**1. JSX (JavaScript XML)**

**Definition:**  
JSX is a syntax extension for JavaScript. It looks like HTML inside JavaScript. React uses JSX to describe the **UI structure** in a declarative way.

Example:

const element = <h1>Hello, React!</h1>;

**Why JSX?**

1. **Declarative Syntax:** Easier to read and write UI code.
2. **Compile-time Optimization:** JSX is compiled to React.createElement() calls.
3. **Dynamic UI:** You can embed JavaScript expressions inside JSX.

**JSX Basics**

1. **Embedding JavaScript in JSX**

Use curly braces {}:

const name = "Veda";

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

1. **JSX is an Expression**

You can use JSX in variables, functions, conditionals:

const isLoggedIn = true;

const element = <h1>{isLoggedIn ? "Welcome!" : "Please login"}</h1>;

1. **JSX Tags must have one parent**

You cannot return multiple sibling elements without a wrapper:

// ❌ Invalid

return (

<h1>Hello</h1>

<p>World</p>

);

// ✅ Valid

return (

<div>

<h1>Hello</h1>

<p>World</p>

</div>

);

// ✅ Also valid using Fragment

return (

<>

<h1>Hello</h1>

<p>World</p>

</>

);

1. **Attributes in JSX**

* JSX attributes are camelCase:

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

<button onClick={handleClick}>Click Me</button>

1. **JSX is not HTML**

* class → className
* for → htmlFor
* Inline styles: object syntax

<div style={{ color: "red", fontSize: "20px" }}>Hello</div>

1. **JSX Compiles to React.createElement()**

const element = <h1>Hello</h1>;

is compiled to:

const element = React.createElement("h1", null, "Hello");

**Advanced JSX Features**

1. **Conditional Rendering**

{isLoggedIn && <h1>Welcome Back!</h1>}

1. **Rendering Lists**

const numbers = [1, 2, 3];

const listItems = numbers.map((n) => <li key={n}>{n}</li>);

return <ul>{listItems}</ul>;

1. **JSX Spread Attributes**

const props = { id: "myDiv", className: "container" };

<div {...props}>Hello</div>;