Argentina programa 4.0

UNIVERSIDAD TECNOLOGICA NACIONAL

Desarrollo en React JS

Componentes funcionales - props - Hooks

Desarrollador Web con React Js

Diferencia entre estado y props

Argentina programa 4.0

Los estados actúan en el contexto del componente y por otro, las propiedades crean una instancia del componente cuando le pasas un nuevo valor desde un componente padre.

Los valores de las propiedades los pasas de padres a hijos y los valores de los estados los defines en el componente, no se inician en el componente padre.

Estado en clases

Argentina programa 4.0

```
1 class App extends React.Component {
2   constructor(props) {
3     super(props);
4     this.state = {date: new Date()};
5   }
6
7   render() {
8     return (
9       <div>
10        Hoy es{this.state.date.toLocaleTimeString()}
11      </div>
12    );
13  }
14 }
```

En los componentes de clase el estado se define como una propiedad de la clase llamada **state** representada por un objeto literal de Javascript, la cual podemos modificar mediante el uso del método **setState** que todos los componentes que extienden a **React.Component** tienen.

Estado en componentes funcionales

Argentina programa 4.0

```
1 import React, { useState } from "react";
2
3 const App = (props) => {
4
5   const [hoy, setHoy] = useState(new Date());
6
7   return (
8     <div>
9       Hoy es {hoy.toLocaleDateString()}
10    </div>
11  );
12 }
13
14 export default App;
```

En el caso del componentes funcionales podemos hacer uso del hook **useState** para definir una variable local a nuestro componente.

Dicha variable podría ser modificada con la función asociada a la misma (en este caso setHoy).

Hooks

Argentina programa 4.0

Los hooks son **funciones especiales** que permiten “conectarnos” a distintas características de React y permiten su reutilización entre componentes.

Por ejemplo usando componentes de clase, la lógica de manejo de estado sería muy difícil (sino imposible) de reutilizar en otros componentes.

Creando nuestros propios hooks podríamos compartir esta funcionalidad con todos los componentes que la necesitaran.

Algunos ejemplos de Hooks son:

- useState
- useEffect

Eventos

Argentina programa 4.0

Los eventos en React se manejan de forma similar a los eventos de HTML con algunas diferencias:

- En React los eventos se nombre con **camelCase** en vez de todo minuscula
- JSX permite pasar una función como manejador del evento en vez de un string como HTML.
- En React no podemos devolver f**alse** para prevenir el comportamiento por defecto del evento. Debemos llamar explícitamente al método preventDefault().

Ejemplo de manejo de click

Argentina programa 4.0

```
1 const MiBoton = (props) => {
2     const [prendido, setPrendido] = useState(false);
3
4     return (
5         <button onClick={() => setPrendido(!prendido)}>
6             {prendido ? 'Apagar' : 'Prender'}
7         </button>
8     );
9 }
```

Listar elementos

Argentina programa 4.0

La forma más común de listar elementos en React es utilizando el método **map()** de los arrays.

De esta forma podemos iterar la lista completa obteniendo cada uno de los ítems de la lista en cada iteración y utilizarlo para devolver un componente u objeto.

Cada vez que listemos elementos de forma dinámica React nos va a pedir que incluyamos una propiedad llamada **key** (sería el id de cada elemento). Esto facilita a React el trabajo de modificar el DOM. **En caso de omitirlo veremos una advertencia en la consola.**

Ruteo

Argentina programa 4.0

Mediante el ruteo podremos hacer que nuestra aplicación o sitio hecho en React responda a diferentes urls y pueda navegar entre ellas.

Como esta no es una funcionalidad incluida en React utilizaremos una librería llamada **React Router DOM** y sus distintos componentes.

Los principales componentes de la librería son:

- **BrowserRouter:** Es una envoltura para nuestra aplicación. Esta envoltura nos da acceso al API de historia de HTML5 para mantener nuestra interfaz gráfica en sincronía con la locación actual o URL. Debemos tener en cuenta que esta envoltura solo puede tener un hijo. Por lo general es Switch.

- **Switch:** Este componente, causa que solo se renderice el primer hijo Route o Redirect que coincida con la locación o URL actual. En el caso que no usemos Switch todas las rutas que cumplan con la condición se renderizarán.
- **Route:** Para definir las diferentes rutas de nuestra aplicación, podemos usar el componente Route. La función de este componente es elegir que renderizar según la locación actual.

  Este componente recibe las siguientes propiedades:

  - **path**: la ruta en la que se renderizará el componente en forma de cadena de texto.
  - **exact**: un booleano para definir si queremos que la ruta tiene o no que ser exacta para renderizar un componente. Ej: /index !== /index/all.
  - **component**: recibe un componente a renderizar. Crea un nuevo elemento de React cada vez. Esto causa que el componente se monte y desmonte cada vez (no actualiza).

```jsx
import React, { Component } from 'react';
import { BrowserRouter, Switch, Route } from 'react-router-dom';
export default class App extends Component {
  render() {
    return (
      <BrowserRouter>
        <Switch>
          <Route path='/' exact component={HomePage} />
          <Route path='/nosotros' exact component={NosotrosPage} />
        </Switch>
      </BrowserRouter>
    );
  }
}
```

- **Link**

  Crea un hipervínculo que nos permite navegar por nuestra aplicación, agrega una nueva locación a la historia.

  Su principal propiedad es:

  to: una locación en forma de cadena de text. Esta será la locación a la que navegaremos cuando le damos click a un hipervínculo.

Argentina programa 4.0

```
1 import React, { Component } from 'react';
2 import { Link } from 'react-router-dom';
3 import './NavBar.css';
4 class NavBar extends Component {
5   render() {
6     return (
7       <div className="NavBar">
8         <div className="link-container">
9           <Link to="/page1" className="link">Página 1</Link>
10        </div>
11        <div className="link-container">
12          <Link to="/page2" className="link">Página 2</Link>
13        </div>
14      </div>
15    );
16  }
17 }
18 export default NavBar;
```

Gracias!