CHECKPOINT 14

1. ¿Cuál es el comando para instalar un módulo de nodo?

En algunos ejemplos o ejercicios que hemos realizado anteriormente hemos tenido todo nuestro código en un archivo, pero en los proyectos de la vida real tener todo el código en el mismo lugar no es práctico.

Los módulos son unidades de funcionalidad, es decir, son archivos donde vamos a tener fragmentos de código listos para usar y que podemos llamar desde otras partes de nuestra aplicación.

Node nos permite tener una gran cantidad de archivos que se conectan y comunican unos con otros y cuando el programa corre, todo se une para que el proyecto funcione.

La ventaja de usar este sistema es que nos permite tener diversos archivos con una cantidad de código asequible para entender y organizar en vez de las miles de líneas que deberíamos tener si escribiéramos todo en el mismo archivo.

Node cuenta con un package manager que se llama npm, y es a través de este que realizaremos las acciones de instalar nuevos paquetes, eliminarlos, actualizarlos, etc. Todas las interacciones con Node las haremos a través de la consola.

En la carpeta de nuestro proyecto habrá un documento llamado package.json donde estarán listados todos los módulos que requiera el mismo, y una vez que queramos instalarlos para que nuestra aplicación corra, daremos el comando >npm install en la consola.

Si por el contario quisiéramos instalar solamente un módulo, debemos escribir >npm i (y el nombre del módulo). Aunque lo mejor es mirar en la documentación del paquete que vamos a instalar ya que allí pondrá el comando exacto que debemos dar para instalarlo. Esta documentación se encontrará en la web oficial de npm.

2. ¿Cuáles son los dos tipos de componentes de React?

React es una librería para crear interfaces de usuario, es decir, el frontend, utilizando el lenguaje JavaScrip.

Está basada en componentes, es decir, diferentes elementos individuales que pueden interactuar entre ellos y pueden usarse en lugares diversos. Lo que facilita React es que estos componentes sean unidades aisladas que funcionen por sí solas y combinen el HTML

y JavaScrip en el mismo documento, ya que si se tuvieran el JavaScrip y el código HTML en documentos distintos, y quisiéramos usar un componente en otra parte de nuestra aplicación, esta podría romperse.

Podríamos decir que los componentes en React son piezas de la UI (Interfaz de Usuario) que son reutilizables y para que lo sean deben tener en el mismo paquete la estructura HTML y la lógica JavaScript ya que de esta manera, estando acoplados podemos llevar estas piezas a cualquier parte de la aplicación que deseemos. Actualmente también se suele acoplar el archivo de estilos css en el componente haciéndolo una unidad completamente independiente.

React nos ofrece dos tipos de componentes distintos. Componentes Class y componentes funcionales.

Los componentes Class emplean la programación orientada a objetos. Los objetos son entidades que combinan atributos que representan sus características y propiedades, y métodos que definen las acciones o comportamientos que van a realizar, todo en un solo paquete (u objeto).

En React los componentes Class se crean extendiendo la clase Component en la que se encuentran los métodos necesarios para controlar el estado o “State” y el ciclo de vida o “LifeCycle” de dicho objeto.

Por otro lado tenemos los componentes funcionales. Estos se llaman así puesto que se redactan igual que una función anónima de JavaScript. Estos componentes no nos permiten alterar el estado o acceder al ciclo de vida, pero cuando no tenemos necesidad de realizar estas acciones, estos componentes son más parecidos a la forma en la que trabajamos en JavaScript y por tanto más intuitivos y sencillos.

3. ¿Por qué usamos props en React?

Las props son las propiedades. Estas son un objeto que tiene datos que le vamos a pasar al componente cuando lo imprimimos en pantalla. En React los componentes usan las props para comunicarse entre ellos.

Cada componente parent puede usarlas para pasar información a sus componentes child. Y a estas props podemos pasarles cualquier tipo de valor de JavaScript, como objetos, arrays y funciones.

4. ¿Qué es JSX?

JSX es una extensión del lenguaje de JavaScript que introduce React y que nos permite escribir elementos de HTML, con la propia sintaxis HTML, en JavaScript.

Es decir, en vez de tener que crear los elementos HTML desde el lenguaje de JavaScript teniendo que gastar mucho tiempo y líneas de código para conseguirlo, podemos escribir diréctamente los elementos que queramos en HTML y React por detrás se encarga de escribir todo el código JavaScript necesario para crearlo.

JSX es una herramienta que nos aporta rapidez y eficacia ya que no tenemos que aprender un lenguaje nuevo ni invertir todo el tiempo necesario para aprender a crear cada elemento HTML desde el lenguaje de JavaScript sino que de esta forma esta conversión sucede de manera automática y si poseemos conocimiento básico de JavaScript y HTML ya podemos comenzar a trabajar con React.

5. ¿Qué es el estado?

En una aplicación web podría haber dos o más personas en la misma página, es decir en la misma URL y aun así estar viendo cosas totalmente distintas. Es decir, la página va a responder a cada usuario de manera diferente: un usuario podría estar logueado con la sesión iniciada, otro usuario podría haber agregado productos al carrito, etc.

Por lo tanto, la manera de asegurarnos de que la misma página pueda mostrar diferentes cosas a cada usuario y sobre todo que esa información se mantenga mientras el usuario está cambiado de páginas es el estado.

Por ejemplo, si agregamos algo al carrito pero luego cambiamos a otra página y vamos a inspeccionar otro elemento o volvemos al menú principal, el carrito debe permanecer con ese elemento agregado. Y aunque no estemos mandando esa información al backend hasta que no se haya hecho la compra y se manda la petición, en el frontend el usuario debe ver su cesta de la compra según vaya añaiendo los elementos, etc.

Toda esta información es lo que se llama estado. Este garantiza que cada usuario vea la interfaz según la interacción que él está teniendo con la aplicación y mientras se mueva por la aplicación no se modifique esa información. Sin el estado tendríamos que estar mandando al backend cada pequeña modificación que hiciera el usuario y esperar el tiempo de respuesta haciendo todo el proceso más lento e ineficaz.

En términos técnicos, el estado es un objeto JavaScrip con todos los datos asociados a los componentes con los que el usuario está interactuando.

6. ¿Con qué tipo de componente usamos un constructor?

Un constructor es un método que se llama cuando creamos un objeto a partir de una clase. Este gestiona las tareas de inicialización como establecer ciertas propiedades del objeto por defecto.

En react el constructor nos permite traer las propiedades de otros componentes. Los constructores los usaremos siempre con los componentes Class que son los que trabajan con la programación orientada a objetos.