
Let's get
EP:8
CLassy

Class based Components

from react lib

```
class componentName extends React.Component {
  render() {
    return (
      == some jsx.
    )
  }
}
```

a function that returns some jsx.

you have constructors to get props.

```
constructor (props) {
    super(props)
}
```

→ to use these props you will use a this keyword.

---

when you create an instance of a class you are loading the class based component on the web page.

constructor is best place to receive props.

Accessing props inside class based component

* use this.props.<property Name>.
* for destructuring you do.

```
{ name, location, handle } = this.props;
```

Making State variables

the hooks are new concept, in class based component we follow older method.

we write,

```
this.state {
    count : 0;
    count2 : 0
}
```

just keep adding more state variables.

this is a whole big object, which holds all state variables

using the state variable,

count = {this.state.count}

Updating state variables

never update state variable DIRECTLY.

✗ `this.state.count = this.state.count + 1;`

how to correctly update?

→ this.setState()

↙ pass an object which contains the updated value of the state variable.

```
this.setState(
{count: this.state.count + 1,});
```

similar to the this.state, you can update multiple state variables in the same setState object

if you have many state variables in this.state, and you are updating only a few in this.setState; ONLY the vars in this.setState, will be updated.

When class based component is rendered, the constructor is called 1st, then render method is called.

Parent constructor
Parent render

Child constructor
Child render

render cycle in react

componentDidMount() ←
called after render()

Parent did mount called only when its completely mounted, that means whole child component should be mounted, and only then, at last. parent did mount is called.

multiple children ?

Parent constructor

Parent render

child 1 constructor
child 1 render
] render phase 1st child

child 1 constructor
child 1 render
] render phase 2nd child.

child 1 did mount

child 2 did mount

Parent did Mount.

Phases of life cycle.

render phase

- constructor
- render

commit phase

- update the DOM
- componentDidMount

react optimises :

- render phase of all children — batched
- commit phase of all children — batched

- this is optimised because the dom manipulation is VERY expensive.
- in render phase, we have the reconciliation cycle, → very fast.
- commit phase is slow

Use of ComponentDidMount

→ making API calls.

Why? It is loaded after whole component is mounted.

We want to QUICKLY render the component → then make API call → then fill in the data and rerender the component.

how?

componentdid update

called after the render when actual data recieved.

component will unmount

when you go to another page, this method is called.

MOUNTING

Constructor (dummy data)

Render (dummy data

< HTML dummy data >

component did mount.

< API call > actual data

< this. set State > → state variable updated with actual data.

UPDATE

render (actual data)

< HTML actual data)

component did update

called after the render cycle

if actual data.