MODULR: REACT-COMPONENTS ,STATE,PROPS

1)COMPONENTS

ANS: we can say that every application you will develop in React will be made up of pieces called components.

=>In React, we mainly have two types of components:

1) [**Functional Components**](https://www.geeksforgeeks.org/reactjs-functional-components/)

20 [**Class Components**](https://www.geeksforgeeks.org/reactjs-class-based-components/):

2)STATE

ANS: React has another special built-in object called state, which allows components to create and manage their own data. So unlike props, components cannot pass data with state, but they can create and manage it internally.

=> Here is an example showing how to use state:

EX=class Test extends React.Component {

constructor() {

this.state = {

id: 1,

name: "test"

};

}

render() {

return (

<div>

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

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

</div>

);

}

}

### =>How do you update a component’s state?

State should not be modified directly, but it can be modified with a special method called setState( ).

this.state.id = “2020”; // wrong

this.setState({ // correct

id: "2020"

});

### =>What happens when state changes?

A change in the state happens based on user-input, triggering an event, and so on. Also, React components (with state) are rendered based on the data in the state. State holds the initial information.

So when state changes, React gets informed and immediately re-renders the DOM – **not the whole DOM, but only the component with the updated state.**This is one of the reasons why React is fast.

And how does React get notified?

You guessed it: with setState( ). The setState( ) method triggers the re-rendering process for the updated parts. React gets informed, knows which part(s) to change, and does it quickly without re-rendering the whole DOM.

In summary, there are 2 important points we need to pay attention to when using state:

* State shouldn’t be modified directly – the setState( ) should be used
* State affects the performance of your app, and therefore it shouldn’t be used unnecessarily

3)PROPS

ANS: Props is short for properties and they are used to pass data between React components. React’s data flow between components is uni-directional (from parent to child only).

**How do you pass data with props?**

ANS:

=>Here is an example of how data can be passed by using props:

=> EX= class ParentComponent extends Component {

render() {

return (

<ChildComponent name="First Child" />

);

}

}

const ChildComponent = (props) => {

return <p>{props.name}</p>;

};

=>

Firstly, we need to define/get some data from the parent component and assign it to a child component’s “prop” attribute.

<ChildComponent name="First Child" />

=>“Name” is a defined prop here and contains text data. Then we can pass data with props like we’re giving an argument to a function:

const ChildComponent = (props) => {

// statements

};

=>And finally, we use dot notation to access the prop data and render it:

return <p>{props.name}</p>;