JSX (JavaScript XML)

Question 1: What is JSX in React.js? Why is it used?

Ans.

JSX (JavaScript XML) is a syntax extension for JavaScript used in React. It allows you to write HTML-like code inside JavaScript, making it easier to describe what the UI should look like.

Example:

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

Under the hood, JSX is **converted (compiled)** into regular JavaScript using React.createElement() :

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

Why is JSX used ?

JSX is used because it makes React code simpler, cleaner, and more readable.

1. Improves Readability

JSX looks like HTML, so developers can easily understand the UI structure.

```
return (
  <div>
    <h1>Welcome</h1>
```

```
  <p>This is React</p>
 </div>
);
```

2. Combines UI and Logic :

JSX allows JavaScript expressions inside {}.

```
const name = "Uday";
<h1>Hello, {name}</h1>
```

3. Better Developer Experience :

- Less code
- Easier debugging
- Clear component structure

4. Prevents Injection Attacks :

JSX automatically escapes values, helping protect against XSS (Cross-Site Scripting).

5. Supports Component-Based Architecture :

JSX works naturally with reusable React components.

Ex. <MyComponent />

Question 2: How is JSX different from regular JavaScript? Can you write JavaScript insideJSX ?

Ans.

How is JSX different from regular JavaScript ? :

Aspect	JSX	Regular JavaScript
Syntax	HTML-like ( )	JS functions & objects
Purpose	Describe UI structure	Logic, calculations
Browser support	Needs transpiling	Runs directly
Compilation	Converted to React.createElement()	No conversion needed

Example comparison :

JSX :-

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

Regular JavaScript :-

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

Can you write JavaScript inside JSX ? :

Yes, you can write JavaScript expressions inside JSX using curly braces {}.

Example :

1. Variables :-

```
const name = "Uday";
<h1>Hello, {name}</h1>
```

2. Expressions :

```
<h2>{10 + 5}</h2>
```

3. Function calls :

```
function greet() {
  return "Good Morning";
}
<p>{greet()}</p>
```

4. Conditional rendering :

{isLoggedIn ? <Dashboard /> : <Login />}

5. Array rendering :

{items.map(item => <li key={item.id}>{item.name})}

What you CANNOT write inside JSX :-

You cannot write statements inside JSX, such as:

- if
- for
- while
- switch

Invalid:

{ if (x > 5) return "Yes"; }

Valid (using expression) :

{x > 5 && "Yes"}

Question 3: Discuss the importance of using curly braces {} in JSX expressions.

ANS.

Importance of Using Curly Braces {} in JSX Expressions

In JSX, curly braces {} are used to embed JavaScript expressions inside JSX. They act as a bridge between JavaScript logic and UI markup.

Why {} are important in JSX

1. Embed JavaScript Values

Curly braces allow you to display variables directly in the UI.

```
const name = "Uday";
<h1>Hello, {name}</h1>
```

2. Evaluate Expressions

You can perform calculations or expressions inside {}.

```
<p>Total: {price * quantity}</p>
```

3. Call Functions

Functions can be executed inside JSX using {}.

```
function greet() {
  return "Welcome!";
}
<h2>{greet()}</h2>
```

4. Conditional Rendering

Curly braces enable conditions inside JSX.

```
{isLoggedIn && <Dashboard />}
{isAdmin ? <Admin /> : <User />}
```

5. Render Lists Dynamically

Used to render arrays using .map().

```
{users.map(user => (
  <li key={user.id}>{user.name}</li>
))}
```

<u>What {} can and cannot contain :</u>

Allowed (Expressions) :

- Variables
- Arithmetic operations
- Ternary operators
- Function calls
- JSX elements

Not Allowed (Statements) :

- if, for, while, switch
- Variable declarations (let, const)

Invalid :

{ if (x > 10) return "High"; }

Valid :

{x > 10 ? "High" : "Low"}
