JSX

**JSX**

JSX stands for JavaScript XML. JSX allows us to write HTML elements with JavaScript code. An HTML element has an opening and closing tags, content, and attribute in the opening tag. However, some HTML elements may not have content and a closing tag - they are self closing elements. To create HTML elements in React we do not use the *createElement()* instead we just use JSX elements. Therefore, JSX makes it easier to write and add HTML elements in React. JSX will be converted to JavaScript on browser using a transpiler - [babel.js](https://babeljs.io/). Babel is a library which transpiles JSX to pure JavaScript and latest JavaScript to older version. See the JSX code below.

// JSX syntax

// we don't need to use quotes with JSX

const jsxElement = <h1>I am a JSX element</h1>

const welcome = <h1>Welcome to 30 Days of React Challenge</h1>

const data = <small>Oct 2, 2020</small>

The above strange looking code seems like JavaScript and it seems like , but it is not JavaScript and it seems like HTML but not completely an HTML element. It is a mix of JavaScript and an HTML elements. JSX can allow us to use HTML in JavaScript. The HTML element in the JSX above is *h1* and *small*.

**JSX Element**

As you have seen in the example above, JSX has a JavaScript and HTML like syntax. JSX element could be a single HTML element or many HTML elements wrapped in a parent HTML element.

This JSX element has only one HTML element which is *h1*.

const jsxElement = <h1>I am a JSX element</h1> // JS with HTML

Let's make more JSX elements by declaring a new variable named title and content inside *h2*.

const title = <h2>Getting Started React</h2>

Let us add a subtitles and other contents to this JSX element by adding additional HTML elements. Every HTML element should be wrapped by an outer HTML element to create a valid JSX element. The name title variable also should be changed to header because our JSX element is containing almost all of the header of the application.

const header = (

<header>

<h1>Welcome to 30 Days Of React</h1>

<h2>Getting Started React</h2>

<h3>JavaScript Library</h3>

</header>

)

Let us keep adding more elements. Additional HTML elements to display the author name and year.

const header = (

<header>

<h1>Welcome to 30 Days Of React</h1>

<h2>Getting Started React</h2>

<h3>JavaScript Library</h3>

<p>Asabeneh Yetayeh</p>

<small>Oct 2, 2020</small>

</header>

)

Example

Now, let us put everything together and render it to the browser. [Live on code pen](https://codepen.io/Asabeneh/full/MWwbYWg).

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>30 Days Of React Challenge</title>

</head>

<body>

<div class="root"></div>

<script

crossorigin

src="https://unpkg.com/react@16/umd/react.development.js"

></script>

<script

crossorigin

src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"

></script>

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>

<script type="text/babel">

// To get the root element from the HTML document

const rootElement = document.querySelector('.root')

// JSX element, header

const header = (

<header>

<h1>Welcome to 30 Days Of React</h1>

<h2>Getting Started React</h2>

<h3>JavaScript Library</h3>

<p>Asabeneh Yetayeh</p>

<small>Oct 2, 2020</small>

</header>

)

// JSX element, main

const main = (

<main>

<p>Prerequisite to get started react.js:</p>

<ul>

<li>HTML</li>

<li>CSS</li>

<li>JavaScript</li>

</ul>

</main>

)

// JSX element, footer

const footer = (

<footer>

<p>Copyright 2020</p>

</footer>

)

// JSX element, app, a container or a parent

const app = (

<div>

{header}

{main}

{footer}

</div>

)

// we render the JSX element using the ReactDOM package

// ReactDOM has the render method and the render method takes two argument

ReactDOM.render(app, rootElement)

// or

// ReactDOM.render([header, main, footer], rootElement)

</script>

</body>

</html>

