React Element :

- A React element is a plain JavaScript object that describes or creates a UI component in React.
- A React element is like a description of what should appear on the screen.

How to create a React Element ?

Using React.createElement() function.

What is React.createElement() ?

React.createElement() is a predefined function in React that allows you to create and describe a React element.

In React, every visible thing on the screen — like a button, a heading, or a text box — is called an element. React.createElement is a function that lets you create these elements.

Think of React.createElement as a way to tell React, "Please create this part of my user interface."

How Does `React.createElement` Work?

The function takes three main parts:

```
React.createElement(type, props, children);
```

Let's break down each part in an easy-to-understand way.

1.type: What kind of element do you want?

This could be any HTML tag, like `"div"`, `"button"`, or `"h1"`. For example, if you want to create a button, you would set the `type` as `"button"`.

2.props: What properties (or attributes) do you want to give this element?

props is an object where you can specify things like:
- `className` to give the element a CSS class,
- `id` for a unique identifier,
- `onClick` to define what happens when you click it, etc.

If you don't need any special properties, you can just pass `null` here.

3.children: What goes inside this element?

This is the content that will appear inside the element. It can be:

- Text (like the button label),
- single element (like a heading inside a div),
- Even a list of multiple elements.

Simple Example: Creating a Button

Let's say you want to make a simple button that says "Click Me".

```
const buttonElement = React.createElement(
  "button",
  { className: "my-btn" },
  "Click Me"         );
```

This tells React to create:

```
<button class="my-btn">Click Me</button>
```

Example with Nested Elements

If you want to create a `div` with a title(h1) and a paragraph(p) inside it, you can nest elements by using `React.createElement` multiple times.

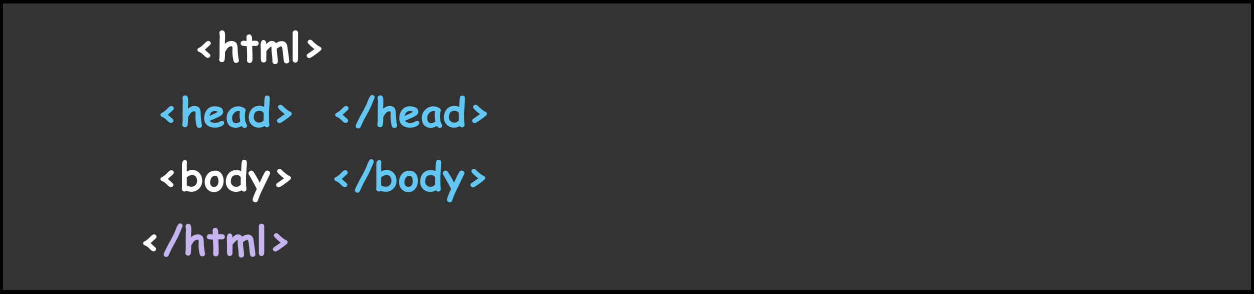
```
const nestedElement = React.createElement(
  "div",
  null,
  [React.createElement("h1", null, "Welcome!"),
React.createElement("p", null, "This is a paragraph")]
);
```

This structure would look like this in HTML:

```
<div>
  <h1>Welcome!</h1>
  <p>This is a paragraph.</p>
</div>
```

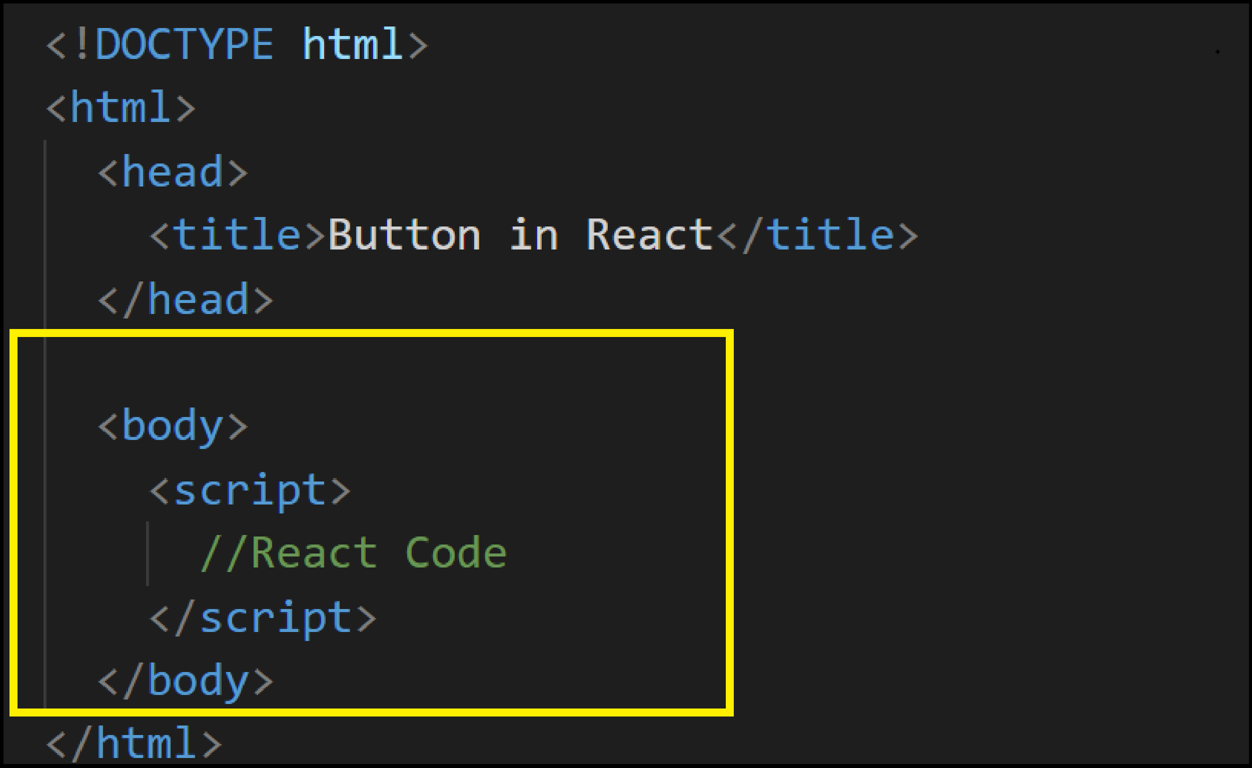
Steps to create React Element :

1. Create an HTML File and Add Basic HTML Code



```
     <html>
   <head>  </head>
   <body>  </body>
  </html>
```

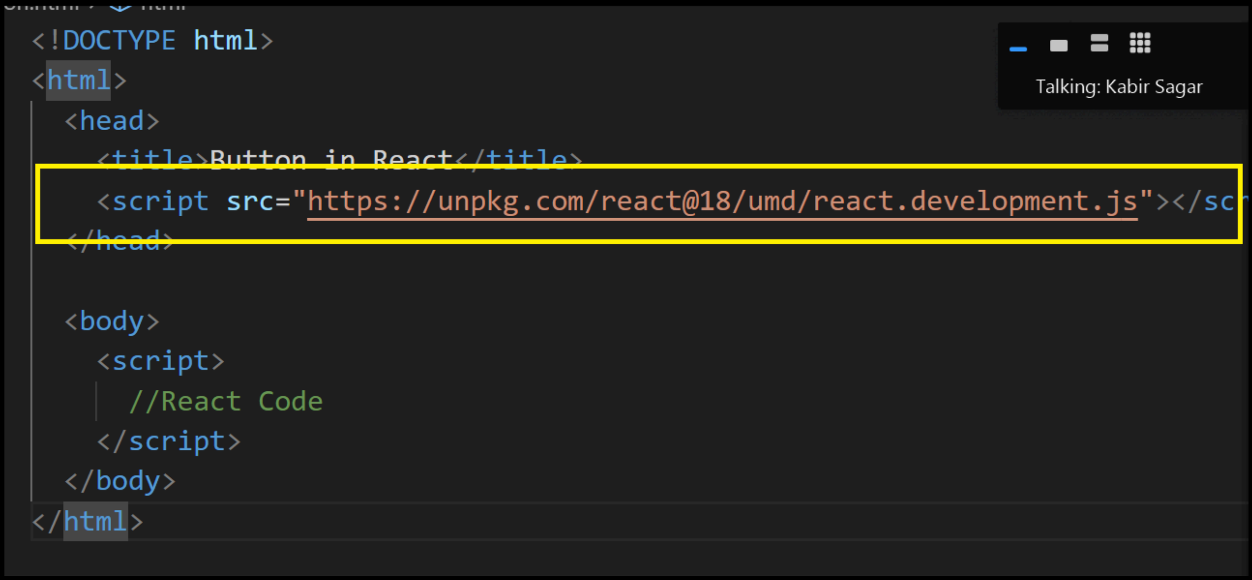
2. Add Script Tag Inside the Body Tag



```
<!DOCTYPE html>
<html>
  <head>
    <title>Button in React</title>
  </head>

  <body>
    <script>
      //React Code
    </script>
  </body>
</html>
```

3.Integrate React Library with HTML Document



```html
<!DOCTYPE html>
<html>
  <head>
    <title>Button in React</title>
    <script src="https://unpkg.com/react@18/umd/react.development.js"></sc
  </head>

  <body>
    <script>
      //React Code
    </script>
  </body>
</html>
```

4.Create React Element (Button) using `React.createElement()`

- Example:

```
var buttonElement = React.createElement(
    "button",
    null,
    "Login Button in React"
  );
```

5.Add React Element in DOM using `ReactDOM` Library

- First, integrate `ReactDOM` with the HTML document.

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

- Use `ReactDOM.render()` to add the element to the DOM:

```
ReactDOM.render(buttonElement,document.querySelector("<DOM_Element>"))
```

6.Load HTML File in the Browser

- Open the HTML file in a browser to see the output.

ReactDOM :

ReactDOM is a library that is responsible for rendering(loading) React elements into the DOM (Document Object Model) of a web page.

What Does ReactDOM Do?

ReactDOM mainly does two things:

1. Puts React Elements on the Web Page: It takes the components you've built in React and makes them visible on your web page.
2. Updates Only What Changes: When something in your app changes, ReactDOM updates only the part of the page that needs to be changed instead of reloading everything. This makes your app faster and more efficient.

ReactDOM.render()

- This method was widely used to render a React element into a DOM node.

```
ReactDOM.render(<App>,document.getElementById('root'))
```

- The render() function takes two arguments:
    - React Element: The component (e.g., <App />) you want to render.
    - DOM Node: The DOM node where the component should be rendered, usually a <div id="root"> element.

However, starting from React 18, ReactDOM.render() has been replaced by ReactDOM.createRoot() for rendering applications.

ReactDOM.createRoot()

- createRoot() is a new API introduced in React 18, used for creating a root element for React's new concurrent rendering capabilities.

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

- This API enables concurrent rendering, which optimizes how updates and re-renders happen in the application by allowing React to pause and resume rendering as needed for performance.

Interview Questions

1. What is React element and purpose of it ?
2. What are the arguments for React.createElement()?
3. Explain the role of each argument in React.createElement(type, props, ...children)
4. What is returned when you use React.createElement()
5. How would you use React.createElement() to create an element with multiple child elements?.

Practice Questions 👍

1. How do you create a React Element without JSX? Write an example to create a simple div element with a className of "container" and text content "Hello, World!".
2. Create h1 to h6 element with React in each different file.
3. Create Paragraph element in React