React JSX

- JSX stands for JavaScript XML.
- JSX, or JavaScript XML, is a syntax extension for JavaScript often used with React.
- It allows you to write HTML elements and components in a syntax that looks similar to XML or HTML within your JavaScript code.
- JSX allows us to write HTML elements in JavaScript and place them in the DOM.
- JSX makes it easier and more instinctive to describe the structure of React components.
- JSX is an extension of the JavaScript language based on ES6, and is translated into regular JavaScript at runtime.
- JSX converts HTML tags into react elements.

JSX Syntax/JSX for React Components

```jsx
import React from 'react';

const MyComponent = () => {
  return (
    <div>
      <h1>Hello, JSX!</h1>
      <p>This is a React component using JSX.</p>
    </div>
  );
};

export default MyComponent;
```

In the above example:

- The **return** statement contains JSX code.
- HTML-like tags (** **, **<h1>**, **<p>**) are used to define the structure of the component.
- JavaScript expressions can be embedded within curly braces (**{}**) to include dynamic content.

JSX gets transformed into JavaScript code before it's rendered by the browser. The example above would be transformed into something like the following using React.createElement:

```jsx
const MyComponent = () => {
  return React.createElement('div', null,
    React.createElement('h1', null, 'Hello, JSX!'),
    React.createElement('p', null, 'This is a React component using JSX
  );
};
```

JSX Syntax/JSX for React Components

```
class App extends Component {
  render() {
    var myStyle = { fontSize: 100, color: '#FF0000' }
    return (
      <div>
        <h1 style={myStyle}>Hello World</h1>
        {1+1}
        {/* Write Comment Here */}
      </div>
    );
  }
}
```

JSX Attributes

- JSX tags have a tag name, attributes and children.
- If an attribute value is enclosed in quotes then the value is string.
- If we wrap the value in curly braces then the value is the enclosed JavaScript expression

```
return (
  <div>
    <div className="red">Children Text</div>;
    <MyCounter count={3 + 5} />;
  </div>
);
```

JSX Expressions

- JSX expressions needs to wrap within curly brackets { }.
- JSX expressions doesn't encapsulate conditional constructs.
- We can use ternary(conditional) expressions for making decisions.

```
var num = 5;
return (
  <div>
    <h1>Hello World</h1>
    <h1>{num > 0 ? "Positive" : "Not Positive"}</h1>
  </div>
);
```

JSX Styling

- React allows inline styles.
- While using inline styles, We need to use camelCase syntax.

```
<div>
  <h1 style={myStyle}>Hello World</h1>
  {1+1}
  {/* Write Comment Here */}
</div>
```

Key points about JSX

- **Readability:** JSX makes the code more readable and resembles the HTML structure of the rendered components.
- **Expressiveness:** JSX allows you to express the intended UI structure and hierarchy more naturally.
- **JavaScript Expressions:** You can embed JavaScript expressions within curly braces '{ }' to include dynamic content, variables, or execute functions.
- **Babel Transformation:** JSX is not directly understood by browsers, so it needs to be transformed into JavaScript using tools like Babel as part of the build process.

Embedding HTML in JavaScript:

- JSX allows developers to embed HTML-like code directly within JavaScript files. This makes it easier to visualize and write the structure of UI components.

jsx Copy code

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

Key points about JSX

JSX Expressions:

- JSX expressions are like JavaScript expressions and can be used inside curly braces `{}`. This allows dynamic values and JavaScript logic to be incorporated into JSX.

```jsx
const name = "World";
const element = <h1>Hello, {name}!</h1>;
```

Attributes in JSX:

- JSX attributes are similar to HTML attributes and can be used to pass data to components.

```jsx
const link = <a href="https://www.example.com">Visit Example</a>;
```

Class vs. className:

- JSX uses `className` instead of `class` to define the CSS class for an element. This is because `class` is a reserved keyword in JavaScript.

```jsx
const element = <div className="my-container">Content goes here</div>;
```

Key points about JSX

- **JSX is not Required:**
- While JSX is a common and recommended syntax for writing React components, it is not mandatory. React can also be written using pure JavaScript without JSX. However, JSX is widely adopted due to its readability and expressiveness.
- Using JSX in React is a convenient way to write UI components, and it is a core part of the React development experience. The Babel transpiler is commonly used to convert JSX code into JavaScript that browsers can understand.