💡 What is JSX?

JSX (JavaScript XML) is a **syntax extension for JavaScript** that lets you write HTML **inside JavaScript**. It's used in React to describe the **UI** structure in a more readable and declarative way.

🧩 Example:

jsx Copy Edit

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

Instead of writing this in pure JS:

javascript Copy Edit

```javascript
const element = React.createElement('h1', null, 'Hello, JSX!');
```

JSX just makes it simpler and cleaner ✨

🧠 Why Use JSX?

- ✅ Looks like HTML
- ✅ Easier to read and write
- ✅ Helps visualize UI directly in JavaScript
- ✅ Works with the full power of JavaScript

📌 Key JSX Rules

1. Must return a single parent element

✅ Valid:

jsx Copy Edit

```jsx
return ( <div> <h1>Hello</h1> <p>Welcome</p> </div> );
```

❌ Invalid:

jsx Copy Edit

```jsx
return ( <h1>Hello</h1> <p>Welcome</p> );
```

Or use a **fragment** (`<> </>`) to group elements without adding an extra `<div>` .

jsx Copy Edit

```jsx
return ( <> <h1>Hello</h1> <p>Welcome</p> </> );
```

2. Use `className` instead of `class`

jsx Copy Edit

```jsx
// Correct <div className="container"></div>
```

> This is because `class` is a reserved keyword in JavaScript.

3. Use curly braces `{}` for JS expressions

You can embed any JavaScript expression inside JSX using `{}` .

jsx Copy Edit

```jsx
const name = "React"; return <h1>Hello, {name}!</h1>; // Output: Hello, React!
```

4. Self-closing tags are required for empty elements

jsx Copy Edit

```jsx
<img src="logo.png" alt="Logo" /> <br />
```

🛠️ JSX in Functional Component Example:

jsx Copy Edit

```jsx
function Welcome(props) { return <h1>Hello, {props.name}!</h1>; }
```

Called like:

jsx Copy Edit

```jsx
<Welcome name="Alice" />
```

🔥 Summary

Concept	Example	Description
JSX	`<h1>Hello</h1>`	Write HTML in JS
Expressions	`{name}`	Insert dynamic values
className	`className="box"`	Use instead of `class`
Fragments	`<> </>`	Return multiple elements
Self-closing	`<img />`	Must close empty tags