# React & JSX Objectives - Answers

## Define JSX

JSX (JavaScript XML) is a syntax extension for JavaScript used with React to describe how the UI should appear. It resembles HTML but compiles to `React.createElement()` calls.

Example:  
const element = <h1>Hello, JSX!</h1>;

## Explain about ECMA Script

## ECMAScript (ES) is the standard specification that JavaScript follows. It outlines the main features of the language. ES6 (ECMAScript 2015) and later versions added features like `let`, `const`, arrow functions, classes, and modules, which are common in React development.

## Explain React.createElement()

`React.createElement()` is the method that creates a React element. This method serves as the basic function for generating virtual DOM elements in React.

Example:  
const element = React.createElement('h1', null, 'Hello World');

## Explain how to create React nodes with JSX

React nodes (elements) can be created using JSX syntax. JSX allows you to write UI components in a format that resembles HTML, which is then converted into React elements.

Example:  
const node = <div><h2>Welcome</h2></div>;

## Define how to render JSX to DOM

JSX elements are rendered to the actual DOM using the `ReactDOM.render()` method. This function links your React code to an HTML DOM node.

Example:  
ReactDOM.render(<h1>Hello!</h1>, document.getElementById('root'));

## Explain how to use JavaScript expressions in JSX

You can embed JavaScript expressions in JSX using curly braces `{}`. These expressions can include variables, calculations, or function calls.

Example:  
const name = 'React';  
<h2>Hello, {name}!</h2>;

## Explain how to use inline CSS in JSX

Inline CSS in JSX is defined as a JavaScript object with camelCase properties. Instead of using traditional string-based CSS, you apply object notation directly in the `style` attribute.

Example:  
const style = { color: 'blue', fontSize: '20px' };  
<h1 style={style}>Styled Text</h1>;

**OUTPUT :-**

