React Lecture - 3

Curly Braces in JSX

Javascript in JSX

JavaScript in JSX

Till now we have learned how to write HTML markup in JSX

Today we will learn how can we write Javascript in JSX

JSX lets you write HTML-like markup inside a JavaScript file, keeping rendering logic and content in the same place. Sometimes you will want to add a little JavaScript logic or reference a dynamic property inside that markup. In this situation, **you can use curly braces in your JSX to open a window to JavaScript**.

Javascript in JSX

Suppose there is a component which renders a image

```
1  export default function Avatar() {
2    return (
3      <img
4        className="avatar"
5        src="https://i.imgur.com/7vQD0fPs.jpg"
6        alt="Gregorio Y. Zara"
7      />
8    );
9  }
```

Javascript in JSX

Now if you want to render that image dynamically using some variable. You can use curly braces.

```
1  export default function Avatar() {
2    const avatar = 'https://i.imgur.com/7vQD0fPs.j
3    const description = 'Gregorio Y. Zara';
4    return (
5      <img
6        className="avatar"
7        src={avatar}
8        alt={description}
9      />
10   );
11 }
```

Where can we use curly braces ?

You can only use curly braces in two ways inside JSX:

1. As text directly inside a JSX tag: <h1>{name}'s To Do List</h1> works, but <{tag}>Gregorio Y. Zara's To Do List</{tag}> will not.

2. As attributes immediately following the = sign: src={avatar} will read the avatar variable, but src="{avatar}" will pass the string "{avatar}".

Using double curly braces

In addition to strings, numbers, and other JavaScript expressions, you can even pass objects in JSX. Objects are also denoted with curly braces, like { name: "Hedy Lamarr", inventions: 5 }. Therefore, to pass a JS object in JSX, you must wrap the object in another pair of curly braces: person={{ name: "Hedy Lamarr", inventions: 5 }}.

The next time you see {{ and }} in JSX, know that it's nothing more than an object inside the JSX curlies!

Using double curly braces

You may see this with inline CSS styles in JSX. React does not require you to use inline styles (CSS classes work great for most cases). But when you need an inline style, you pass an object to the style attribute:

```
export default function TodoList() {
  return (
    <ul style={{
      backgroundColor: 'black',
      color: 'pink'
    }}>
      <li>Improve the videophone</li>
      <li>Prepare aeronautics lectures</li>
      <li>Work on the alcohol-fuelled engine</li
    </ul>
  );
}
```

What's allowed inside curly braces

- **Function calls**: **Allowed**. You can call functions inside curly braces, and the returned value will be rendered. Example:

```
function greet() {
  return "Hello!";
}
<h1>{greet()}</h1>
```

- **Arrays**: **Allowed.** Arrays are often used in JSX for rendering lists. If an array contains valid JSX elements or strings, they will be rendered. Example:

```
const items = ["Item 1", "Item 2"];
<div>{items}</div> // Renders: Item 1Item 2
```

What's allowed inside curly braces

- **JavaScript comments**: **Not allowed**. You cannot place comments inside curly braces because JSX expects a valid JavaScript expression. To add comments in JSX, use the following syntax outside curly braces:

```
{/* This is a valid JSX comment */}
```

- **Conditional (ternary) operators**: **Allowed**. Ternary operators are valid JavaScript expressions and work well in JSX for conditional rendering.

```
const isLoggedIn = true;
<p>{isLoggedIn ? "Welcome back!" : "Please log in."}</p>
```