PRIMER TALLER REACT

WENDY MAYLIN DONNEYS

MUJERES DIGITALES 2024

GRUPO 1

DESARROLLO WEB FRONTEND REACT & REDUX

INSTRUCTOR SANTIAGO GIRALDO

3 DE SEPTIEMBRE 2024

¿Qué es React.js? Un Vistazo a la Popular Biblioteca de JavaScript

La industria del desarrollo web ha experimentado un crecimiento constante en los últimos años; y a medida que este crecimiento continúa, surgen constantemente nuevas tecnologías para ayudar a los desarrolladores a crear sitios y aplicaciones web fáciles de usar.

A lo largo de los años, hemos visto cómo los lenguajes de programación web producen características adicionales, cómo se utilizan más lenguajes de programación para crear tecnologías web, e incluso cómo se construyen frameworks y bibliotecas sobre las estructuras de las tecnologías existentes.

En este tutorial, hablaremos de React — la biblioteca de JavaScript basada en componentes más popular utilizada para crear interfaces de usuario. Aunque este tutorial será totalmente apto para principiantes, también puede servir como guía de referencia para los desarrolladores experimentados de React.

Hablaremos de las características de React, los pros y los contras, por qué deberías usarlo y cómo instalar y usar React. También veremos algunas de las funcionalidades principales de React utilizando ejemplos prácticos de código.

Al final de este tutorial, deberías entender qué es React y cómo funciona, y ser capaz de utilizarlo en la construcción de proyectos increíbles.

¿Qué es React?

React.js, comúnmente llamado simplemente React, es una biblioteca de JavaScript que se utiliza para construir interfaces de usuario. Toda aplicación web React se compone de componentes reutilizables que conforman partes de la interfaz de usuario — podemos tener un componente distinto para nuestra barra de navegación, otro para el pie de página, otro para el contenido principal, etc. Entenderás mejor esto cuando lleguemos a la sección en la que tenemos que trabajar con componentes.

Tener estos componentes reutilizables facilita el desarrollo porque no tenemos que repetir el código reiterativo. Sólo tendríamos que crear su lógica e importar el componente en cualquier parte del código donde se necesite.

React también es una aplicación de una sola página. Por tanto, en lugar de enviar una petición al servidor cada vez que hay que renderizar una nueva página, el contenido de la página se carga directamente desde los componentes de React. Esto conduce a una renderización más rápida sin recargas de la página.

En la mayoría de los casos, la sintaxis utilizada para construir aplicaciones React se llama JSX (JavaScript XML), que es una extensión de la sintaxis de JavaScript. Esto nos permite combinar la lógica de JavaScript y la lógica de la interfaz de usuario de una manera única. Con JSX, eliminamos la necesidad de interactuar con el DOM utilizando métodos como document.getElementById, querySelector, y otros métodos de manipulación del DOM.

Aunque el uso de JSX no es obligatorio, facilita el desarrollo de aplicaciones React.

¿Por qué React?

Muchos desarrolladores y organizaciones han elegido React en lugar de otras bibliotecas/frameworks; veamos el motivo a continuación:

- **Fácil de aprender:** React es fácil de aprender y comprender siempre que se conozcan bien los requisitos previos. React tiene una sólida documentación y muchos recursos gratuitos online. creados por otros desarrolladores a través de la muy activa comunidad de React.
- **Componentes reutilizables:** Cada componente en React tiene su propia lógica que puede reutilizarse en cualquier parte de la aplicación. Esto reduce la necesidad de reescribir el mismo trozo de código varias veces.
- **Oportunidades de trabajo:** Un mayor porcentaje de oportunidades de desarrollo web frontend en este momento tienen React como una de las habilidades requeridas. Por lo tanto, entender cómo funciona React y ser capaz de trabajar con él aumenta tus posibilidades de conseguir un trabajo.
- **Mejora del rendimiento:** Con el DOM virtual de React, la renderización de las páginas puede hacerse más rápidamente. Al utilizar una biblioteca de enrutamiento como React Router, tendrás diferentes páginas renderizadas sin necesidad de recargar la página.
- **Ampliamente extensible:** React es una librería que sólo renderiza la UI de nuestra aplicación. Depende del desarrollador elegir con qué herramientas trabajar, como bibliotecas para renderizar diferentes páginas, bibliotecas de diseño, etc.

¿Quién utiliza React?

React ha sido utilizado por muchos ingenieros de front-end tanto en startups como en empresas establecidas como Facebook, Netflix, Instagram, Yahoo, Uber, The New York Times, y más.

Aunque todas las empresas mencionadas no construyeron todo su producto con React, algunas de sus páginas se construyeron con React. Esto se debe al alto rendimiento, la facilidad de uso y la escalabilidad de React.

Casos de uso de React

React se utiliza generalmente para construir la interfaz de usuario (frontend) de las aplicaciones web. Ofrece una rápida renderización de las páginas y un mayor rendimiento. React puede utilizarse para construir cualquier producto que se ejecute en la web.

Éstas son algunas de las cosas para las que se suele utilizar React:

- Aplicaciones de reproducción de música
- Aplicaciones de redes sociales
- Aplicaciones de chat en tiempo real
- Aplicaciones web Full-stack
- Aplicaciones de Comercio Electrónico
- Aplicaciones meteorológicas
- Aplicaciones de listas de tareas

Características de React

React tiene una plétora de características increíbles que siguen haciendo que sea una opción muy popular para los desarrolladores.

Estas son algunas de las características principales de React:

- **JSX:** Es una extensión de la sintaxis de JavaScript que amplía las características de ES6 (ECMAScript 2015). Nos permite combinar la lógica y el marcado de JavaScript en un componente.
- **DOM virtual:** Se trata de una copia del objeto DOM que primero actualiza y vuelve a renderizar nuestras páginas cuando se realizan cambios; luego compara el estado actual con el DOM original para mantenerlo sincronizado con los cambios. Esto hace que la renderización de las páginas sea más rápida.
- **Componentes:** Las aplicaciones React están formadas por diferentes componentes reutilizables que tienen su propia lógica e interfaz de usuario. Esto hace que sea eficiente para escalar aplicaciones y mantener un alto rendimiento porque no se duplica el código tan a menudo como en otros frameworks.

Pros y Contras de React

Puede que React sea una herramienta muy popular para construir nuestra interfaz de usuario, pero todavía hay razones por las que algunos desarrolladores o principiantes deciden no utilizarla.

En esta sección, hablaremos de las ventajas y desventajas de React.

Estos son los pros de usar React:

1. React es fácil de aprender y entender.
2. React tiene una comunidad muy activa en la que puedes contribuir y obtener ayuda cuando la necesites.

3. Hay muchas oportunidades de trabajo para los desarrolladores de React.
4. React ofrece un mayor rendimiento de las aplicaciones.

Estos son algunos de los contras de usar React:

1. Los principiantes que no tengan un conocimiento sólido de JavaScript (especialmente de ES6) pueden tener dificultades para entender React.
2. React viene sin algunas funcionalidades comunes como la gestión de estado único y el enrutamiento; tendrías que instalar y aprender a usar bibliotecas externas para ellas.

Timeline de React

Para comprender mejor React, echemos un vistazo a los hitos más importantes en la historia de React.

2012: Algo nuevo había comenzado en Facebook. Los anuncios de Facebook se volvieron difíciles de administrar, por lo que Facebook necesitaba encontrar una buena solución. Jordan Walke trabajó en el prototipo y creó React. Durante ese mismo año, Instagram fue adquirida por Facebook, y requería adoptar la nueva tecnología de Facebook. Por esto, Facebook tenía la presión de desvincular React de Facebook y hacerlo de código abierto. Y poco después Mark Zuckerberg en TechCrunch Disrupt San Francisco, hizo las siguientes declaraciones: "Nuestro mayor error fue apostar demasiado por HTML5". Prometió que Facebook brindaría mejores experiencias móviles muy pronto.

2013: El año del Gran Lanzamiento. En mayo, Jordan Walke presentó React en la JS ConfUS. React se volvió de código abierto, aunque el público se mostró escéptico. La mayoría de la gente pensó que React era un gran paso atrás. Esto sucedió porque la mayoría de los "early adopters" asistieron a esta conferencia; sin embargo, React se centró en los "innovadores". Los creadores de React se dieron cuenta de este error a tiempo y decidieron comenzar una gira de React para darlo a conocer. Poco después React (por Facebook) está disponible en JSFiddle y React y JSX disponibles en Ruby on Rails y Python y a final de año David Nolen presenta OM, basado en React. Explica cómo React es increíble, lo que llegó a los primeros usuarios. Este artículo mostró cómo React es mejor que las otras alternativas que existen, lo que impulsó el reconocimiento de React.

Escribe una breve explicación sobre por qué Facebook decidió crear React.

Facebook decidió crear React en 2013 para abordar varios desafíos que enfrentaban al desarrollar aplicaciones web complejas y de gran escala. En ese momento, las aplicaciones web de Facebook, como su interfaz de usuario, eran difíciles de mantener debido a su creciente complejidad y la necesidad de actualizar dinámicamente la interfaz en respuesta a las acciones del usuario.

Uno de los principales problemas era la eficiencia en la actualización de la interfaz de usuario. Las soluciones existentes requerían la manipulación directa del DOM, lo que podía volverse lento y propenso a errores a medida que la aplicación crecía. React fue creado para simplificar este proceso al introducir un enfoque basado en componentes y un "DOM virtual". El DOM virtual permite a React manejar eficientemente las actualizaciones de la interfaz, actualizando solo las partes que han cambiado en lugar de renderizar toda la página.

Además, React promovió un enfoque declarativo para construir interfaces de usuario, lo que facilitó a los desarrolladores describir cómo debería verse la interfaz en cualquier estado dado, en lugar de especificar cómo cambiarla. Esto mejoró la claridad y mantenibilidad del código, lo que fue crucial para aplicaciones grandes y dinámicas como Facebook.

Una single page application (SPA) o aplicación de una sola página, es un tipo de aplicación web que carga una única página inicial desde el servidor y luego actualiza dinámicamente su contenido, mediante las interacciones del usuario, sin necesidad de cargar páginas adicionales del servidor.

Las SPA, en lugar de realizar solicitudes tradicionales al servidor para obtener nuevas páginas HTML, utiliza tecnologías como JavaScript, Ajax y frameworks de frontend como Angular, React o Vue.js para mostrar datos sin necesidad de recargar toda la página.

Su característica principal es que proporciona una experiencia de usuario fluida y rápida, similar a una aplicación de escritorio, pues los cambios de contenido ocurren de forma instantánea y sin interrupciones. Esto se logra mediante la manipulación del DOM (Modelo de Objetos del Documento) y el enrutamiento interno en la app.

Al cargar la página inicial, se descargan los recursos necesarios, como archivos HTML, CSS y JavaScript, con los que se construye y actualiza la interfaz de usuario. Las interacciones de los consumidores, como hacer clic en un enlace o enviar un formulario, activan acciones en JavaScript que actualizan solo las partes relevantes de la página, sin tener que recargar todo el sitio nuevamente.

También, las SPA se emplean en el desarrollo web moderno, debido a su capacidad para crear experiencias de usuario ricas y altamente interactivas y aplicaciones más rápidas y eficientes al reducir la carga en el servidor.

¿Cómo React facilita la creación de una SPA?

1. **Componentes Reutilizables**: React permite dividir la interfaz de usuario en componentes pequeños y reutilizables. Cada componente maneja su propio estado y lógica, lo que facilita el mantenimiento y la escalabilidad de la aplicación.
2. **Virtual DOM**: React utiliza un Virtual DOM para mejorar el rendimiento. Cuando el estado de un componente cambia, React actualiza el Virtual DOM y luego compara las diferencias con el DOM real, aplicando solo los cambios necesarios.
3. **Enrutamiento en el Cliente**: Con bibliotecas como React Router, es posible manejar la navegación dentro de la aplicación sin recargar la página. Esto permite una experiencia de usuario más fluida y rápida.
4. **Gestión del Estado**: React proporciona herramientas para manejar el estado de la aplicación de manera eficiente, como el hook useState y bibliotecas externas como Redux para estados más complejos.

En un proyecto React, las carpetas src y public tienen roles específicos y esenciales:

- src **(source)**: Esta carpeta contiene todo el código fuente de la aplicación. Aquí es donde se desarrollan los componentes de React, los estilos, las imágenes y otros recursos que se utilizan directamente en el código. Básicamente, es el corazón del proyecto donde se escribe la lógica y la estructura de la aplicación[1].
- public: Esta carpeta almacena archivos estáticos que no se procesan por Webpack, como imágenes, fuentes y el archivo index.html. Los archivos en esta carpeta se sirven tal cual al navegador. Es útil para recursos que no necesitan ser transformados o empaquetados durante el proceso de construcción[2].

Las actividades de Scrum

El Sprint Planning

¿Para qué sirve?	¿Quién es responsable?
- **Para establecer la meta del Sprint (Sprint Goal) con el Product Owner** - Para recoger la funcionalidad a desarrollar - **Para planificar en detalle el Sprint** - Para crear las *User Stories* - Para separar las User Story en tareas y determinar el esfuerzo de cada tarea - **Para determinar los criterios de aceptación** - Para resolver dudas	- Los técnicos (autoorganización) - El Product Owner ha de resolver dudas, aclarar la priorización de los ítems del product backog, y consensuar la meta del sprint (Sprint Goal) - El Scrum Master ha de vigilar que se cumplen las normas y que se escoge un volumen de tareas correcto
	¿Quién debe asistir?
	Scrum Team → Developers + Product Owner a la 1ª parte + El Scrum Master
¿Cuál es su TimeBox?	**¿Qué debe pasar después?**
8h para Sprints de 4 semanas	Daily Meeting