REACT

LESSON 7 – State

CO2214 - PRACTICAL WORK ON CO2224

Props vs State

Props	State
Props get passed to the component.	State is managed within the component.
Function parameters	Variables declared in the function body.
props are immutable. (Parents own the props, cannot be changed by children)	State can be changed. (Component has full control to change the States.)
props - Functional Components	useState Hook - Functional Components
this.props - Class Components	this.state - Class Components

- Ultimately props and State both hold information and influences UI in the browser.

How State can be used in Class Components?

- Go to Components Folder and Create **Message.js** file
- Write code in **Message.js** file

```
import React, {Component} from 'react'

class Message extends Component {
   render ()
   {
      return <h1>Welcome visitor</h1>
   }
```

```
}
export default Message
```

- in App.js

```
import Message from './Components/Message';

function App () {
  return (
    <div className="App">
     <Message />
   </div>
  );
}

export default App;
```

- Now we are going to Create Subscribe Button below the text and when we click on the button the text being displayed should change from Welcome Visitor to Thank you for Subscribing.

- For that we have to use Component State.

in **Message.js**

- First step is to create an object and initialized it and the step is usually done in the Class Constructor.

- Within the Constructor we call super () this is required, because we extend React Components Class and it call has to be made to the base class Constructor.

```
constructor ()
{
        super ()
        this.state ={

        }
}
```

- So, we created the state object. Let's go ahead and initialize the property.
- We are going to call message and the value is **Welcome Visitor**.

```
constructor ()
   {
        super ()
        this.state={
             message: 'Welcome Visitor'
        }
   }
```

- Now the second step is to bind this state value in the render function. to do this we use this.state instead of this.props in the return statements within curly braces.

```
render ()
   {
      return <h1> {this.state.message} </h1>
   }
```

- Now look at the browser, there is no any changes in output, but we are using state so, we have ability to change the message.
- Let's create a button and on click on that button will change the message.

```
render ()
   {
      return (
      <div>
      <h1>{this.state.message}</h1>
      <button>Subscribe</button>
      </div>
   )
}
```

- Last Step is click to the button and Change Message, to that on the button element we add onClick attribute and very important the event in camelCase and C in Uppercase.

```
<button onClick = { ( ) => this.changeMessage ( ) }>Subscribe
</button>
```

- after the constructor and before the render message to change state of the component we need to call setstate method within the body of changeMessage.

- This method accepts an object which is nothing but new state of the component.in the message we are going to add the new message.

```
changeMessage()
   {
      this.setState({
         message:'Thank you for Subscribing'
      })
   }
```

- Now Look at the browser and click Subscribe button and see the changes.

❖ State is nothing but an object that is privately maintain inside the Component.

❖ State can influence what is rendered in the browser.

❖ State can be change within the component.