1].age} > Second Student </Student>**

**</div> );**

**}**

**}**

**export default App;**

1. Implement the **MakeOlder**demo app.
   1. Instead of hardcoded values in the makeOlder method, increase all students’ age one by one when click on the button using the “for of” loop.

**import React from "react";**

**class Student extends React.Component{**

**render(){**

**return (**

**<div>**

**<hr/>**

**{this.props.children}**

**<p> Name : {this.props.name} </p>**

**<p> Age : {this.props.age} </p>**

**</div>**

**)**

**};**

**}**

**export default Student;**

**import React from "react";**

**import Student from "./Student";**

**class App extends React.Component {**

**state = {students:[**

**{name:"John", age :10},**

**{name: "Mary", age: 11}**

**]};**

**makeOlder = ()=>{**

**const students = this.state.students;**

**for (const student of students){**

**student.age = student.age +1;**

**this.setState({students});**

**}**

**};**

**render(){**

**return (**

**<div className = "App">**

**<Student name={this.state.students[0].name} age={this.state.students[0].age} > First Student </Student>**

**<Student name={this.state.students[1].name} age={this.state.students[1].age} > Second Student </Student>**

**<button onClick = {this.makeOlder} > Make Older {this.state.counter} </button>**

**</div> );**

**}**

**}**

**export default App;**

* 1. Create a method **Make Younger.** Pass that as a prop to the child Student component. When click on the Student component’s <p> tage, it should decrease the age.

**import React from "react";**

**class Student extends React.Component{**

**render(){**

**return (**

**<div>**

**<hr/>**

**{this.props.children}**

**<p> Name : {this.props.name} </p>**

**<p> Age : {this.props.age} </p>**

**</div>**

**)**

**};**

**}**

**export default Student;**

**import React from "react";**

**import Student from "./Student";**

**class App extends React.Component {**

**state = {students:[**

**{name:"John", age :10},**

**{name: "Mary", age: 11}**

**]};**

**makeYounger = ()=>{**

**const students = this.state.students;**

**for (const student of students){**

**student.age = student.age - 1;**

**this.setState({students});**

**}**

**};**

**render(){**

**return (**

**<div className = "App">**

**<Student name={this.state.students[0].name} age={this.state.students[0].age} > First Student </Student>**

**<Student name={this.state.students[1].name} age={this.state.students[1].age} > Second Student </Student>**

**<button onClick = {this.makeYounger} > Make Younger {this.state.counter} </button>**

**</div> );**

**}**

**}**

**export default App;**

* 1. Pass **index**to the method and only update the clicked student’s age.

1. Add a new state count to the app.js and increment it with Make Older using the setState with the **updater function**.

**import React from "react";**

**class Student extends React.Component{**

**render(){**

**return (**

**<div>**

**<hr/>**

**{this.props.children}**

**<p> Name : {this.props.name} </p>**

**<p onClick= {()=>{**

**this.props.increaseAge(this.props.index)**

**}}> Age : {this.props.age} </p>**

**</div>**

**)**

**};**

**}**

**export default Student;**

**import React from "react";**

**import Student from "./Student";**

**class App extends React.Component {**

**state = {students:[**

**{name:"John", age :10},**

**{name: "Mary", age: 11}**

**]};**

**makeOlder = ()=>{**

**const students = this.state.students;**

**for (const student of students){**

**student.age = student.age +1;**

**this.setState({students});**

**}**

**};**

**increaseIndividualAge = (index)=>{**

**const students = this.state.students;**

**students[index].age = students[index].age +1;**

**this.setState({students});**

**}**

**render(){**

**return (**

**<div className = "App">**

**<Student name={this.state.students[0].name} age={this.state.students[0].age} index={0} increaseAge={this.increaseIndividualAge}> First Student </Student>**

**<Student name={this.state.students[1].name} age={this.state.students[1].age} index={1} increaseAge={this.increaseIndividualAge} > Second Student </Student>**

**{/\* <button onClick = {this.makeOlder} > Make Older {this.state.counter} </button> \*/}**

**</div> );**

**}**

**}**

**export default App;**

Research

1. Read the lecture thoroughly. In your opinion, what are the 3 main points? Please share.
2. What is DOM?

**The Document Object Model (DOM) is an application programming interface (API) for HTML. With the Document Object Model, programmers can build documents, navigate their structure, and add, modify, or delete elements and content.**