**Eventos en React**

Los eventos de usuario son un pilar fundamental en cualquier aplicación front end. A través de ellos, los usuarios podemos interactuar con las capas visuales y de datos de la aplicación. Esto hace que nuestra aplicación sea dinámica y pueda responder en forma casi instantánea a las distintas acciones del usuario.

Como vimos anteriormente, React nos permite manejar los eventos con algunas particularidades. A modo de repaso, podemos mencionar las siguientes:

-Se deben nombrar utilizando camelCase.

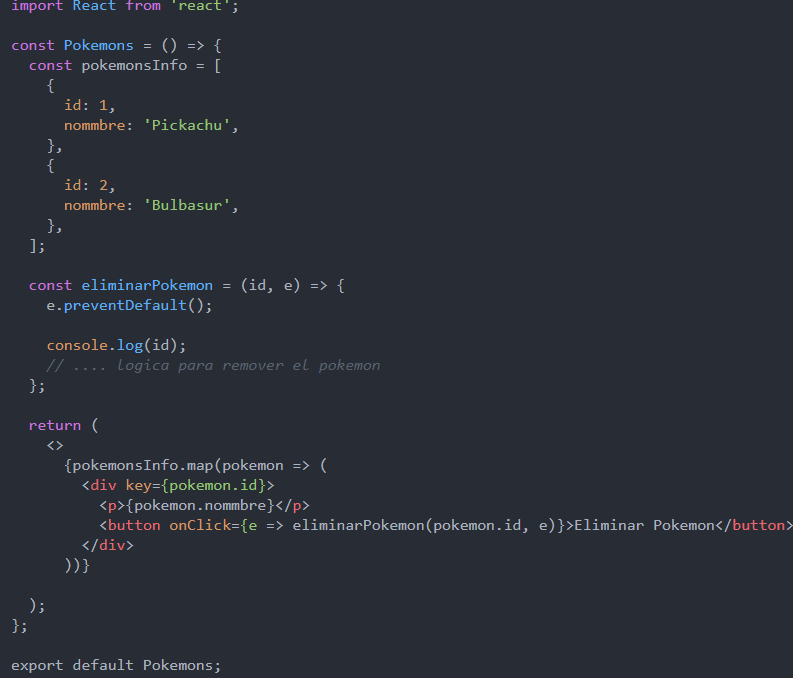
-Se debe pasar un manejador del evento dentro de { }, en vez de un string.

-Debemos utilizar el método preventDefault en forma expresa si deseamos prevenir el comportamiento por defecto. Si utilizamos JavaScript vainilla podríamos simplemente retornar false en nuestro callback.

Dado que React nos permite pasar los manejadores de eventos directamente en el JSX, no será necesario recurrir a los métodos addEventListener y removeEventListener, que empleamos cuando trabajamos con JavaScript vainilla. Será React quien se ocupe de agregar y remover el evento según el elemento se renderice en pantalla o no.

Pasando argumentos

En ciertos casos, puede que sea necesario pasar algún argumento extra a la función que se encargará de manejar nuestro evento. Pensemos en el siguiente ejemplo:

  
  
Aquí tenemos un listado de pokemons, y cada uno de ellos mostrará (además de su nombre), un botón que permitirá eliminarlo de la lista.

Lo lógico es que queramos utilizar el ID del pokémon para poder filtrar la lista y remover el que deseamos. Para ello, necesitamos pasarle el ID como argumento al manejador de eventos.

Pero, dado que nuestro manejador de eventos no es otra cosa que una función, basta con agregar el argumento al momento de declararla y luego pasarlo cuando la empleamos en el JSX.

De esta manera, podremos hacer llegar a nuestro manejador cualquier argumento extra, además del evento, de acuerdo a nuestras necesidades.