Lecture #9

React me class compt or function compt use hoty hen pehle class compt ko smjhty hen ,

Asani k liye ham class ko 3 hisson me taqseem kr skty hen :

```
class A{
    //1. Properties


    //2. Constructor

    //3. Methods
}
```

Aap ek **class** ko samajhne aur organize karne ke liye 3 hisso me taqseem kar sakte hain. Yeh breakdown class structure ko zyada logical aur easy to understand banata hai. Class ka structure kuch is tarah se samjha ja sakta hai:

1. Properties (Instance Variables/Fields):

- Is hissay me aap **properties** define karte hain, jo har object ke liye specific hoti hain. In properties ko constructor function ke andar initialize kiya jata hai.
- These are like variables that hold data related to the object (e.g., name, age).

2. Constructor:

- **Constructor** ek special method hota hai jo class ke objects banane ke liye use hota hai. Jab bhi class ka koi naya object banaya jata hai, constructor method automatic call hota hai.
- Isme properties ko initialize kiya jata hai.

3. Methods (Functions/Behaviors):

- Is area me aap **methods** define karte hain, jo actions ya behaviors ko represent karte hain. Methods ka kaam hota hai class ke objects par operations perform karna.
- These could include functions to get or set property values, perform calculations, etc.

Key Points:

- **Properties**: Har object ke specific data ko store karne ke liye.
- **Constructor**: Object ko banane ke waqt initialize karta hai.
- **Methods**: Object par actions ya operations perform karne ke liye.

*Class me mojood variabkes ko initialize krne ki zimadari constructor function ki ha

*aesi property jo class me define hi na ho wo bhi constructor(0 me initialize ho skti ha

*

```
> A > constructor
class A{
    //1. Properties/Variable
    name; // This is only decleration
    surname=''; // THis is decleration and initialization
    address='neemuch';// THis is decleration and initialization

    //2. Constructor
    constructor(){
        // The role of constructor is to initilize the properties
        //this.member
        this.name = 'Anil';
        this.surname="Dollor";
        this.fatherName='';
    }

    //3. Methods
```

```
//3. Methods
showName(){
    console.log(this.name); //this is an internal object
}
```

Ab hamen ek object create kna pare ga : uska generic formula ye ha :

```
// Lets create the class Object
// let object = new ClassName();

let obj = new A();

// this obj is an external object
// object.member
obj.showName() // . is member selection operator
```

Ab ye file test ki ha q k ham apne concrpts ko revise kr rhy hen to isko trminal p chala kr output check krty hen:

```
PROBLEMS    OUTPUT    TERMINAL    COMMENTS

PS D:\react\proj8> node .\test.js
AnilDollor
PS D:\react\proj8>
```

****

Now ab ham ek or class define krty hen B :

```
26 class B extends A{
27     //1 Properties
28     friends1; // This is only decleration
29     friends2=''; // THis is decleration and initialization
30
31     //2. Constructor
32     constructor(frnd3){ //frnd3 is formal argument
33         super();
34         this.friends1='Rakesh';
35         this.friends2='Dev';
36         this.friends3=frnd3;
37     }
38
39
40     //3. Methods
41     listMyFriends(){
42         console.log(this.friends1);
43         console.log(this.friends2);
```

Ab jo “frnd3” ha wo accept hoga wahan se jab ham is class se ek object instatntited kren gy .

```
42 }
43
44 let obj2 = new B('Pushpendra');
45
```

Super() is used to call parent constructor

Obj2 is the object of class B but isme hamne inherits kiye hen class A k saray methods.

So:

```
51
52 obj2.listMyFriends();
53 obj2.showName();
```

****

Ab ham ne class revise krli ha to ab ham index.js me ja kr practice k liye ek class compt banaty hen:

```
index.html M    index.js M
src > index.js > A > render
import ReactDOM from 'react-dom/client';

class A extends  React.Component{
  //1.


  //2.
  constructor(){
    super();
  }


  //3
  render(){
    return <h1>OKLABS</h1>;
  }
}
```

Ab ham is compt ko root me render kr dety hen>

```
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<A name="Anil" surname="Dollor">Neemuch</A>);
```

Ab is compt ko ham data pass kr rhy hen jisko ham compt me receive kren gy !

Name & surname is coming as a prop and city name is coming as children.

Remember! Every compt can have its own data in react we call it as “State”.

Now assume kro k property area me hamary pass ek ‘state’ para hoa ha .

```
class A extends  React.Component{
    //1.
    state;


  //2.
  constructor(){
    super();
    this.state = {}
  }
```

Ab constructor me ham state ko define krty hen kuch values put krty hen .

```
//2.
constructor(){
  super();
  this.state = {name:"Rakesh",surname:"Sharma",address:"Manasa"}
}
```

Ab ham is state ko access kr skty hen :

```
  //3
  render(){
    return <h1>OKLABS {this.state.name}</h1>;
  }
}
```

Means that compt ka internal kuch data ha jo ham compt k under state me define krty hen or ek data wo ha jo bahior se aata ha jo ham props or children k through receive krty hen.

State class k under hota ha . class compt me agar data mojood ha to wo state area me hi hoga .

```
render(){
  return  <>
            <h1>OKLABS {this.state.name} {this.state.surname}</h1>
            <h1>OKLABS {this.state.address}</h1>
            <h1>OKLABS {this.props.name}</h1>
          </>
}
```

So ham state area me compt ka data define krty hen

State is a JS object which holds compt's internal data

```
5
6  //Every Compoent can have its own data/states
7  class A extends  React.Component{
8      //1.
9
10
11   //2.
12   constructor(){
13     super();
14     this.state = {
15                   name:"Rakesh",
16                   surname:"Sharma",
17                   address:"Manasa"}// state initialization
18   }
19
```

<u>Here comes a new thing</u>

<u>Changing State object</u>

Ab ham state me data initialize kr lia ha ab usko ham change kr skty hen . us k liye react hamen ek built-in method deta ha setState() …… ye method apne parent se aa raha ha jo k REACT.COMPONENT ha.

Ab ham within constructor ek mthod define kr skty hen

In React class components, the setState() method is typically **not** called directly inside the render() method. This is because calling setState() triggers a re-render, and if you call it inside render(), it can lead to an infinite loop as each state update will cause another re-render.

Instead, setState() should be used in lifecycle methods like componentDidMount(), componentDidUpdate(), or event handlers. However, if it's absolutely necessary to update state during the rendering process, you can conditionally check and prevent unnecessary updates, but it's generally not a good practice.

****

React Components

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML via a render() function.

Components come in two types, Class components and Function components, in this chapter you will learn about Class components.

Create a Class Component

When creating a React component, the component's name must start with an upper case letter.

The component has to include the `extends React.Component` statement, this statement creates an inheritance to React.Component, and gives your component access to React.Component's functions.

The component also requires a `render()` method, this method returns HTML.

Example

Create a Class component called `Car`

```
class Car extends React.Component {
  render() {
    return <h2>Hi, I am a Car!</h2>;
  }
}
```

Now your React application has a component called Car, which returns a `<h2>` element.

To use this component in your application, use similar syntax as normal HTML: `<Car />`

Example

Display the `Car` component in the "root" element:

```
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<Car />);
```

Run Example »

Component Constructor

If there is a `constructor()` function in your component, this function will be called when the component gets initiated.

The constructor function is where you initiate the component's properties.

In React, component properties should be kept in an object called `state`.

You will learn more about `state` later in this tutorial.

The constructor function is also where you honor the inheritance of the parent component by including the `super()` statement, which executes the parent component's constructor function, and your component has access to all the functions of the parent component (`React.Component`).

Example

Create a constructor function in the Car component, and add a color property:

```
class Car extends React.Component {
  constructor() {
    super();
    this.state = {color: "red"};
  }
  render() {
    return <h2>I am a Car!</h2>;
  }
}
```