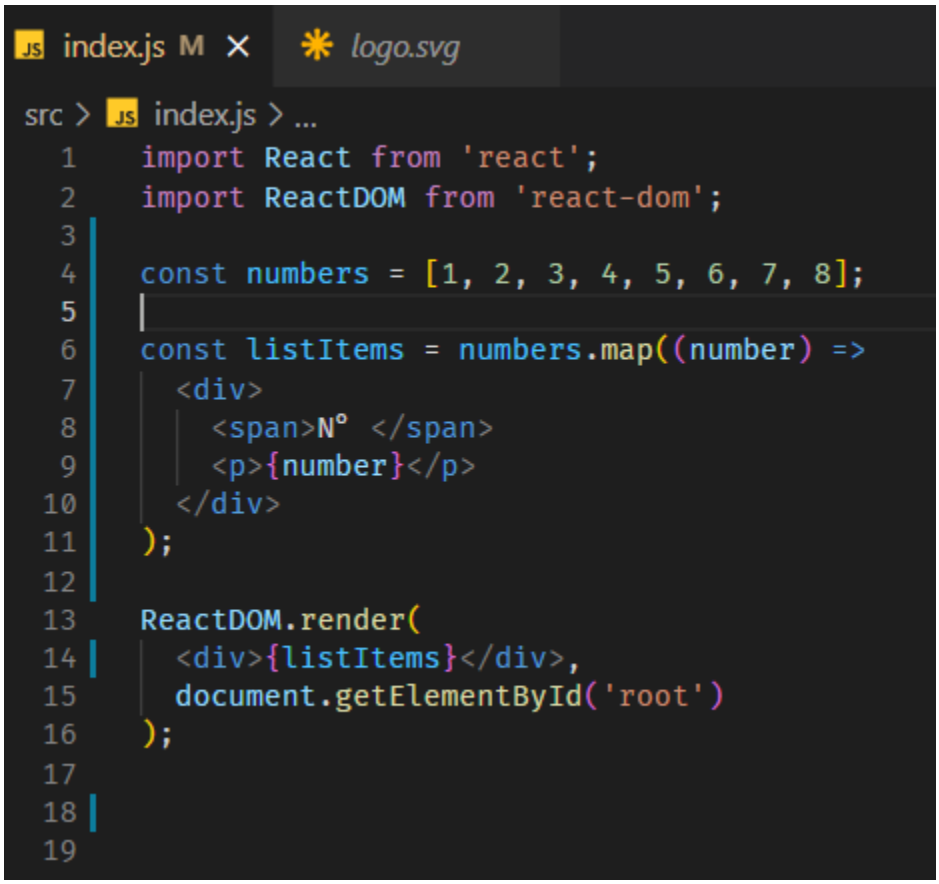
Certified Tech Developer
The Ultimate Degree

Front End III: Frameworks

Fragment y Children

Cuando se trabaja con componentes en React, es necesario retornarlos dentro de una etiqueta que los envuelva. Esto produce que en un HTML se creen etiquetas vacías innecesarias y lo resuelve Fragment. Una posible solución es agrupar todo en una , pero generaría etiquetas extra una y otra vez. Para ello React nos da la posibilidad de usar Fragment y así utilizar un envoltorio que finalmente no generará un nodo extra en nuestro DOM.

Aquí un ejemplo de un array map que retorna un y un <p> envueltos en un sin Fragment:



```
▼<body cz-shortcut-listen="true"> == $0
  <noscript>You need to enable JavaScript to run this app.</noscript>
  ▼<div id="root">
    ▼<div>
      ▼<div>
        ▼<div>
          <span>N° </span>
          <p>1</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>2</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>3</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>4</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>5</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>6</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>7</p>
        </div>
        ▼<div>
          <span>N° </span>
          <p>8</p>
        </div>
      </div>
    </div>
    <!--
```

Aquí un ejemplo usando Fragment:

```
index.js > ...
import React from 'react';
import ReactDOM from 'react-dom';

const numbers = [1, 2, 3, 4, 5, 6, 7, 8];

const listItems = numbers.map((number) =>
  <React.Fragment>
    <span>N° </span>
    <p>{number}</p>
  </React.Fragment>
);

ReactDOM.render(
  <>{listItems}</>,
  document.getElementById('root')
);
```

```
<body cz-shortcut-listen="true">
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root">
    <span>N° </span>
    <p>1</p>
    <span>N° </span>
    <p>2</p>
    <span>N° </span>
    <p>3</p>
    <span>N° </span>
    <p>4</p>
    <span>N° </span>
    <p>5</p>
    <span>N° </span>
    <p>6</p> == $0
    <span>N° </span>
    <p>7</p>
    <span>N° </span>
    <p>8</p>
  </div>
```

Es notorio el ahorro en líneas y la simpleza del código con el uso de Fragment. Por último hay 2 formas de implementar Fragment:

<></>

<React.Fragment></React.Fragment>

Children

Seguramente van a existir casos en los que tengamos un componente y queramos tan dinámico su uso que no va a ser suficiente con el simple uso de las props. Es aquí en donde cobran un particular protagonismo los **children**. A través de ellos vamos a tener la capacidad de enviar (como si fuera una prop) cualquier tipo de estructura HTML.

¿Cuándo usamos un children?

En caso de no saber qué contenido puede llegar a haber dentro de un componente padre, es común que los componentes actúen como cajas o contenedores genéricos de otros componentes. En estos casos es recomendable usar la prop special children para pasar elementos hijos directamente en el resultado. Esto permite que otros componentes pasen hijos arbitrarios anidando el JSX.

index.js > ...

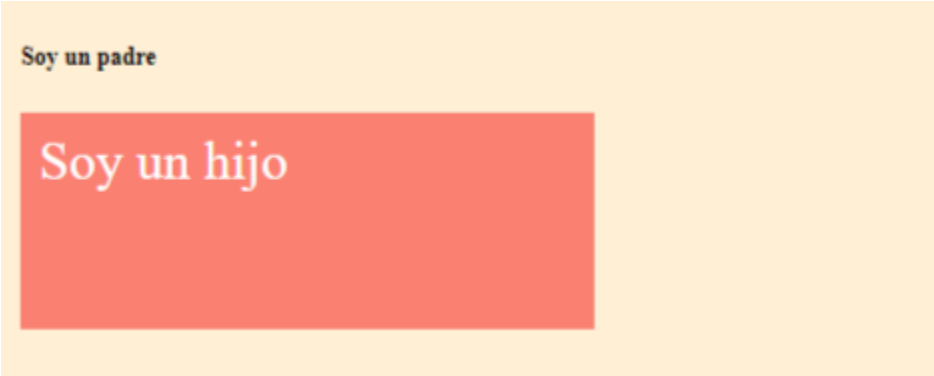
```jsx
import React from "react";
import ReactDOM from "react-dom";

const Padre = (props) => {
  return (
    <div
      style={{
        width: "75%",
        background: "#FFEFD5",
        height: "200px",
        padding: "20px",
      }}
    >
      <h5>Soy un padre</h5>
      {props.children}
    </div>
  );
};

const Hijo = (props) => {
  return (
    <div
      style={{
        width: "50%",
        background: "#FA8072",
        height: "100px",
        padding: "10px",
        color: "white",
        fontSize: "30px",
      }}
    >
      {props.autor}
    </div>
  );
};

const App = () => {
  return (
    <Padre>
      <Hijo autor="Soy un hijo" />
    </Padre>
  );
};

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



Cualquier elemento que pasemos dentro de la etiqueta JSX <Padre> se pasará a Padre como prop.children. Como resultado obtendremos al Hijo envuelto en el contenedor Padre.

¡Hasta la próxima!