# Mastering JSX

## Table of Contents:



## Introduction to JSX

```
const tag = <h1>Hello</h1>

// gets converted to

const tag = React.createElement("h1", {}, "Hello")
```

- JSX is syntactic sugar used to create very specific JavaScript objects. JSX stands for JavaScript XML.

## React is Declarative!

- You define *what* the UI should look like based on state, and React takes care of updating it efficiently.
- **Imperative programming**: You explicitly define *how* to do something, step by step.

  ```
  const h1 = document.createElement("h1");
  h1.innerText = "Hello World";
  document.body.appendChild(h1);
  ```

- **Declarative programming**: You describe *what* you want, and the system figures out how to do it. React uses a Declarative programming approach.

  ```
  <h1>Hello World</h1>
  ```

## JSX Syntax

### ▼ Use parenthesis to insert a large block of JSX.

```
const myElement = (
  <ul>
    <li>Apples</li>
    <li>Bananas</li>
    <li>Cherries</li>
  </ul>
);
```

### ▼ One Top Level Element

The HTML code must be wrapped in *ONE* top-level element.

So if you like to write *two* paragraphs, you must put them inside a parent element, like a `div` element. Or you can also use Fragment or empty tag `<>`

### ▼ Expressions in JSX {}

With JSX you can write expressions inside curly braces `{ }`.

The expression can be a React variable, or property, or any other valid JavaScript expression. JSX will execute the expression and return the result:

```
const myElement = <h1>React is {5 + 5} times better with J
```

### ▼ The class attribute

The `class` attribute is a much-used attribute in HTML, but since JSX is rendered as JavaScript, and the `class` keyword is a reserved word in JavaScript, you are not allowed to use it in JSX.

Use attribute `className` instead.

JSX solved this by using `className` instead. When JSX is rendered, it translates `className` attributes into `class` attributes.

```
const myElement = <h1 className="myclass">Hello World</h1>
```

▼ **Conditions - if statements**

React supports `if` statements, but not *inside* JSX. Because the if statements are not *expressions* in Javascript. You could use a ternary expression instead:

```
const x = 5;

const myElement = <h1>{(x) < 10 ? "Hello" : "Goodbye"}</
h1>;
```

▼ **Using double curlies `{{}}`**

- To pass objects in react:

```
person={{ name: "Hedy Lamarr", inventions: 5 }}
```

- For inline styling:

```
<ul style={{
     backgroundColor: 'black',
     color: 'pink'
   }}>
```

## CSS Styling

- Inline styling

```
const buttonStyle = {
  backgroundColor: "blue",
  color: "white",
  padding: "10px",
  borderRadius: "5px"
};

function App() {
  return <button style={buttonStyle}>Click Me</button>;
}
```

- Importing via CSS files

```
import "./styles.css";

function App() {
  return <button className="button">Click Me</button>;
}
```