Conditional Rendering

* Conditional Rendering in React JS

We can create multiple components which encapsulate behaviour that we need. After that, we can render them depending on some conditions or the state of our application.  There is more than one way to do conditional rendering in React. They are given below.

* if
* ternary operator
* logical && operator
* switch case operator
* Conditional Rendering with enums
* Logical && Operator

It works because in JavaScript, true && expression always evaluates to expression , and false && expression always evaluates to false . Therefore, if the condition is true , the element right after && will appear in the output. If it is false , React will ignore and skip it.

**Example:**

render() {

const count = 0; return (

<div>

{count && <h1>Messages: {count}</h1>} </div>

);

}

* Switch case operator

You can use a switch case operator to handle the conditional rendering of these multiple states.

**Example:**

function Notification({ text, state }) {

switch(state) {

case 'info':

return <Info text={text} />

case 'warning':

return <Warning text={text} />

case 'error':

return <Error text={text} />

default:

return null

}

}

* Conditional Rendering with enums

 If there are more than 3 components available to render conditionally, if-else becomes complex. So, developers should use enums to keep code clean.

In this file, we will add some basic React code to render on the webpage. The user needs to add the following code to the ‘first.js’ file.

**Filename: first.js**

|  |
| --- |
| import React, { Component } from 'react';  // Some basic code to render first component  class First extends Component {    render() {      return (        <div>          <h2>This is a first component</h2>        </div>      );    }  }  export default First; |

In this file, we will add some basic React code that is different from the first component. So, we can know which component is rendering on the screen. Edit the ‘second.js’ file and add the below code inside it.

**Filename: second.js**

|  |
| --- |
| import React, { Component } from 'react';  // some basic code to render second component  class Second extends Component {    render() {      return (        <div>          <h2>This is a second component.</h2>        </div>      );    }  }  export default Second; |

**Rendering component using enum**

**Step 1:** In javascript, we can create an object with key-value pairs and use it as an enum. Below, you can see the demo of a javascript object with key-value pair.

**Syntax:**

const Enumobj = {

key: value,

};

Example:

const Enumobj = {

first: <First />,

second: <Second />

};

**Step 2:** Now, we will make a javascript function that takes a state as a parameter and return a React component based on the state.

**Syntax:**

function Enum({state}){

return {object[state]};

}

**Example:**

function Enum({ state }) {

return <div>{Enumobj[state]}</div>;

}

**Step 3:**Let’s embed the ‘Enum‘ function in our ‘App‘ component. While calling the ‘Enum‘ function, we will add state value as props.

**Syntax:**

return (

<div>

<Enum state="Value"></Enum>

</div>

);

**Example:**

return (

<div>

<Enum state="first"></Enum>

<Enum state="second"></Enum>

</div>

);

**Filename: App.js**

In the App.js file, we will create an enum object first. After that, we will add an ‘enum’ function to render components according to state value. At last, we will edit the ‘App‘ component and call the ‘enum‘ function inside the component to render it conditionally. The user needs to add the below code to the ‘App.js‘ file.

**Filename: App.js**

|  |
| --- |
| import React, { Component } from 'react';  import Second from './components/second'  import First from './components/first'  // Creating enum object  const Enumobj = {    first: <First />,    second: <Second />  };  // Creating enum function to return  // Particular component according to state value  function Enum({ state }) {    return <div>{Enumobj[state]}</div>;  }  // Call enum function inside the App component  class App extends Component {    render() {      return (        <div>          <Enum state="first"></Enum>          <Enum state="second"></Enum>        </div>      );    }  }  export default App; |

**Steps to run:**The user needs to run beneath the command to the terminal in the current directory to see the output.

npm start

**Output:**

This is a first component.

This is a second component.