TUTORIAL DE REACT

Eric Muñoz Rouillion
Santiago Sáez Caselles

Índice

1. Introducción y características principales

React es una biblioteca de JavaScript desarrollada por Facebook para construir interfaces de usuario. React es eficiente y flexible para construir interfaces de usuario, te permite componer interfaces de usuario complejas a partir de fragmentos de código pequeños y aislados llamados “componentes”

React fue creado por **Jordan Walke**, un ingeniero de software de Facebook, que lanzó un primer prototipo de React llamado “FaxJS”. Fue influenciado por XHP , una biblioteca de componentes HTML para PHP. Se implementó por primera vez en las Noticias de Facebook en 2011 y luego en Instagram en 2012. Se hizo de código abierto en JSConf US en mayo de 2013

Aquí sus características principales:

- Uso de componentes: React permite construir aplicaciones mediante la creación de componentes reutilizables.

- Virtual DOM: React utiliza una representación virtual del DOM para optimizar el rendimiento de la aplicación (El DOM virtual es un concepto de programación donde una representación ideal o “virtual” de la interfaz de usuario se mantiene en memoria y de forma síncrona con el DOM real. Le puedes decir a React en qué estado quieres que esté la interfaz de usuario y se hará cargo de llevar el DOM a ese estado.

- Enfoque en la actualización de estado: React se enfoca en la actualización del estado de la aplicación en lugar de manipular el DOM directamente.

- Sintaxis JSX: React utiliza JSX, una sintaxis que permite escribir código HTML y JavaScript juntos.

- Enfoque en unidireccional de datos: React se basa en un enfoque unidireccional de datos, lo que significa que los datos fluyen de arriba hacia abajo en la jerarquía de componentes.

- En resumen, React es una herramienta popular para construir aplicaciones web interactivas y escalables mediante la utilización de componentes, un enfoque en la actualización del estado y un enfoque unidireccional de datos.

2. Comparativa con Vue

Ambas librerías tienen sus propias ventajas y desventajas, y se utilizan para resolver problemas diferentes.
A continuación os explico las diferencias más notables entre los dos:

- **Tamaño de aplicaciones:** React es utilizado principalmente para crear aplicaciones escalables y de alto rendimiento. React es ampliamente utilizado y tiene una gran comunidad de desarrolladores detrás de él

  Vue, por otro lado, fue creado por un desarrollador independiente y se ha convertido en uno de los frameworks más populares en los últimos años. Vue se enfoca en ser fácil de aprender y utilizar, y se utiliza principalmente para crear aplicaciones de pequeña y mediana escala

- **Enfoque de la librería:** React utiliza un enfoque basado en componentes y una arquitectura de estado para administrar el estado de la aplicación. React también utiliza una librería llamada Redux para administrar el estado global de la aplicación

  Vue, en cambio, utiliza un enfoque basado en componentes y una arquitectura basada en modelos-vistas. Vue utiliza una librería llamada Vuex para administrar el estado global de la aplicación

- **Lenguaje:** React utiliza un lenguaje de programación llamado JSX, que es una extensión de JavaScript que permite escribir código HTML directamente dentro del código JavaScript.

  Vue, por otro lado, lo que hace es utilizar templates basados en HTML para escribir su código

- **Rutas de navegación:** React utiliza un sistema de enrutamiento integrado llamado React Router, que permite a los desarrolladores crear rutas y navegaciones dentro de la aplicación.

  Vue tiene una librería de enrutamiento oficial llamada Vue Router, que permite a los desarrolladores crear rutas y navegaciones dentro de la aplicación

3. ¿Cuándo usar React?

Obviamente existe el tema de preferencias personales y gustos, pero si que hay ocasiones en las que React puede ser una buena alternativa a escoger ya que ofrece diversas ventajas como:

- **Desarrollo rentable multiplataforma** (React nos permite desarrollar aplicaciones multiplataforma, es decir, en lugar de crear 2 veces la misma aplicación por ejemplo para Android y IOS, basta con implementar el mismo código para ambas plataformas)
- **Entrega rápida de proyectos de aplicaciones:** React permite a los desarrolladores de aplicaciones aprovechar varios componentes listos para crear funciones de aplicaciones de forma más rápida y utilizando menos recursos, ya que React requiere menos código en comparación con otras plataformas
- **Función de recarga activa:** La función de recarga activa permite al desarrollador implementar cambios en un código y ver el efecto en la aplicación en tiempo real. Por lo tanto, permite actualizar una aplicación que ya está activa
- **Está compuesto por componentes:** Esto facilita que la aplicación sea más escalable y fácil de mantener ya que los errores sucederán en la propia funcionalidad del componente
- **Integración con Redux:** Podemos agrupar la librería de React con otras librerías como por ejemplo Redux para conseguir mayor funcionalidad y facilitar el desarrollo. Redux y React ambos trabajan con estados. Mientras que cada componente React tiene su propio estado, la función de Redux es emitir actualizaciones de los estados en respuesta a acciones, por lo que estas dos librerías se complementan bien
- **React Native:** Con React tenemos más facilidad para generar aplicaciones móviles utilizando el código de la aplicación web que hayamos creado

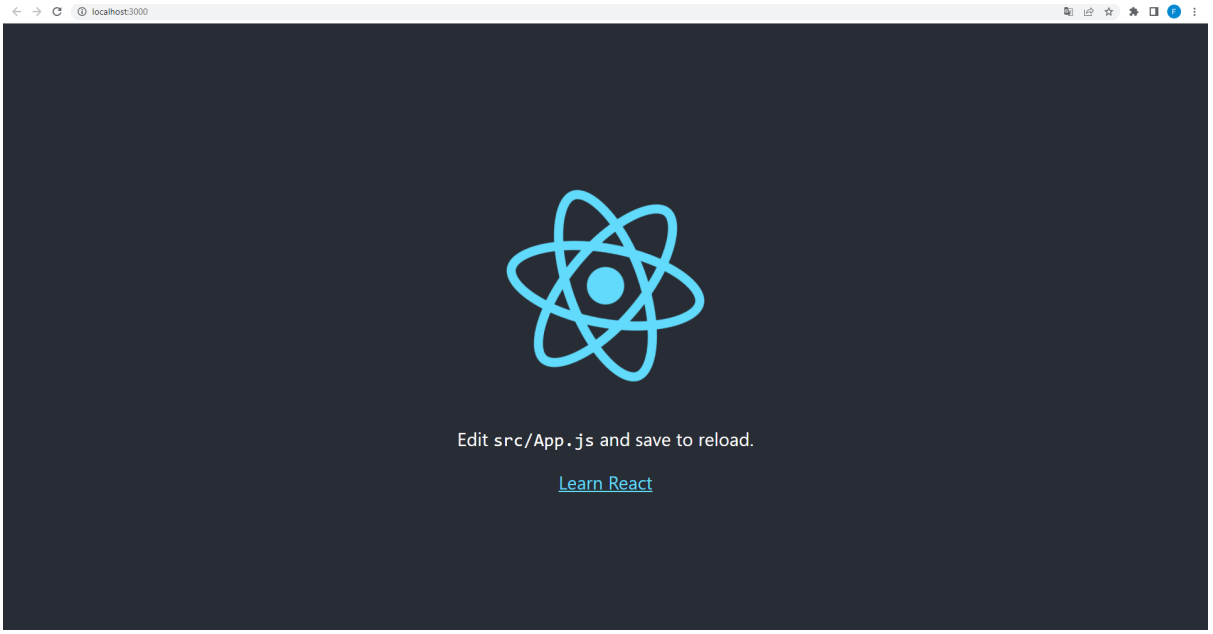
En conclusión React es una librería muy completa aunque en ciertas ocasiones los desarrolladores requieren de más funcionalidades que se pueden suplir con librerías de terceros de forma muy sencilla. Proporciona flexibilidad, rapidez y organización del código lo que hace de React una de las opciones favoritas por los desarrolladores. Tiene muchísimos usuarios lo que hace que no deje de evolucionar para suplir las necesidades de nuevas funcionalidades y características. Hoy en día las aplicaciones más importantes, como puede ser Netflix, Instagram o Facebook utilizan React

4. Tutorial de React

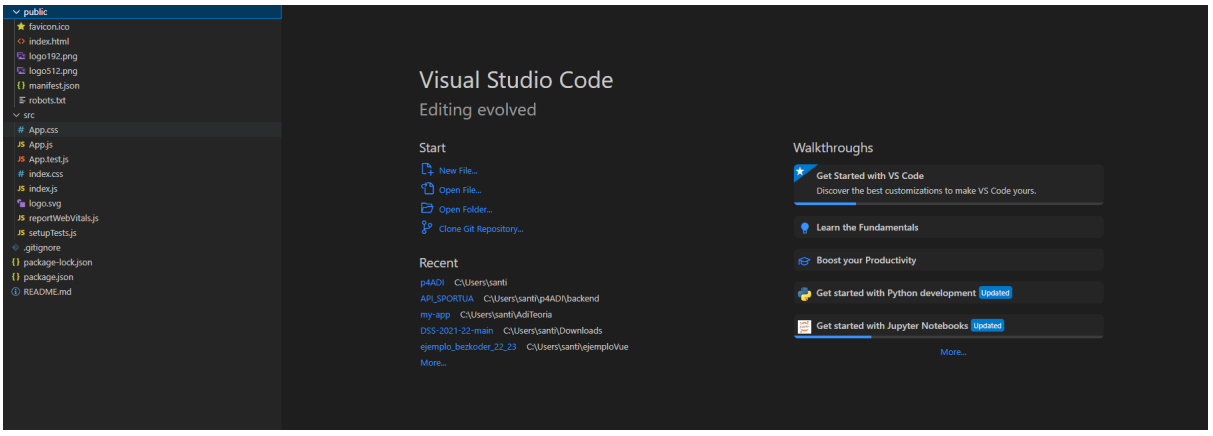
Para empezar a utilizar react debemos tener instalado node.js en nuestra máquina. Para ello se pueden seguir los siguientes pasos ->

1.) Ir a la página oficial de node y descargar el msi
2.) Ejecutar e instalar
3.) En una cmd de windows ejecutar npm -v para comprobar que se ha instalado correctamente

Una vez tengamos instalado npm lo primero que debemos hacer es iniciar el proyecto react desde una terminal con el siguiente comando: “npx create-react-app <nombreproyecto>”, cuando hayamos hecho esto nos movemos a donde se haya creado la carpeta e instalamos npm con “npm install” e iniciamos la aplicación con “npm start”.

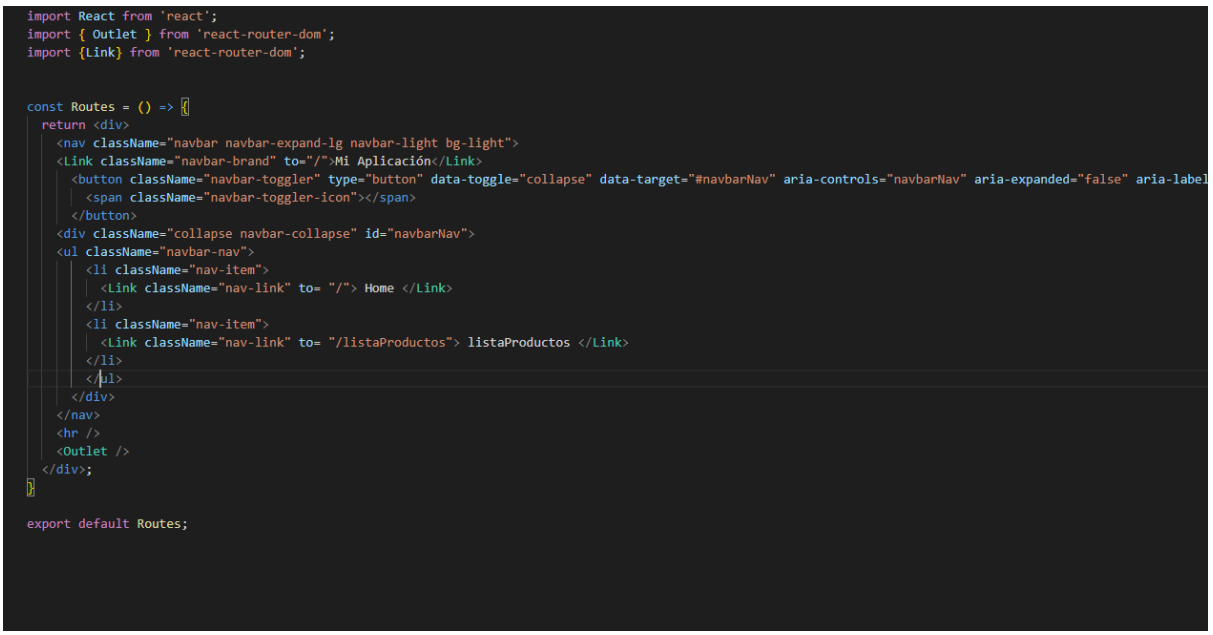


Al iniciar lo primero que nos encontramos es una página de bienvenida. Si miramos la estructura del proyecto nos encontramos lo siguiente:



Tenemos src/App.js que contiene el código de la página principal de la app, este es el código en el que nos podemos basar para crear los diferentes componentes del proyecto que estamos creando.

Ahora bien, vamos a hacer un pequeño ejemplo para ver cómo implementar un listado de productos en la página principal de nuestra app, este listado deberá sacarse de la api "http://localhost:3001/" que es la API de productos de deporte creada para la práctica 1.
Para ello vamos a empezar creando dos componentes.
El primero será ProductList.js, que contendrá la página donde vamos a mostrar estos productos.
Luego de una manera similar a como lo hemos hecho en Vue vamos a crear las rutas. Para ello instalamos el sistema de rutas de react con "npm install react-router-dom", luego crearemos un archivo que en mi caso se llamará "routes.js" e introducimos el siguiente código:



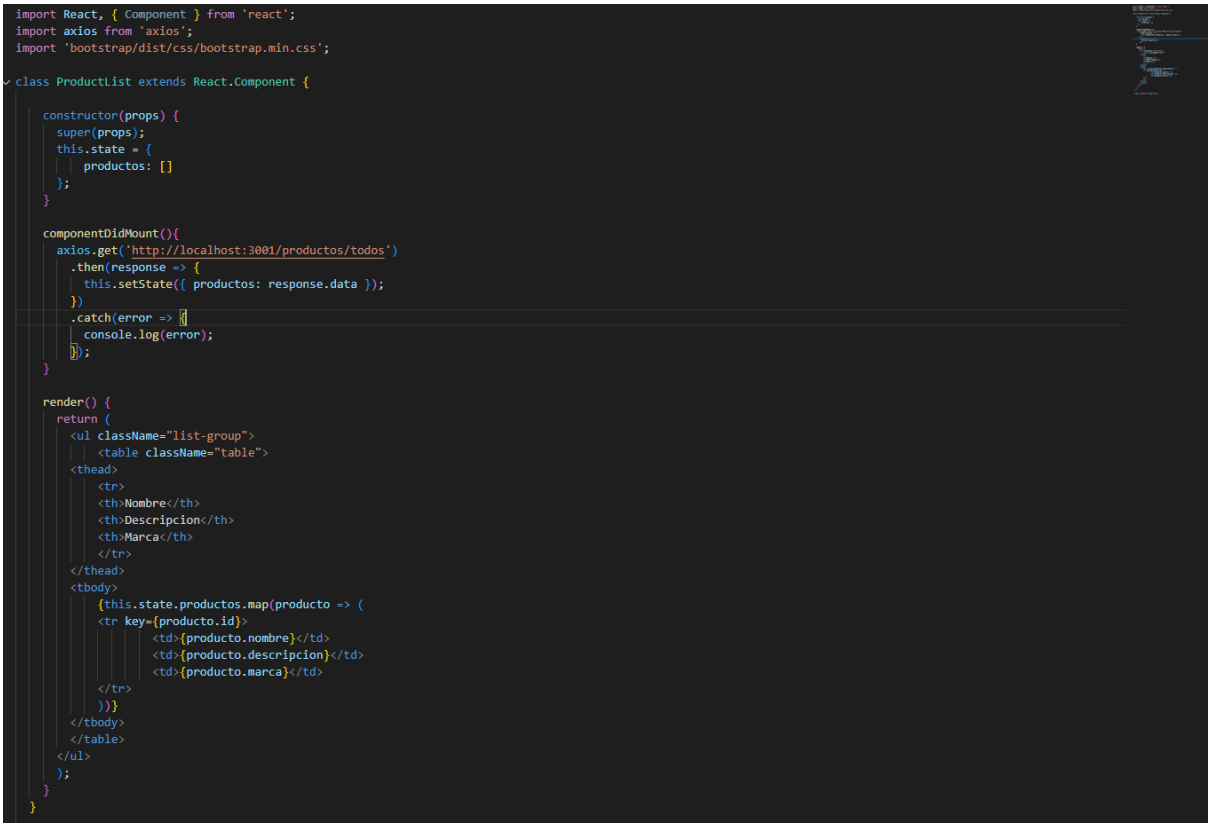
```
import React from 'react';
import { Outlet } from 'react-router-dom';
import {Link} from 'react-router-dom';


const Routes = () => {
  return <div>
    <nav className="navbar navbar-expand-lg navbar-light bg-light">
    <Link className="navbar-brand" to="/">Mi Aplicación</Link>
      <button className="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label
        <span className="navbar-toggler-icon"></span>
      </button>
    <div className="collapse navbar-collapse" id="navbarNav">
    <ul className="navbar-nav">
        <li className="nav-item">
          <Link className="nav-link" to= "/"> Home </Link>
        </li>
        <li className="nav-item">
          <Link className="nav-link" to= "/listaProductos"> listaProductos </Link>
        </li>
        </ul>
      </div>
    </nav>
    <hr />
    <Outlet />
  </div>;
}

export default Routes;
```

Aquí hemos configurado un componente de react que carga un navbar con rutas que introduciremos en la app. Para este ejemplo solo hay dos, una que nos lleva a la página principal(la cual está vacía) y otra que se llama listadoProductos la cual explicaremos más adelante. Recordemos renderizar el componente en index.js introduciendo lo siguiente:

```jsx
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

import { BrowserRouter } from "react-router-dom";


const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <BrowserRouter>
  <App />
  </BrowserRouter>,
);



// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

Cuando nosotros corramos la app veremos el siguiente contenido:

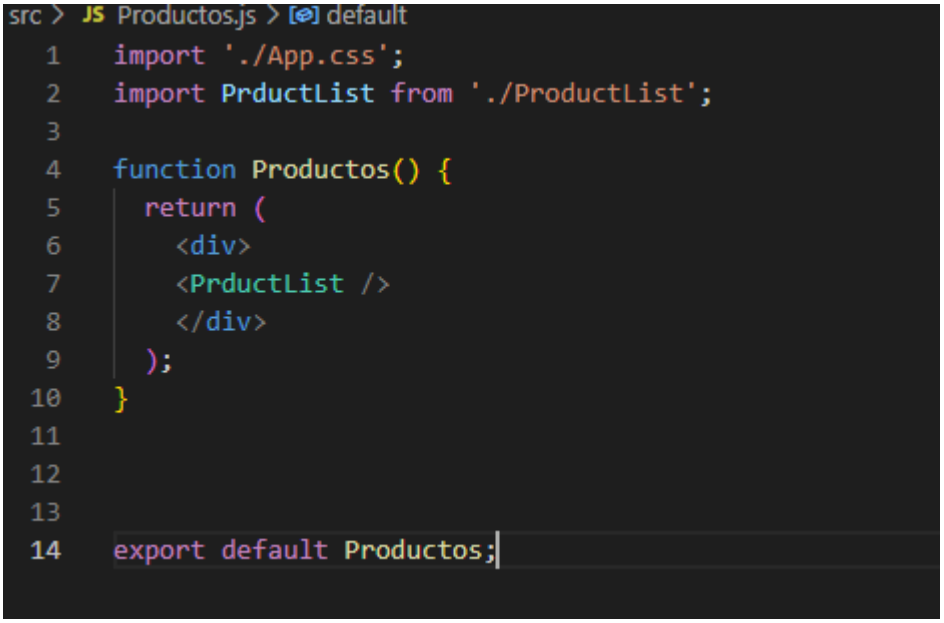
Ahora ya estamos listos para desarrollar el contenido de la página del listado de productos. Lo primero que haremos será crear una clase que contendrá el render que mostrará la lista de productos. Esta clase consumirá nuestra API y nos devolverá un html con el listado de productos.



```
import React, { Component } from 'react';
import axios from 'axios';
import 'bootstrap/dist/css/bootstrap.min.css';

class ProductList extends React.Component {

  constructor(props) {
    super(props);
    this.state = {
      productos: []
    };
  }

  componentDidMount(){
    axios.get('http://localhost:3001/productos/todos')
      .then(response => {
        this.setState({ productos: response.data });
      })
      .catch(error => {
        console.log(error);
      });
  }

  render() {
    return (
      <ul className="list-group">
          <table className="table">
        <thead>
            <tr>
            <th>Nombre</th>
            <th>Descripcion</th>
            <th>Marca</th>
            </tr>
        </thead>
        <tbody>
            {this.state.productos.map(producto => (
            <tr key={producto.id}>
                    <td>{producto.nombre}</td>
                    <td>{producto.descripcion}</td>
                    <td>{producto.marca}</td>
            </tr>
            ))}
        </tbody>
        </table>
      </ul>
    );
  }
}
```

src/ProductList.js

Por último vamos a crear otro archivo en src llamado “Productos.js”, éste cargará el componente y lo mostrará:



```
src > JS Productos.js > default
1   import './App.css';
2   import PrductList from './ProductList';
3
4   function Productos() {
5     return (
6       <div>
7       <PrductList />
8       </div>
9     );
10  }
11
12
13
14  export default Productos;
```

Declaramos las rutas en src/App.js:

```
import './App.css';
import { Routes, Route } from 'react-router-dom';
import Rutas from './routes';
import ListaProductos from './Productos'

function App() {
  return (
  <div>
    <Routes>
      <Route path = "/" element={<Rutas/>}>
        <Route path = "/listaProductos" element={<ListaProductos/>} />
      </Route>
    </Routes>
  </div>
  );
}



export default App;
```

Y una vez hecho esto ya podremos navegar por la aplicación y cambiar la página a la que hemos creado para hacer el listado de productos:

Y listo, con estas directrices ya se puede construir una aplicación con varias rutas donde se puedan lanzar peticiones a las APIS usando React.js.

5. Referencias

https://kinsta.com/es/blog/vue-vs-react/
https://desarrolloweb.com/articulos/que-es-react-motivos-uso.html
https://sistemasgeniales.com/software/las-10-ventajas-principales-de-usar-react-js/
https://www.tithink.com/es/2018/11/14/7-razones-para-utilizar-react/