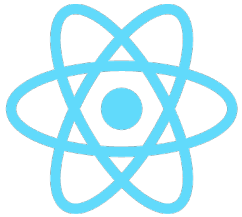
Desarrollo y programación de Aplicaciones con React

6. Patrones avanzados



Patrones avanzados

Comunicación de componentes mediante jerarquía padre-hijo

Paso de datos de componente padre a componente hijo:

- Props
- Manejadores con parámetros

Paso de datos de componente hijo a componente padre

- Invocación de manejadores del padre con envío de los datos en los argumentos.

Patrones avanzados

Comunicación de componentes mediante servicios

Cuando varios componentes comparten datos y, normalmente, además esos datos provienen del exterior de la aplicación, los servicios son los elementos ideales para estructurar la aplicación.

En React serán archivos con una colección de funciones para recibir datos y devolver datos. Las cuales normalmente tendrán como origen y destino externo un servicio API.

Patrones avanzados

Render props

Es una de las formas de estructuración y reutilización de código para los componentes en React, especialmente diseñada para los componentes de tipo clase.

Su sintaxis utiliza un método render en las props con una serie de parámetros correspondientes a las props que podrán ser utilizadas con diferentes implementaciones en los elementos de un componente padre.

Patrones avanzados

Render props

Componente padre

```
<elemento>
 render= (this.props.prop1, this.props.prop2, ...) => { //código }
</elemento>
```

Componente hijo

```
<elemento>
    {this.props.render(this.props.prop1, this.props.prop2, ...)}
<elemento />
```

core networks

Patrones avanzados

High Order Component

HOC es un patrón el cual permite que una función reciba como parámetro un componente y devuelve un componente diferente, mejorado.

Base Component => Enhanced Component

Como los Render props, es un patrón pensado para components de clase.

Patrones avanzados

High Order Component Declaración

```
const withHOC = (Component) =>
        class ComponenteHOC extends React.Component {
            // Lógica del componente
        }
        return ComponenteHOC;
}
```

core networks

Patrones avanzados

High Order Component Invocación

```
class Componente extends Component {

        // invocación con props.métodoManejador

export default withComponente(Componente);
```

Patrones avanzados

Custom Hooks

Un Hook personalizado es una función de JavaScript cuyo nombre comienza con ”use” y que puede llamar a otros Hooks.

A diferencia de un componente de React, un Hook personalizado no necesita tener una firma específica. Podemos decidir lo que adopta como argumentos y lo que deberá devolver.

Patrones avanzados

Custom Hooks Declaración

```
export function useHook(parámetros...) {

    // otros Hooks (normalmente useEffect)

    return {data}
}
```

core networks

Patrones avanzados

Custom Hooks Invocación

```
const {data} = useHook(argumentos);
```

core networks