Componentes en React

Un componente en React es una pieza autónoma y encapsulada de código que se puede utilizar para construir interfaces de usuario complejas. Estos componentes pueden contener tanto la lógica como la interfaz de usuario necesaria para renderizar una determinada parte de una aplicación.

Son bloques de construcción fundamentales para la creación de interfaces de usuario en aplicaciones web. Proporcionan modularidad, reutilizabilidad y un flujo eficiente de actualización del DOM, lo que hace de React una biblioteca popular para el desarrollo frontend en JavaScript.

Componentes en React

Los componentes nos van a permitir sacar pequeños fragmentos de nuestro código y encapsularlos para luego poder reutilizarlos en varias partes de nuestra aplicación. La sintaxis básica para crear un componente es en forma de función flecha:

```
1  const MiComponente = () => {
2    return (
3      <div>
4        <h1>Hola, soy un componente</h1>
5        <p>¡Este es mi contenido!</p>
6      </div>
7    );
8  }
```

Los nombres de los componentes deben empezar con una letra mayúscula y seguir la regla de CamelCase. Por ejemplo, `MiComponente` en lugar de `Mi_componente`.

También puedes simplificar aún más la sintaxis utilizando una versión abreviada:

```
1  const MiComponente = () => (
2    <div>
3      <h1>Hola, soy un componente</h1>
4      <p>¡Este es mi contenido!</p>
5    </div>
6  );
```

En esta forma abreviada, no es necesario utilizar la palabra clave `return` ni las llaves `{}`. El valor devuelto por la función se considera implícitamente como el resultado.

Componentes en React

A continuación, usaremos componentes para transformar nuestro código escrito anteriormente, por ello, crearemos un componente "Aula" dónde guardaremos toda la información que se estaba mostrando anteriormente en nuestra App.

```
10  const Aula = () =>{
11    return(
12      <>
13        <h1 style={{color:color}}>¡Hola {nombre}!</h1>
14        <p>Te doy la Bienvenida a esta Experiencia de Aprendizaje</p>
15        {clase && <p>Estamos en clase del {clase}</p>}
16        <h4>Estos son los profesores:</h4>
17        <ul>
18        {instructores.map((instructor, index) => {
19          return <li key={index}>{instructor}</li>
20        })}
21        </ul>
22      </>
23    );
24  };
```

Componentes en React

Seguidamente crearemos un componente llamado “App" en el cuál se retornará el código JSX de nuestra App.

```
26  const App = () =>{
27    return(
28      <>
29        {sesion === true ?
30        <>
31          <Aula/>
32        </>
33        :
34        <h1>No has iniciado sesión</h1>
35        }
36      </>
37    )
38  };
```

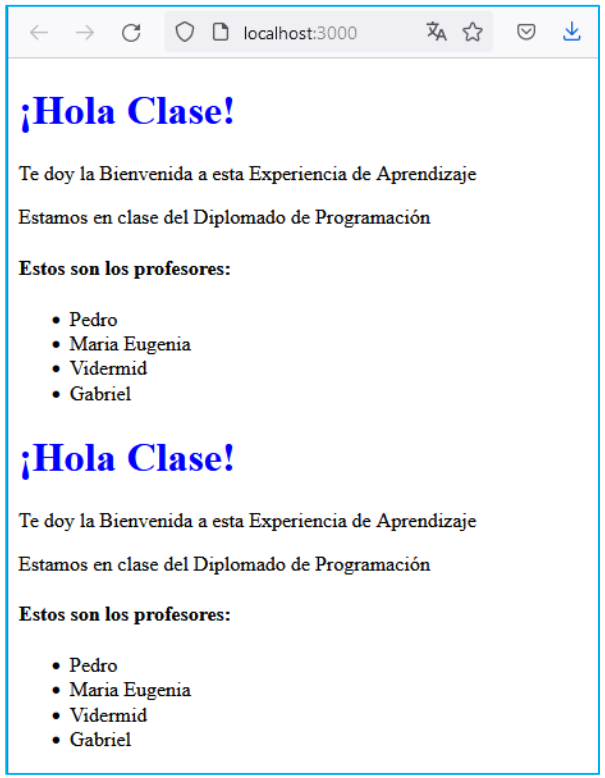
Y para finalizar sustituiremos la variable “JSX" por el componente “App" en la línea final de nuestro código, para así, mandar a imprimir nuestro componente que contiene toda la App.

```
40  const root = ReactDOM.createRoot(document.getElementById('root'));
41  root.render(<App/>);
```

Componentes en React

De esta forma hemos optimizado nuestra aplicación y ya será mucho más sencillo reutilizar el código. Para probarlo, colocaremos 2 veces el componente "Aula" dentro del componente "App" y de esa forma veremos en pantalla el código indicado 2 veces:

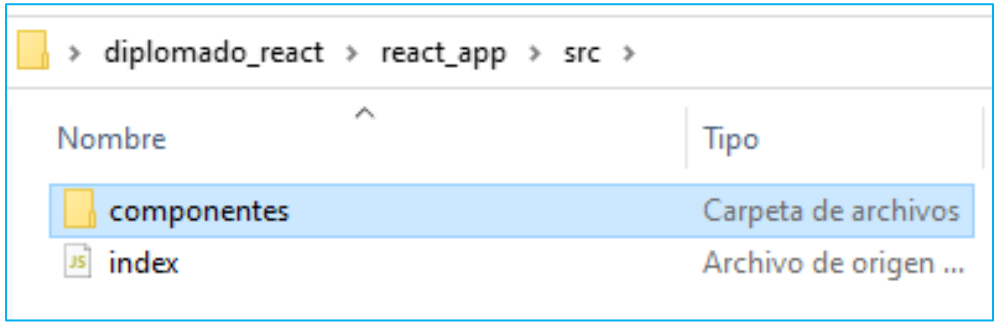
```
26  const App = () =>{
27    return(
28      <>
29        {sesion === true ?
30        <>
31          <Aula/>
32          <Aula/>
33        </>
34        :
35        <h1>No has iniciado sesión</h1>
36        }
37      </>
38    )
39  };
```

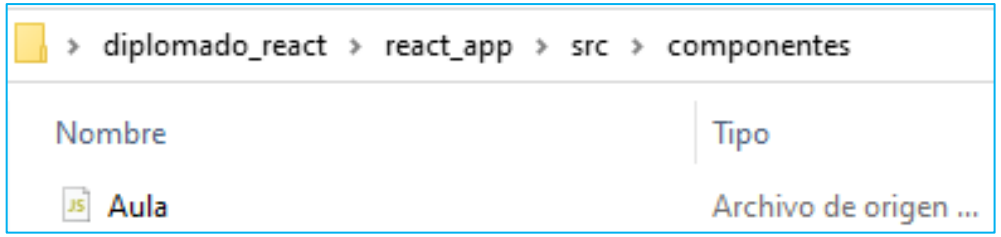


Organización de Componentes

Lo ideal es que cada componente sea ubicado en un archivo separado para mantener una mejor estructura en nuestra aplicación y así tener un ambiente optimo para poder escalar el proyecto y realizar cambios.

Tomando en cuenta esto, dentro de "src" crearemos una carpeta "componentes" con todas sus letras en minúsculas y dentro de la misma haremos un archivo .js por cada componente, respetando la regla de CamelCase. En este caso crearemos el archivo para nuestro componente "Aula":





Organización de Componentes

En este archivo importamos react, colocamos el componente Aula y dentro del mismo, antes del "return" añadimos las variables/constantes que necesita para mostrar todo su contenido:

```
1   import React from 'react';
2   const Aula = () =>{
3       const nombre = "Clase";
4       const color = "blue"
5       const clase = "Diplomado de Programación";
6       const instructores = ["Pedro", "Maria Eugenia", "Vidermid", "Gabriel"];
7       return(
8         <>
9           <h1 style={{color:color}}>¡Hola {nombre}!</h1>
10          <p>Te doy la Bienvenida a esta Experiencia de Aprendizaje</p>
11          {clase && <p>Estamos en clase del {clase}</p>}
12          <h4>Estos son los profesores:</h4>
13          <ul>
14          {instructores.map((instructor, index) => {
15            return <li key={index}>{instructor}</li>
16          })}
17          </ul>
18        </>
19      );
20  };
```

En este punto, luego de guardar los cambios, debemos borrar del index el componente de "Aula" y las variables/constantes que usa el mismo.

Luego de esto, comenzará a salir un error en nuestra aplicación porque no se encuentra definido el componente de <Aula />

Exportación e Importación Individual de Componentes

Ahora que tenemos nuestro componente en un archivo separado del index, debemos exportarlo, para ello colocaremos en la última línea "export default" seguido del nombre del componente:

```
22    export default Aula;
```

Seguidamente en el archivo index, debemos importar nuestro componente colocando en las primeras líneas "import NombreDelComponente from ruta":

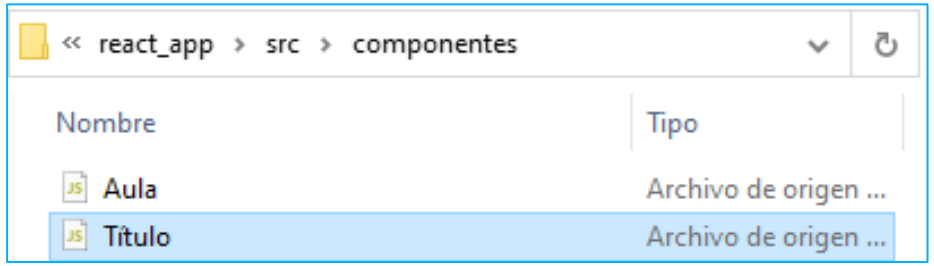
```
1    import React from 'react';
2    import ReactDOM from 'react-dom/client';
3    import Aula from './componentes/Aula';
```

De esta forma ya estamos trabajando correctamente la aplicación con nuestros componentes organizados, pero podemos segmentarla más dependiendo de la necesidad, haciendo más componentes pequeños para cada parte de la misma.

Anidación de Componentes

La anidación de componentes en React es una forma de componer una interfaz de usuario más compleja combinando componentes más pequeños en una estructura jerárquica. En React, se pueden colocar componentes dentro de otros componentes de manera similar a cómo se anidan las etiquetas HTML, a continuación, vamos a ver cómo hacerlo a través del a creación del componente Título y su posterior inserción dentro del componente "Aula".

Lo primero que debemos hacer es crear el archivo para nuestro componente "Título" dentro de la carpeta de componentes:



Anidación de Componentes

En este archivo importamos react, creamos el componente Titulo y dentro del mismo, añadimos las variables/constantes que necesita para mostrar todo su contenido, seguidamente el return y dentro del mismo colocamos las etiquetas <h1> con su estilo y contenido:

```
1   import React from 'react';
2   const Titulo = () =>{
3       const nombre = "Clase";
4       const color = "blue"
5       return(
6         <>
7           <h1 style={{color:color}}>¡Hola {nombre}!</h1>
8         </>
9       );
10  };
11
12  export default Titulo;
```

Asimismo, en la línea 12 exportamos nuestro nuevo componente.

Anidación de Componentes

Ahora en el archivo del componente Aula, vamos a importar el componente Titulo, borramos las constantes de nombre y color y sustituimos la línea que muestra las etiquetas <h1> por el componente <Titulo />. De esta forma ya hemos completado nuestra primera anidación:

```
1   import React from 'react';
2   import Titulo from './Titulo';
3   const Aula = () =>{
4       const clase = "Diplomado de Programación";
5       const instructores = ["Pedro", "Maria Eugenia", "Vidermid", "Gabriel"];
6       return(
7         <>
8           <Titulo />
9           <p>Te doy la Bienvenida a esta Experiencia de Aprendizaje</p>
10          {clase && <p>Estamos en clase del {clase}</p>}
11          <h4>Estos son los profesores:</h4>
12          <ul>
13          {instructores.map((instructor, index) => {
14            return <li key={index}>{instructor}</li>
15          })}
16          </ul>
17        </>
18      );
```

Exportación de Varios Componentes

Para aprender a exportar varios componentes, vamos a nuestro archivo de Titulo, cambiaremos el nombre a nuestro componente de Titulo por TituloAzul y seguidamente duplicaremos ese componente para cambiarle el nombre al duplicado por TituloNaranja y en el valor de la constante "color" del mismo colocaremos "orange":

```
1   import React from 'react';
2   const TituloAzul = () =>{
3       const nombre = "Clase";
4       const color = "blue"
5       return(
6         <>
7           <h1 style={{color:color}}>¡Hola {nombre}!</h1>
8         </>
9       );
10  };
```

```
11  const TituloNaranja = () =>{
12      const nombre = "Clase";
13      const color = "orange"
14      return(
15        <>
16          <h1 style={{color:color}}>¡Hola {nombre}!</h1>
17        </>
18      );
19  };
```

Para la exportación, se debe colocar al final la palabra export y seguidamente, entre llaves y separado por comas (,) el nombre de cada componente a exportar:

```
21  export {TituloAzul,TituloNaranja};
```

Importación de Varios Componentes

Luego, en el archivo aula.js debemos importar nuestros componentes colocando en las primeras líneas "import {Componente1, Componente2, ... , ComponenteN} from ruta":

```
1   import React from 'react';
2   import {TituloAzul,TituloNaranja} from './Titulo';
```

De esta forma, ya podemos usarlos y vamos a hacerlo modificando nuestro componente aula, colocando ambos componentes de título para que luego ellos sean mostrados en pantalla:

```
3   const Aula = () =>{
4       const clase = "Diplomado de Programación";
5       const instructores = ["Pedro", "Maria Eugenia", "Vidermid", "Gabriel"];
6       return(
7         <>
8           <TituloAzul />
9           <TituloNaranja />
10          <p>Te doy la Bienvenida a esta Experiencia de Aprendizaje</p>
11          {clase && <p>Estamos en clase del {clase}</p>}
12          <h4>Estos son los profesores:</h4>
```