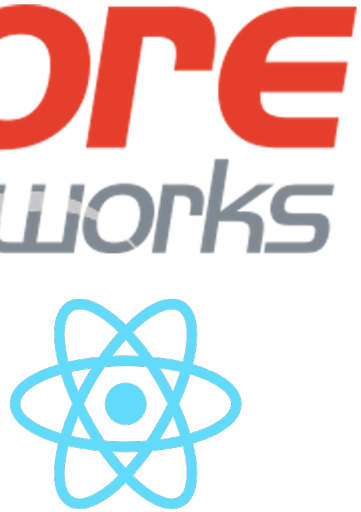
Desarrollo y programación de Aplicaciones con React

4. Componentes de tipo clase



Componentes de clase

Un componente de clase es una clase JavaScript que extiende la clase Component de React.

Permite guardar su estado y controlar lo que ocurre durante su ciclo de vida, exponiendo métodos como componentDidMount o componentWillUnmount.

Componentes de clase

```
class Identificador extends React.Component {
    constructor(props){
        super(props)
        this.state = {
            //
        }
    }

    componentDidMount(){
        //
    }

    componentWillUnmount(){
        //
    }

    render(){
        return //
    }
}
```

Componentes de clase

Props y Estado

Las props y los estados son ambos atributos de una clase, por eso puedes utilizar this.state y/o this.pros, pero tienen propósitos diferentes: mientras las propiedades son inmutables, los valores de los estados pueden cambiar, son mutables.

Los estados actúan en el contexto del componente mientras que las propiedades crean una instancia del componente cuando le pasas un nuevo valor desde un componente padre.

Componentes de clase

Props y Estado

El objeto estado, que es el que se modifica en el ciclo de vida de cada componente, tiene una particularidad, y es que cuando es modificado se vuelve a ejecutar el método render() y por tanto, se actualiza el componente.

Componentes de clase

Props y Estado

Por tanto, los valores de las props se pasan de padres a hijos y los valores del estado se definen en el componente.

Una particularidad de las props es que, como son inmutables, si se modifican esos cambios no se propagan a los elementos hijos, con lo cual las props sirven para pasar valores pero, si estos se van a modificar, deberán estar asociados al estado del componente padre.

Componentes de clase

setState()

Los cambios en el estado no se pueden modificar por asignación directa, sino que deberemos usar el método setState() de la clase Component de React.

this.setState(updater, callback-opcional)

El método recibe un updater, objeto con la propiedad del estado con su nuevo valor, pero no es necesario incluir en él las propiedades del estado que no cambian.

Componentes de clase

setState()

El updater también puede ser una función callback cuando tengamos que usar el valor previo de la propiedad a actualizar en la expresión.

~~this.setState({propiedad: this.state.propiedad + 1})~~

```
this.setState(prevState => {
    return ({propiedad: prevState.propiedad + 1})
})
```

Componentes de clase

setState()

this.setState(updater, callback-opcional?)

La callback opcional como segundo parámetro sirve para que ejecutar cualquier lógica una vez que se haya actualizado el valor de esa propiedad del estado. Esta callback está disponible porque React, para optimizar las operaciones asíncronas en jerarquías complejas tiene un cierto delay en la actualización del estado.

Componentes de clase

Propiedad key en las iteraciones

También para optimizar las actualizaciones de elementos HTML renderizados por la iteración de un array, React exige la incorporación de una propiedad key en cada elemento con un valor único, de manera que en los procesos de re-renderizado, identificará con esta clave el elemento a modificar.

<elemento key=<valor-único> ... />

No es conveniente generar esa clave en tiempo de ejecución.

Componentes de clase

Eventos y bind

Para el manejo de eventos, React recomienda usar método manejadores (se emplea el prefijo handle) de la clase del componente que son invocados desde los elementos HTML usando el evento JavaScript en formato camelCase para cumplir con la sintaxis JSX los cuales reciben la invocación del método.

```
handleEvento() {...}

<elemento onClick={this.handleEvento} />
```

Componentes de clase

Eventos y bind

Si el método manejador necesita modificar el estado, algo bastante frecuente, debemos “bindear” el método al this del componente en el constructor.

```
constructor(props) {
    ...
    this.handleEvento = this.handleEvento.bind(this);
}

handleEvento() { this.setState(...)}
```

Componentes de clase

Eventos y bind

Una curiosidad de React es que cuando invocamos el método manejador no se utilizan los paréntesis, esto se debe a que si lo hiciéramos cada vez que el componente se renderizara se invocaría la función.

<elemento onClick={this.handleEvento} />

Componentes de clase

Eventos y bind

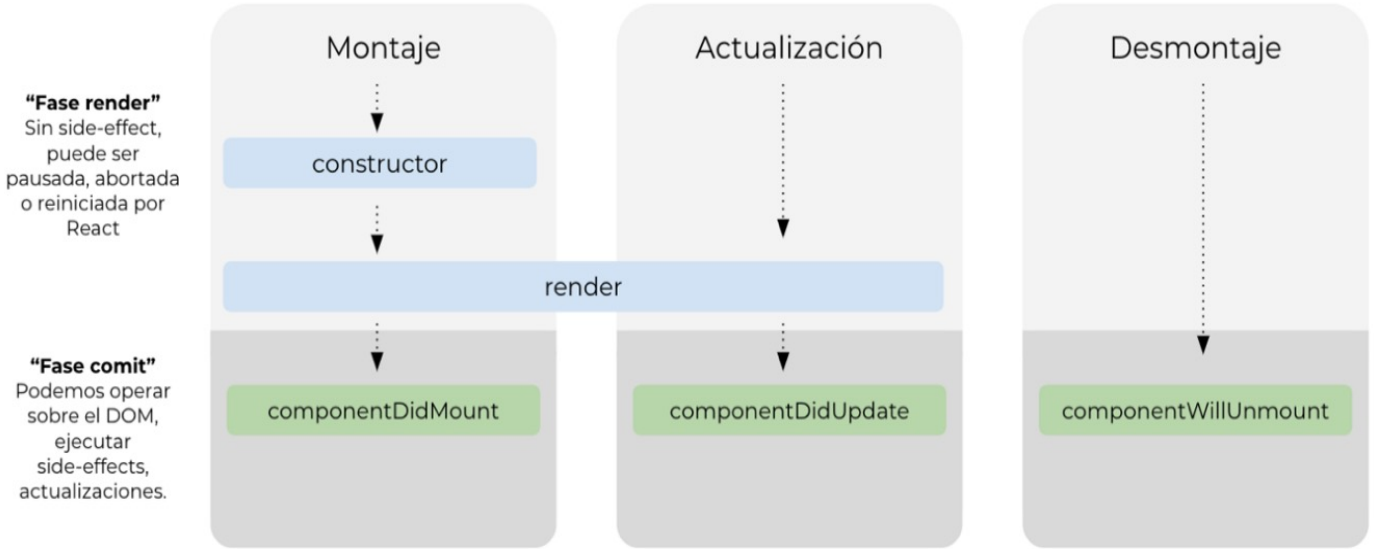
¿Cómo usamos entonces parámetros en esos métodos manejadores cuando lo necesitemos? Usando una callback en la invocación.

handleEvento(parametro) {...}

<elemento onClick={() => this.handleEvento(argumento)} />

Componentes de clase

Ciclo de vida



Componentes de clase

Ciclo de vida

Se maneja con tres métodos denominados ”hooks”:

componentDidMount()

componentDidUpdate(prevProps, prevState)

componentWillUnmount()