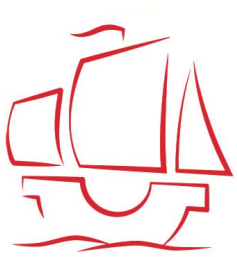


FORMACIÓN PROFESIONAL
MONTECASTELO

React (II)

Desarrollo Web en Entorno Cliente

Ciclo Superior de Desarrollo de Aplicaciones Web

2023/2024

Gestión de eventos

- En React, el manejo de eventos es similar a JS.
- Los eventos en React se nombran usando ***camelCase*** en lugar de minúsculas, y se pasa una función como manejador de eventos.
- Es decir, no se puede añadir la función en un atributo HTML.

Gestión de eventos

- Por ejemplo, para manejar un evento `onClick`, se proporciona una función como `onClick={handleClick}`.
  - No se puede hacer, por ejemplo: `onClick = { handleClick(); }`
- Dentro de la función del manejador de eventos, se puede acceder al objeto de evento del mismo modo que con JS. Por ejemplo: `event.target.value`.

Integración de componentes

- Un componente se puede utilizar de forma independiente cuántas veces se quiera en React.
- De este modo, un mismo componente puede funcionar de forma independiente en varias partes de la aplicación.

Integración de componentes

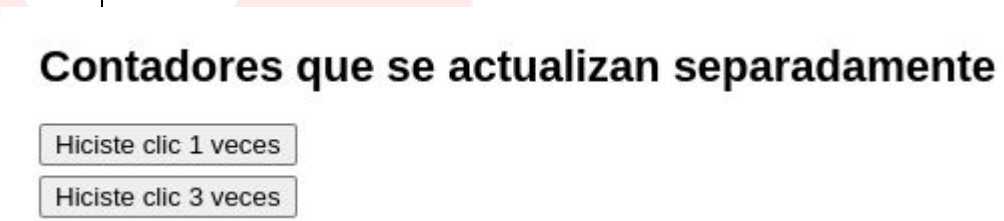
```
import { useState } from 'react';

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

  function handleClick() {
    setCount(count + 1);
  }

  return (
    <button onClick={handleClick}>
      Hiciste clic {count} veces
    </button>
  );
}
export default MyButton;
```

```
function MyApp() {
  return (
    <div>
      <h1>Contadores que se actualizan separadamente</h1>
      <MyButton />
      <MyButton />
    </div>
  );
}
export default MyApp;
```



Integración de componentes

- En el ejemplo anterior, los dos botones son independientes. Pero en muchos casos, se necesita una **comunicación del estado de los componentes** para que compartan datos y se actualicen siempre en conjunto.
- Para hacer se necesita mover el estado de los componentes individuales «hacia arriba» al componente más cercano que los contiene a todos.

Integración de componentes

- Esto se hace convirtiendo los `states` del componente en `props` y que la función que cambia el estado del componente se le pasa por referencia.
- Esta función que modifica el `state`, así como el `state`, estarán ubicados en el componente general

Integración de componentes

```
import { useState } from 'react';
function MyApp() {
  const [count, setCount] = useState(0);

  function handleClick() {
    setCount(count + 1);
  }
  return (
   <div>
  <h1>Contadores que se actualizan separadamente</h1>
      <MyButton count={count} onClick={handleClick}/>
      <MyButton count={count} onClick={handleClick} />
    </div>
  );
}
export default MyButton;
```

```
function MyButton({count, handleClick}) {
  return (
    <button onClick={handleClick}>
      Hiciste clic {count} veces
    </button>
  );
}
export default MyApp;
```

Hooks

- Las funciones que comienzan con use se llaman **Hooks**. `useState` es un Hook nativo dentro de React.
- El resto de Hooks nativos en la referencia de la API de React.
- También puedes escribir tus propios Hooks mediante la combinación de otros existentes.

Hooks

- Los Hooks son más restrictivos que las funciones regulares ya que no puedes llamar a los Hooks en funciones o segundos niveles de los componentes.
- Es decir, si necesitas utilizar `useState` en un condicional o bucle, tienes que crear un componente y ubicar dicho componente en el lugar donde querrías utilizar ese `useState`.

Metodología React

- Aunque el desarrollo es libre, en React es importante seguir una metodología que nos ayude a mantener un orden coherente para organizar y optimizar el código.
- Para ilustrar la metodología, utilizaremos como ejemplo el diseño de una aplicación de lista de tareas.

Metodología React

- Esta lista de tareas, se actualizará dinámicamente en función de un formulario para introducir nuevas tareas o marcarlas como completadas.
- **Paso 0: Boceto de UI**.
  - Se comienza con un boceto, que en frontend, sería diagrama del interfaz de usuario (UI) para visualizar cómo se dividirá en componentes y cómo fluirá la información entre ellos.

Metodología React

- **Paso 1: Separar la UI en una jerarquía de componentes**
  - Cada componente debe representar una parte lógica y reutilizable de la UI (piensa en los componentes como piezas de LEGO que se ensamblan para construir la UI completa).
  - *En el ejemplo, se pueden dividir los siguientes componentes: encabezado, la lista de tareas y el formulario para agregar nuevas tareas*.

Metodología React

- **Paso 2: Construye una versión estática en React**
  - Con la jerarquía de componentes definida, se construye una versión estática del UI sin preocuparse por el estado o la interactividad (sólo jerarquía de componentes mostrando información).
  - *En el ejemplo, se crean los componentes para mostrar la lista de tareas y el encabezado, pero sin la capacidad de agregar o eliminar tareas por ahora.*

Metodología React

- **Paso 3: Representación mínima pero completa del estado de la UI**
  - Se identifican los elementos de la UI que deben ser dinámicos y cambian con el tiempo. Se realiza la interacción de forma más simple posible para renderizar la UI.
  - *En el ejemplo, el estado mínimo pero completo podría ser un array de objetos representando las tareas, donde cada objeto tiene propiedades como* `id`, `texto` *y* `completada`.

Metodología React

- **Paso 4: Decidir en qué componente(s) debe haber un estado.**
  - Los estados de la aplicación deben estar en los componentes cercanos a los elementos que hacen cambiar la UI.
  - *En el ejemplo, el estado podría encontrarse en el componente principal que contiene la lista de tareas y el formulario para agregar nuevas tareas.*

Metodología React

- **Paso 5: Añade flujo de datos inverso.**
  - Finalmente, se establece un flujo de datos inverso para permitir que los componentes hijos actualicen el estado en el componente padre.
  - *En el ejemplo, tener una función en el componente principal que agrega una nueva tarea al estado cuando se envía el formulario desde un componente hijo o cuando se marca completada.*

Metodología React

- **Paso 6: Estiliza la aplicación.**
  - Por último, y una vez que funcionalmente la aplicación está completada, se estiliza la aplicación teniendo en cuenta todos los estados de la UI.

Referencias

- React en web oficial React developers
- React en w3schools

FORMACIÓN PROFESIONAL

MONTECASTELO