**1**

If you want to print something on an HTML page using Core React and JavaScript, first, you need to create an HTML file. In that file, you should have a <div> tag with an id of your choice. For example, here I have given it the id "root."

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<body>

<div id="root"></div>

</body>

</html>

Now, since we are going to use the raw React library, we need to import the React library's script file.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<body>

<div id="root"></div>

<script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>

<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>

<script src="test.js"></script>

</body>

</html>

Here, I have created a script file named "test.js" where we will write all our React code. Since we are using the React library to render content inside the "root" div as we wish, we need to first select the "root" div in the "test.js" script file.,

let container = document.querySelector('#root') ;

Now, we will create a React element. The function for creating a React element looks like this:

let myElement = React.createElement('div', null, 'Hello World') ;

In the `React.createElement` function, the first parameter specifies the type of element we want to create. Here, we want to create a `Div` type element. Then, we can set the second parameter to `null` because currently, we are not setting any attributes or properties.

Next, in the third parameter, we specify the content that we want to print inside the element.

Now, we will use React DOM to render this element inside the "root" div.

let container = document.querySelector('#root') ;

let myElement = React.createElement('div', null, 'Hello World') ;

ReactDOM.render(myElement, container) ;

This is how the code will look if we want to print something using the <p> tag instead of "Hello World":

let container = document.querySelector('#root') ;

let myElement = React.createElement('div', null, React.createElement('p', null, 'Hello world from Paragraph')) ;

ReactDOM.render(myElement, container) ;

If you want to print multiple items using multiple <p> tags within the "root" div, you can create an array of React elements as the third parameter in React.createElement. Here's how you can do it:

let container = document.querySelector('#root') ;

let myElement = React.createElement('div', null,[

React.createElement('p', null, 'Hello world from Paragraph'),

React.createElement('p', null, 'Hello world from Paragraph 2'),

React.createElement('p', null, 'Hello world from Paragraph 3'),

]) ;

ReactDOM.render(myElement, container) ;

This way, using React, we can create elements and use all kinds of HTML tags to display anything on an HTML page.

English Title : How to show simple 'Hello World ' with Core React in a html page ?

Bangla Title : রিয়েক্ট লাইব্রেরী দিয়ে একটি html পেইজে কিভাবে একটি সিম্পল Hello World লেখা প্যারাগ্রাফ প্রিন্ট করতে পারি।

**2**

import React from 'react';

import ReactDOM from 'react-dom'

function Test({user}){

  return (

    <h2>Hello Testing {user}</h2>

  )

}

ReactDOM.render(<Test user="russell" />, document.getElementById('root'));

**3**

import React from 'react';

import ReactDOM from 'react-dom'

class Test{

  print(){

    return(

      <h2>Hello World </h2>

    )

  }

}

const Testcomponent = new Test();

ReactDOM.render(Testcomponent.print(), document.getElementById('root'));

**4**

React দিয়ে স্ক্রীনে একটি hello world প্রিন্ট করতে হলে আমাদেরকে প্রথমে একটি JSX ইলেমেন্ট বানাতে হবে এরকম করেঃ

const dummy = (

  <h2 className="storng-text">Hello World</h2>

)

এটা মূলত ব্যাবেল দিয়ে ট্রান্সপাইল হয়ে অবজেক্ট আকারে জাভাস্ক্রিপ্টের কাছে যায় এবং স্ক্রীনে প্রিন্ট করেঃ

const dummy = {

  type : 'h2',

  props : {

    className : 'strong-text',

    children : 'Hello World'

  }

};

এই অবজেক্টটাই হলো রিয়েক্ট *ইলেমেন্ট।*

**5**

Doing a repetitive task on a website can involve designing a specific div or a frontend functionality. To accomplish this using raw JavaScript, you would need to write functions repeatedly, redesign the elements repeatedly, and manipulate those elements using the DOM repeatedly. For larger websites or applications, this can be quite cumbersome. To avoid this repetition and make it more efficient, React was born.

For example, let's say we want to create a small counter application where pressing a button increments a number by 1.

<https://i.ibb.co/820F5KX/Xnip2023-09-21-21-15-50.jpg>

Now, to do this in plain JavaScript, you first need to create an HTML document with the necessary HTML structure.

<https://i.ibb.co/4TKMNqX/code.png>

Then, you need to write CSS for it, followed by writing the necessary JavaScript code in a script file for its functionality.

<https://i.ibb.co/JFHNf5b/zsdf.jpg>

Here, first, we need to select the button and the h tag through which we will display the number using the DOM. Then, we'll declare a variable. After that, we need to add a click event listener to the button. When the button is clicked, we'll increment the value of a variable called 'number' by one, and then manipulate the DOM to display that value in the 'text' h2 tag.

If you need to repeat this task several times, you will have to write new HTML elements, write new CSS, handle the HTML elements using the DOM, and write the script separately each time, as shown in the image below.

<https://i.ibb.co/qCHPX1q/Xnip2023-09-21-22-14-35.jpg>

html : <https://i.ibb.co/jZWYJ31/4.jpg>

js : <https://i.ibb.co/wQBb4Q1/5.jpg>

This is a repetitive task. React has come to the rescue for such problems. Thanks to React, you can write HTML, CSS, and JavaScript just once and use them as many times as you want for the same task without writing new code every time. If we had written the above code in React,

Html : <https://i.ibb.co/mJFjqNn/6.jpg>

Js : <https://i.ibb.co/1b1cJc3/7.jpg>

ReactDOM.render(<>

<Element />

<Element />

<Element />

</>, main)

Using the ReactDOM.render() function, inside the element with the ID "main," we can use the same <div> element as many times as we want. This means that by creating the element only once, we can reuse it multiple times.

**6**

Core React দিয়ে একটি স্ক্রিণে একটি hello world প্রিন্ট করার