Guía de buenas prácticas React

1. Organizar la estructura de directorios

En React, no hay un patrón particular para organizar la estructura de carpetas. Pero siempre es mejor organizar la estructura de tu directorio. Puedes estructurar de tal manera que los componentes del contenedor estén presentes en un directorio, los componentes reutilizables en otro, etc., etc., o puedes estructurarlos en función del nombre de los componentes. Esta decisión depende totalmente de tu comodidad. Hay varias formas de organizarse. Busca en Google y elige el enfoque que creas que será más fácil de entender y leer.

2. Mantén tus componentes compactos

Siempre es una buena práctica mantener los componentes pequeños y asegurarse de que los componentes estén construidos de tal manera que sea responsable de alguna funcionalidad. Si hay un fragmento de código que crees que es irrelevante en ese componente y se puede dividir en varios componentes. Puedes dividir el código en varios componentes y luego combinarlos para lograr el mismo resultado que tenías antes. Algunas de las principales ventajas de mantener los componentes pequeños son:

Reutilización: es más fácil reutilizar el código en varios otros componentes.

Mantenimiento: los componentes más grandes son difíciles de mantener en comparación con los más pequeños.

3. No Te Repitas

Esta es una regla común para los programadores, donde siempre es una mejor práctica no volver a escribir tu propio código en otra parte del proyecto. Hay varios problemas que surgen con el código redundante. En primer lugar, el código definitivamente se volverá lento. Además, si tuvieras que hacer algún cambio, tendrás que cambiar en todos los lugares donde se ha repetido, y si te olvidas de modificar en cualquier lugar, se producirán problemas. Cuando la aplicación crezca, puede convertirse en un dolor de cabeza. Por lo tanto, siempre es mejor escribir código común separado en esos escenarios. En React, puedes crear componentes comunes, de modo que pueda reutilizarse en cualquier lugar del proyecto. Una de las formas de lograr este principio en todos los componentes es utilizar el concepto de componente de orden superior.

4. Lógica de representación y lógica de estado independientes

Mantén tu lógica de carga de datos con estado separada de su lógica de renderizado. Es mejor tener un componente con estado para cargar los datos y otro componente sin estado para mostrar esos datos. Esto reduce la complejidad de los componentes. Por ejemplo, si tu aplicación está obteniendo algunos datos, podemos administrar los datos en el componente principal y pasar la lógica de representación a un subcomponente pasando los datos obtenidos como accesorios. De esta manera, el código se ve ordenado y fácilmente comprensible.

5. Utiliza PropTypes (o TypeScript) para la verificación de tipos

PropTypes es una forma de verificación de tipos utilizada en React. Sin PropTypes o alguna forma de verificación de tipos, corremos el riesgo de pasar un tipo de datos incorrecto a un componente, lo que podría causar algún comportamiento inesperado en la aplicación. Por lo tanto, al usar PropTypes o TypeScript, puedes evitar fallos inesperados en su aplicación.

6. Pon CSS en JavaScript

Cuando comienzas a trabajar en un proyecto, es una práctica común mantener todos los estilos CSS en un solo archivo SCSS. El uso de un prefijo global evita posibles colisiones de nombres. Sin embargo, cuando tu proyecto se amplíe, es posible que esta solución no sea factible.

Hay muchas bibliotecas que te permiten escribir CSS en JS. EmotionJS 1 y Glamorous 1 son las dos bibliotecas de CSS más populares para JS.

7. Comenta solo cuando sea necesario

Agrega comentarios al código solo cuando sea necesario. Esto no solo está en consonancia con las mejores prácticas de React, sino que también tiene dos propósitos al mismo tiempo:

Mantendrá el código visualmente libre de desorden.

Evitará un posible conflicto entre el comentario y el código si modificas el código en algún momento posterior.

8. Utiliza mayúsculas para los nombres de los componentes

Si, como la mayoría de la gente, estás utilizando JSX (una extensión de JavaScript), los nombres de los componentes que crees deben comenzar con letras mayúsculas. Por ejemplo, nombra los componentes como SeleccionarBoton en lugar de seleccionarBoton, o Menu en lugar de menu. Hacemos esto para que JSX pueda identificarlos de manera diferente a las etiquetas HTML predeterminadas.

Las versiones anteriores de React mantuvieron una lista de todos los nombres integrados para diferenciarlos de los nombres personalizados. Pero como la lista necesitaba una actualización constante, se eliminó y las mayúsculas se convirtieron en la norma.

En caso de que JSX no sea tu lengage de elección, puedes usar letras minúsculas. Sin embargo, esto puede reducir la reutilización de componentes más allá de tu proyecto.

9. El código debe ejecutarse como se espera y ser comprobable

Realmente, esta regla no necesita explicación. El código que escribas debe comportarse como se espera y ser comprobable fácil y rápidamente. Es una buena práctica nombrar tus archivos de prueba idénticos a los archivos de origen con un sufijo .test. Entonces será fácil encontrar los archivos de prueba.