**State ها در react**

The state is a **built-in React** object that is used to contain data or information about the component. A component's state can change over time; whenever it changes, the component re-renders

Difference between props and state :

1 – **PROPS** It is Immutable (cannot be modified) but The **STATE** It is Mutable ( can be modified).

2 – **PROPS** can be used with state and functional components. The **STATE** can be used only with the state components/class component (Before 16.0).

3 – **PROPS** are read-only. The **STATE** is both read and write.

**نکته :** هر وقت در ری اکت موردی را ایمپورت کردید که با use شروع می شد بدانید آن یک هوک است .

مثال : useState

const [count , setCount] = useState(0)

هوک useState یک آرایه به ما بر می گرداند

ایتدکس 0 = مقدار خروجی ( count )

ایندکس 1 = متدی که به ما آجازه می دهد state را تغییر بدهیم . ( setCount )

**تفاوت state در کامپوننت های نوع کلاسی و تابعی در چیست ؟**

در کامپوننت ها نوع کلاسی زمانی که یک state رو ویرایش می کنید بقیه state ها بدون تغییر می مانند اما در کامپوننت های نوع تابعی زمانی که یک state رو تغییر می دهیم بقیه state ها پاک می شود . !!! و برای جلوگیری از این کار باید برای هر متغیر یک state رو تعریف کنیم .

**مدیریت State ها در react**

**مدیریت رویداد در کامپوننت های نوع کلاسی به صورت زیر هست :**

**روش اول :**

import { Component } from "react";

class Counter extends Component {

    constructor() {

        super();

        this.state = {

            name: "amin"

        }

        this.changeName = this.changeName.bind(this);

    }

    changeName (){

        if(this.state.name === 'amin')

            this.setState({name : "ali"});

        else

            this.setState({name : "amin"});

    }

    render() {

        return (

            <>

                <p>{this.state.name}</p>

                <button onClick={this.changeName}>تغییر نام</button>

            </>

        )

    }

}

export default Counter;

در این روش فانکشن تغییر دهنده state باید بایند کنیم .

this.changeName = this.changeName.bind(this);

اما در روش دوم اگر ما فانکشن رو به صورت فانکشن های فلشی تعریف کنیم نیاز به بایند کردن نیست یعنی به جای تعریف به صورت زیر :

changeName (){

        if(this.state.name === 'amin')

            this.setState({name : "ali"});

        else

            this.setState({name : "amin"});

    }

به این ضورت استفاده کنیم : ( فانکشن های نوع فلشی )

در روش پایین نیازی نیست که کد بایند کردن را بنویسیم

changeName = () => {

    if (this.state.name === 'amin')

        this.setState({ name: "ali" });

    else

        this.setState({ name: "amin" });

}

**مدیریت رویداد ها در کامپوننت های نوع تابعی**

در یک کامپوننت پدر مثلا App.jsx یک state به نام count رو تعریف کردیم و چندین متد برای تغییر وضعیت state ایجاد کردیم

increaseCount

decreaseCount

resetCount

const App = () => {

  const [count , setCount] = useState(0);

  const increaseCount = () =>{

    setCount(count + 1);

  }

  const decreaseCount = () =>{

    setCount(count - 1);

  }

  const resetCount = () =>{

    setCount(0);

  }

  return (

    <>

      <Counter inc={increaseCount} dec={decreaseCount} res={resetCount} count={count}/>

    </>

  );

}

export default App;

حال در کامپوننت Counter فانکشن های تعریف شده در کامپوننت App را به صورت props به Counter می دهیم و از آن ها استفاده می کنیم .

const Counter = ({inc , dec , res , count}) =>{

    return(

        <>

            <h2>شمارنده من : </h2>

            <p>{count}</p>

            <br/>

            <button onClick={inc}>+</button>

            <button onClick={dec}>-</button>

            <button onClick={res}>🎇</button>

        </>

    )

}

export default Counter;

**اعتبار سنجی Props ها**

از کجا بفهمیم props های درستی را به Counter ارسال کردیم ؟

استفاده از PropsType

در سایت npm.js نصب کنید .....

npm i prop-types

نحوه استفاده بعد از Import

import PropTypes from 'prop-types'

Counter.propTypes={

    inc: PropTypes.func ,

    dec: PropTypes.func ,

    res: PropTypes.func ,

    count: PropTypes.number

}

به این صورت مشخص می کنیم هر prop دریافتی از چه نوعی هست . . . .

در صورتی که از کامپوننت والد تایپ اشتباهی رو برای prop ارسال کنیم خطایی با این شکل و شمایل دریافت می کنید . . . .

