**React - Components, State, Props**

**Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML. Components come in two types, Class components and Function components, in this tutorial we will concentrate on Function components.**

**1)Class Component :** React Class components have a built-in state object. You might have noticed that we used state earlier in the component constructor section. The state object is where you store property values that belongs to the component. When the state object changes, the component re-renders.

Ex:

import React from 'react';

class StateEx extends React.Component

{

constructor()

{

super();

this.state={count :0}

}

inc = () =>

{

this.setState((prev)=>(

{count:prev.count+1}

))

}

render()

{

return(

<>

{this.state.count}

<button onClick={this.inc}>Inc</button>

</>

)

}

}

export default StateEx;

2) Function Component : Functional component is just a simple javascript function; it accepts the data in the form of props and returns the react element. Whereas the class component will be created using the class keyword, and it extends the React. Component to make the class as a react component.

Ex:

import React, from 'react'

Function Example()

{

Return(

<div>

Any tage and code…..

</div>

)

}

export default Example;

State

The state is a built-in React object that is used to contain data or information about the component.

**Ex :**

**import React, { useState } from 'react'**

**function UseStateArray() {**

**const [dataX,setDataX] = useState([1,2,"hi"])**

**const handleChangeX = ()=>{**

**setDataX([dataX[1]=34])**

**}**

**return (**

**<div>**

**{/\* <h2>Index 0: value {dataX[0]}</h2> \*/}**

**<h2 onClick={handleChangeX}>Index 1: {dataX[1]}</h2>**

**{/\* <h2>Index 2: value {dataX[2]}</h2> \*/}**

**</div>**

**)**

**}**

**export default UseStateArray;**

**ETC…**

**Props**

Props are an important concept to understand in React. You use props to pass data and values from one component to another to get dynamic and unique outputs.

**Ex :**

**import React from 'react'**

**class ClassPropEx extends React.Component**

**{**

**render()**

**{**

**return(**

**<>**

**test123**

**{this.props.name}**

**{this.props.age}**

**{this.props.course}**

**<ClassPropEx1 name="xyz"/>**

**</>**

**)**

**}**

**}**

**export default ClassPropEx;**

**class ClassPropEx1 extends React.Component**

**{**

**render()**

**{**

**return(**

**<>**

**<h3> THIS IS CHILD COMPO</h3>**

**USername : {this.props.name}**

**</>**

**)**

**}**

**}**

**Props Type EX :**

**import React from 'react';**

**import PropTypes from "prop-types";**

**function PropTypesEx(prop) {**

**return (**

**<div>**

**{prop.arrayProp}**

**{prop.stringProp}**

**{prop.numberProp}**

**</div>**

**)**

**}**

**PropTypesEx.propTypes = {**

**arrayProp:PropTypes.array,**

**stringProp:PropTypes.string,**

**numberProp:PropTypes.number**

**}**

**PropTypesEx.defaultProps = {**

**arrayProp : [1,2,3,4],**

**stringProp : "Hello",**

**numberProp :10**

**}**

**export default PropTypesEx**