Sesión # 8 Componente Práctico

Desarrollo de Front-end web con React

Como pudiste aprender en el componente práctico, una de las ventajas de React es su capacidad de renderizar solo una sección de la aplicación cuando se activa un disparador, a diferencia de las aplicaciones web tradicionales donde se renderiza todo el documento incluso si solo cambia una pequeña parte del DOM. Un componente react puede poseer algunas propiedades intrínsecas, que cambian cuando algo le sucede al componente. Tales propiedades son lo que llamamos estado del componente.

Los estados se originan dentro del componente que lo posee; esto hace que el estado sea privado y solo accesible para el componente propietario. En algunas ocasiones cuando queremos que el componente principal sepa acerca del estado de un componente hijo, debemos elevarlo al nivel principal, por lo tanto, "Levantar".

En el presente componente práctico vamos a crear un formulario que reciba un texto desde un componente y eleve su estado hacia otro.

1. Crea un nuevo proyecto de react al que llamaremos sesion8
   npx create-react-app sesion8

2. Borra todos los archivos existentes en la carpeta src (recuerda no eliminar la carpeta, solo los archivos dentro de ella). Si deseas puedes hacer uso de la siguiente plantilla: https://github.com/Misiontic-Ciclo-4A/react-template

3. Crea un componente funcional FormTextInput, que utilice prop para establecer el estado inicial de la leyenda en el fieldset. Ten en cuenta que el valor de props.label nunca cambiará durante la vida útil del componente FormTextInput. Replica:

```jsx
import React from "react";
import "./index.css";

function FormTextInput(props) {
  return (
    <fieldset>
      <legend>{props.label}</legend>
      <input type="text" placeholder="Enter a text" onChange={props.onChange} />
    </fieldset>
  );
}

export default FormTextInput;
```

4. Ahora, crea un nuevo componente funcional Form.jsx donde haremos el llamado al componente FormTextInput. Este componente tiene una jerarquía más alta en

comparación con FormTextInput. Como puedes ver, no hay forma de que el componente Form pueda recuperar el estado de FormTextInput. ¡Aquí es donde tenemos que levantarlo!

```
import "./index.css";
import FormTextInput from "./FormTextInput";
import { useState } from "react";

function Form(props) {
  return (
    <div className="form">
      <h2 className="result">{props.title}</h2>
      <FormTextInput label="Our label" />
    </div>
  );
}

export default Form;
```

Queremos monitorear el cambio en el componente FormTextInput (hijo) para actualizar el texto del Título del formulario en el componente Form (padre). Aquí está la definición del componente, con algunas modificaciones…

```
import React from "react";
import "./index.css";
import FormTextInput from "./FormTextInput";
import { useState } from "react";

function Form(props) {
  const [title, setTitle] = useState(props.title);
  const handleChange = (e) => {
    let target = e.target;
    if (!e.target.value) setTitle(() => ({ value: props.title }));
    else {
      setTitle(() => ({ value: target.value }));
    }
  };

  return (
    <div className="form">
      <h2 className="result">{title.value}</h2>
      <FormTextInput onChange={handleChange} label="Our label" />
    </div>
  );
}

export default Form;
```

La función handleChange del componente Form supervisa y controla los cambios en FormTextInput. Esta es la función que deberemos llamar cada vez que cambia la entrada de nuestro formulario, por lo que la pasamos como prop en onChange al componente FormTextInput.

```jsx
import React from "react";
import "./index.css";

function FormTextInput(props) {
  return (
    <fieldset>
      <legend>{props.label}</legend>
      <input type="text" placeholder="Enter a text" onChange={props.onChange} />
    </fieldset>
  );
}

export default FormTextInput;
```

Ahora nuestro componente FormTextInput recibe una propiedad llamada onChange que contiene una función que maneja los cambios. Aquí es donde ocurre el levantamiento de estado. En el componente FormTextInput, ya no tenemos un controlador para el cambio, sino que usamos solo el que pasó del padre a través de la función onChange.

5. Crea una archivo index.css y añade el siguiente código:

```css
body {
  text-align: center;
  font-family: sans-serif;
  color: black;
}
.form {
  display: block;
  border: 0px solid rgb(0, 145, 255);
  text-align: center;
  padding: 1em;
}
.result {
  color: rgb(0, 145, 255);
  font-weight: bold;
  font-size: 1.5em;
  text-transform: capitalize;
}
fieldset {
  text-align: left;
```

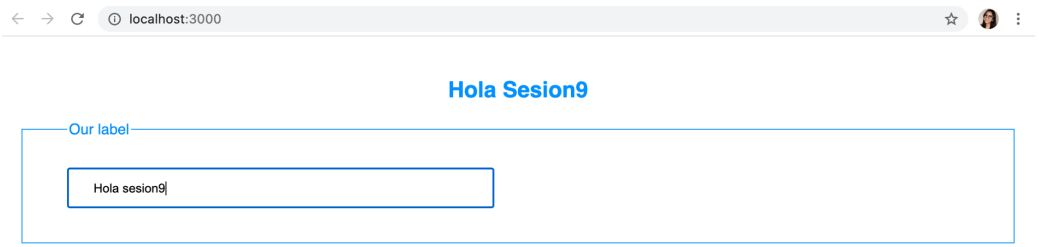
```
  padding-left: 3em;
  color: rgb(0, 145, 255);
  border: 1px solid rgb(0, 145, 255);
}
input[type="text"] {
  width: 40%;
  height: 2em;
  padding-left: 2em;
  padding-right: 2em;
  padding-top: 0.5em;
  padding-bottom: 0.5em;
  margin: 2em auto;
}
```

6. Importa los elementos necesarios y renderiza el componente Form en el archivo index.js

```
import React from "react";
import ReactDOM from "react-dom";
import "./index.css";
import Form from "./Form";

ReactDOM.render(
  <React.StrictMode>
    <Form title="Form Title" />
  </React.StrictMode>,
  document.getElementById("root")
);
```

7. Ejecuta desde la terminal npm start y mira tu aplicación. Como puedes ver, el título se actualiza a medida que ingresas información en el formulario y esto gracias a los conceptos aprendidos.



8. Investiga las diferencias en React entre 'props' y estados

PROYECTO SOLUCIÓN: https://github.com/Misiontic-Ciclo-4A/sesion8-solucion