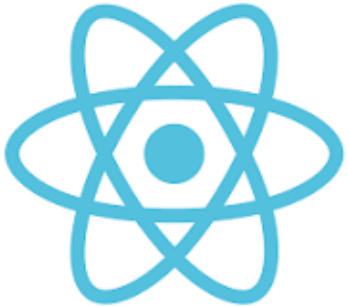
<codoa codo/>

Agencia de Aprendizaje a lo largo de la vida

ESPECIALIZACIÓN REACT



BA

<codoa codo/>

Agencia de Aprendizaje a lo largo de la vida

Clase 05. REACT JS

COMPONENTES 2

BA

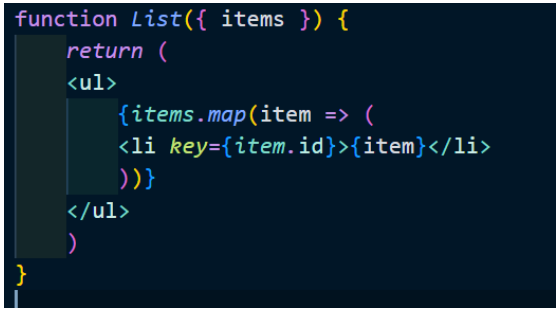
COMPONENTES II

Agencia de Aprendizaje a lo largo de la vida

Buenos Aires Ciudad
Educación

¿Qué es el renderizado de listas en React?

El renderizado de listas es la forma de iterar un array de elementos y renderizar elementos de React para cada uno de ellos.
Para hacer renderizado de listas en React usamos el método map de los arrays:



```
function List({ items }) {
    return (
    <ul>
        {items.map(item => (
        <li key={item.id}>{item}</li>
        ))}
    </ul>
    )
}
```

En este caso, se renderiza una lista de elementos usando el componente List. El componente List recibe una prop items que es un array de strings. El componente List renderiza un elemento li por cada elemento del array.

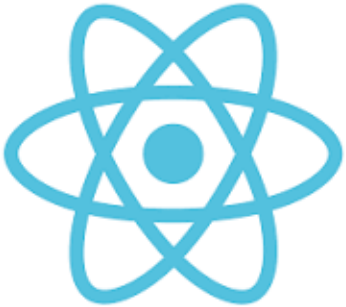
El elemento li tiene una prop key que es un identificador único para cada elemento. Esto es necesario para que React pueda identificar cada elemento de la lista y actualizarlo de forma eficiente. Más adelante hay una explicación más detallada sobre esto.

¿Cómo añadir un evento a un componente en React?

Para añadir un evento a un componente en React usamos la sintaxis con y el nombre del evento nativo del navegador en camelCase:

```
function Button({ text, onClick }) {
    return (
    <button onClick={onClick}>
        {text}
    </button>
    )
}
```

En este caso, el componente Button recibe una prop onClick que es una función. Cuando el usuario hace clic en el botón, se ejecuta la función onClick.



¿Entonces qué correlación hay entre el **render**, las props, el estado y los eventos?

Para saber qué debe renderizar, React busca ciertas condiciones específicas:

Cambio en las props <Title text=”newtext”/>

Cambio en el estado
this.setState({count: 2}) / Class based
setCount(2) / Fn + Hooks

Eventos. Al ocurrir eventos, programáticamente modificaremos el estado, lo cual detona los dos primeros puntos.

HOOKS

Agencia de Aprendizaje a lo largo de la vida

Buenos Aires Ciudad
Educación

¿Qué son los hooks?

Los Hooks son una API de React que nos permite tener estado, y otras características de React, en los componentes creados con una function.

Esto, antes, no era posible y nos obligaba a crear un componente con class para poder acceder a todas las posibilidades de la librería.

Hooks es gancho y, precisamente, lo que hacen, es que te permiten enganchar tus componentes funcionales a todas las características que ofrece React.

¿Qué hace el hook useState?

El hook useState es utilizado para crear variables de estado, quiere decir que su valor es dinámico, que este puede cambiar en el tiempo y eso requiere una re-renderización del componente donde se utiliza

Recibe un parámetro:

El valor inicial de nuestra variable de estado.
Devuelve un array con dos variables:

En primer lugar tenemos la variable que contiene el valor
La siguiente variable es una función set, requiere el nuevo valor del estado, y este modifica el valor de la variable que anteriormente mencionamos
Cabe destacar que la función proporciona cómo parametro el valor actual del propio estado. Ex: setIsOpen(isOpen => !isOpen)
En este ejemplo mostramos como el valor de count se inicializa en 0, y también se renderiza cada vez que el valor es modificado con la función setCount en el evento onClick del button:

¿Qué hace el hook useState?

```jsx
import { useState } from 'react'

function Counter() {
const [count, setCount] = useState(0)

return (
    <>
    <p>Contador: {count}</p>
    <button onClick={() => setCount(count => count + 1)}>Aumentar</button>
    </>
)
}
```

¿Qué hace el hook useEffect?

El hook useEffect se usa para ejecutar código cuando se renderiza el componente o cuando cambian las dependencias del efecto.

Recibe dos parámetros:

La función que se ejecutará al cambiar las dependencias o al renderizar el componente.
Un array de dependencias. Si cambia el valor de alguna dependencia, ejecutará la función.
En este ejemplo mostramos un mensaje en consola cuando carga el componente y cada vez que cambia el valor de count:

¿Qué hace el hook useEffect?

```
import { useEffect, useState } from 'react'

function Counter() {
const [count, setCount] = useState(0)

useEffect(() => {
    console.log('El contador se ha actualizado')
}, [count])

return (
    <>
    <p>Contador: {count}</p>
<button onClick={() => setCount(count + 1)}>Aumentar</button>
    </>
)
}
```