ARYATECHNO

React - HTML

Topics : React JS
Written on January 01, 2024

In React, instead of writing traditional HTML directly in your JavaScript files, you use JSX (JavaScript XML) to describe what the UI should look like. JSX is a syntax extension for JavaScript recommended by React, and it allows you to write HTML-like code within your JavaScript files. JSX gets transpiled into JavaScript by tools like Babel before it is executed in the browser.

Here's a quick overview of how JSX works in React:

1. **Basic JSX:** You can use JSX to describe the structure of your components. JSX elements look similar to HTML tags, but they are actually JavaScript expressions.

   ```
   const element = <h1>Hello, JSX!</h1>;
   ```

2. **Embedding Expressions:** You can embed JavaScript expressions inside JSX using curly braces `{}`. This allows you to dynamically include values or expressions within your JSX.

   ```
   const name = 'John'; const element = <h1>Hello, {name}!</h1>;
   ```

3. **Attributes in JSX:** JSX supports HTML-like attributes for elements. These attributes can also include dynamic values using curly braces.

   ```
   const imageUrl = 'https://example.com/image.jpg';
   const element = <img src={imageUrl} alt="An example image" />;
   ```

4. **JSX Within Components:** When defining React components, you use JSX to describe their structure. Components can be either functional or class-based.

   ```
   // Functional Component
   const MyComponent = () => {
     return <div>Hello from MyComponent!</div>;
   };

   // Class Component
   class MyComponentClass extends React.Component {
     render() {
       return <div>Hello from MyComponentClass!</div>;
     }
   }
   ```

5. **Using JSX in Render Method:** When writing class components, the `render` method is where you return JSX to describe what should be rendered.

```
class Greeting extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}!</h1>;
  }
}
```

6. **Conditional Rendering:** JSX allows you to use JavaScript expressions for conditional rendering. This can be achieved using the ternary operator or logical && operator.

```
const isLoggedIn = true;

const element = (
  <div>
    {isLoggedIn ? <p>Welcome back!</p> : <p>Please log in.</p>}
  </div>
);
```