JSX in React:	Example code
JavaScript + XML: JSX is a combination of JavaScript and XML-like syntax. It allows you to write HTML-like code within your JavaScript or React code.	
Declarative UI: JSX is a declarative way to define the structure of your user interface. You can express what your UI should look like and React will take care of updating the actual DOM to match it.	
Component Rendering: JSX is primarily used to define the structure of React components. You create components by writing functions or classes that return JSX elements.	
Embedding Expressions: You can embed JavaScript expressions within JSX using curly braces {}. This allows you to dynamically generate content or compute values and include them in your UI. For example:	const name = "John"; const greeting = Hello, {name}! ;
Attributes and Properties: In JSX, you can set HTML attributes and React component properties using a syntax similar to HTML.	
Self-closing Tags: JSX supports self-closing tags for elements without children, like <img /> or <input />.	

Babel Transformation: JSX code is not directly understood by browsers. It needs to be transpiled by tools like Babel into regular JavaScript code that creates and manipulates the DOM.	

Simple example of jsx in react component

```
import React from 'react';

function MyComponent() {
  return (
    <div>
      <h1>Hello, JSX!</h1>
      <p>This is a paragraph.</p>
    </div>
  );
}

export default MyComponent;
```

In this example, we define a React component MyComponent that returns JSX elements. When this component is rendered, it will create a div containing an h1 heading and a p paragraph, forming a simple user interface.
JSX simplifies the process of defining UIs in React and makes it more readable and maintainable compared to pure JavaScript DOM manipulation.