Current Skill JSX and HTML

Introduction

In the previous track, we have made some big steps and we learned how to create a web page using **HTML**, **CSS** and **JS**.

In this chapter, we will explore how to create elements in React, using **JSX**.

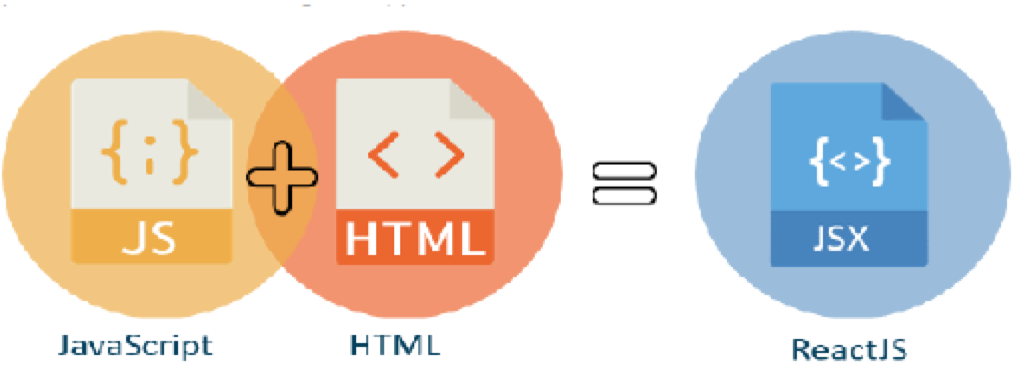
JSX is an extension to JavaScript and it helps when we want to produce React elements.

We will be reviewing its features and how it operates.

What's JSX?

JSX is a syntax extension to JavaScript usually used with React to describe what the UI should look like.

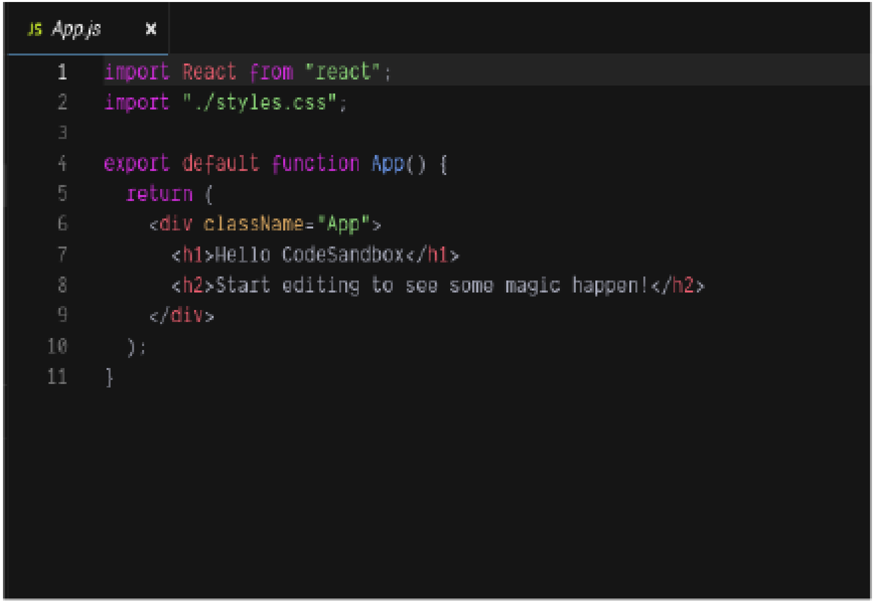
JSX may remind you of a template language, but it comes with the full power of JavaScript.



JSX and HTML

If you take a look at the **App.js** which is inside the src folder, you will find a combination of **JavaScript** and **HTML (kind of HTML)**.

Inside the function **App** in the "return" statement, You will find a code that looks like **HTML**. Notice when you edit that code, the page refreshes automatically in the browser. That's called **Live Reload** or **Hot Reload**.



```
import React from "react";
import "./styles.css";

export default function App() {
  return (
    <div className="App">
      <h1>Hello CodeSandbox</h1>
      <h2>Start editing to see some magic happen!</h2>
    </div>
  );
}
```

JSX and HTML

Take a look at the function **App**, You are probably wondering why HTML is placed awkwardly inside JavaScript. This unusual tag syntax is **neither a string nor HTML**. It's called **JSX**.

```
function App() {
 return (
   <div className="App">
     <header className="App-header">
       <img src={logo} className="App-logo" alt="logo" />
       <p>
         Edit <code>src/App.js</code> and save to reload.
       </p>
       <a
         className="App-link"
         href="https://reactjs.org"
         target="_blank"
         rel="noopener noreferrer"
       >
         Learn React
       </a>
     </header>
   </div>
```

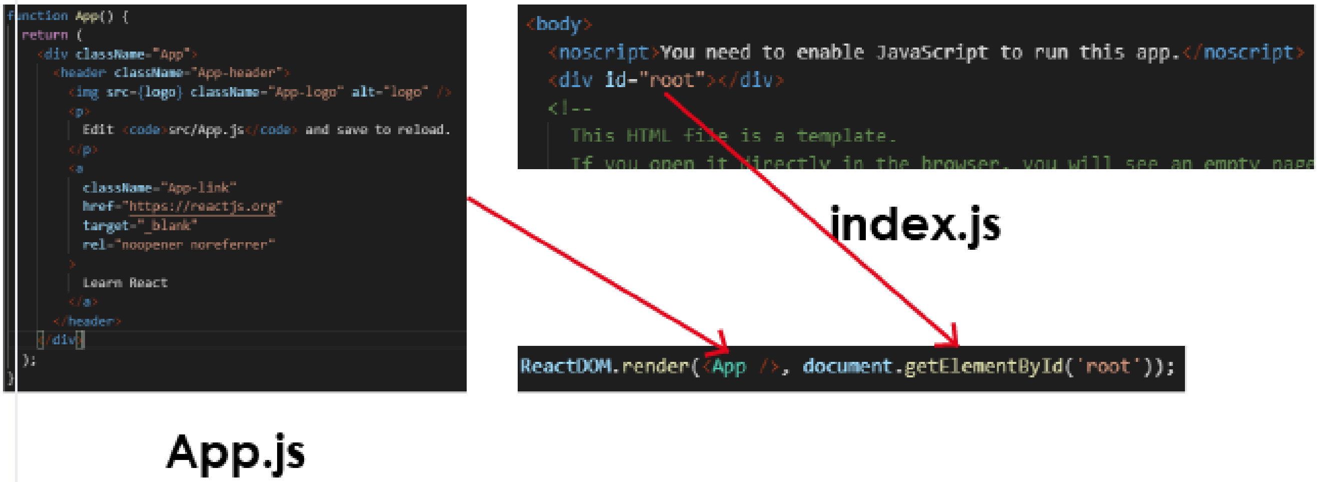
```
  );
}
```

How does it work?

At first glance, JSX and HTML look very similar, but in reality they are very different. Working with JSX makes the manipulation of the HTML elements much easier and more effective.

Since every website is built with HTML and browsers naturally understand HTML and not JSX, you might be wondering: what is the link between JSX and HTML?

Tip: *if you inspect the source code of React's webpage in your browser, you will only find HTML and not JSX.*



```
function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}
```

```
<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>
  <!--
    This HTML file is a template.
    If you open it directly in the browser, you will see an empty page
```

```
ReactDOM.render(<App />, document.getElementById('root'));
```

Thanks to React, JSX is transformed into HTML and then it gets injected into an **HTML element** (root) in the well-known HTML page, the **index.html**.

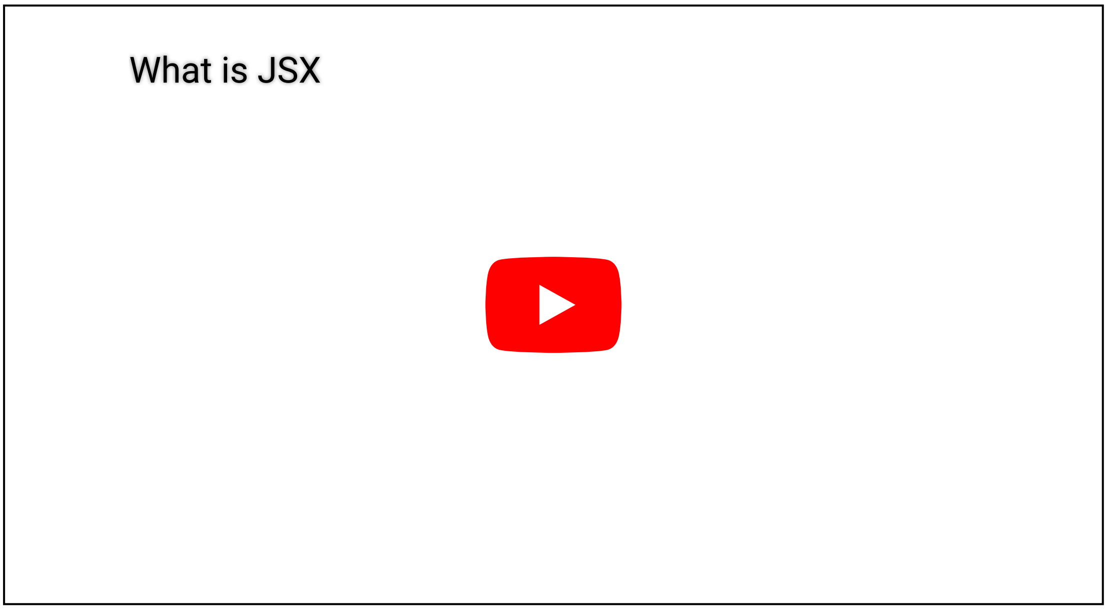
This injection is actually happening in the **index.js** file thanks to **ReactDOM**. This process only happens one time, because the **App** is the only root component in our case (we usually need only one root component).

JSX and HTML Recap:

Let's recap!

- The injection of a component inside an HTML element is called **rendering**.
- In React, we usually have only one root component (called **App**).
- We only need one **root component** to build a React website.

- The root **component (App.js)** is the entry point of **React**.
- We only need one HTML file called **index.html** in any single-page application (like React).
- JSX is compiled and rendered inside the **root element** in the **index.html**.



‹ Previous

next ›