React JXS

What is JSX?

JSX stands for JavaScript XML.

JSX allows us to write HTML in React.

JSX makes it easier to write and add HTML in React.

JSX allows us to write HTML elements in JavaScript and place them in the DOM without any createElement()  and/or appendChild() methods.

JSX converts HTML tags into react elements.

You are not required to use JSX, but JSX makes it easier to write React applications.

**It uses JSX**

import React from 'react';

import ReactDOM from 'react-dom';

const myelement = <h1>I Love JSX!</h1>;

ReactDOM.render(myelement, document.getElementById('root'));

**It does not uses JSX**

import React from 'react';

import ReactDOM from 'react-dom';

const myelement = React.createElement('h1', {}, 'I do not use JSX!');

ReactDOM.render(myelement, document.getElementById('root'));

JSX is an extension of the JavaScript language based on ES6, and is translated into regular JavaScript at runtime.

## 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:

import React from 'react';

import ReactDOM from 'react-dom';

const myelement = <h1>React is {5 + 5} times better with JSX</h1>;

ReactDOM.render(myelement, document.getElementById('root'));

## Inserting a Large Block of HTML

To write HTML on multiple lines, put the HTML inside parentheses:

import React from 'react';

import ReactDOM from 'react-dom';

const myelement = (

<ul>

<li>Apples</li>

<li>Bananas</li>

<li>Cherries</li>

</ul>

);

ReactDOM.render(myelement, document.getElementById('root'));

## One Top Level Element

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

So if you like to write two headers, you must put them inside a parent element, like a div element

import React from 'react';

import ReactDOM from 'react-dom';

const myelement = (

<div>

<h1>I am a Header.</h1>

<h1>I am a Header too.</h1>

</div>

);

ReactDOM.render(myelement, document.getElementById('root'));

JSX will throw an error if the HTML is not correct, or if the HTML misses a parent element.

## Elements Must be Closed

JSX follows XML rules, and therefore HTML elements must be properly closed.

import React from 'react';

import ReactDOM from 'react-dom';

const myelement = <input type="text" />;

ReactDOM.render(myelement, document.getElementById('root'));

Fundamentally, JSX just provides syntactic sugar for the React.createElement(component, props, ...children) function. The JSX code:

<MyButton color="blue" shadowSize={2}>

Click Me

</MyButton>

compiles into:

React.createElement(

MyButton,

{color: 'blue', shadowSize: 2},

'Click Me'

)

You can also use the self-closing form of the tag if there are no children. So:

<div className="sidebar" />

compiles into:

React.createElement(

'div',

{className: 'sidebar'}

)