Memoria Programación Web Cliente

Óscar Corrochano

13 de noviembre de 2024

1. Introducción

En esta documentación se presenta el proyecto de Programación Web Cliente, en el cual se han implementado varios componentes en React. Los componentes desarrollados son los siguientes:

- `App.jsx`
- `App.css`
- `AdvancedSearch.jsx`
- `ComicCarrousel.jsx`
- `FavouriteComic.jsx`
- `Menu.jsx`

2. Objetivos

Los objetivos principales de esta práctica era diseñar una plataforma interactiva en la que se mostraran los cómics más recientemente actualizados de marvel con la posibilidad de almacenar los favoritos y mostrar detalladamente cada cómic.
Personalmente he decidido añadir algunas mejoras para mejorar tanto la experiencia del usuario como mi aprendizaje personal.
Para ello he añadido un buscador avanzado, una sección separada entre cómics favoritos y recientes, la fecha de lanzamiento de cada cómic, filtros para una búsqueda avanzada y la disposición en modo carrousel de los cómics recientes.

3. Componentes

3.1. App.jsx

`App.jsx` es el componente principal de la aplicación.

3.1.1. Importación de componentes

```
1     import { useState, useEffect } from 'react';
2     import './App.css';
3     import Menu from './components/Menu.jsx';
4     import FavoriteComics from './components/FavoriteComics.
      jsx';
5     import ComicCarousel from './components/ComicCarrousel.
      jsx';
6     import AdvancedSearch from './components/AdvancedSearch.
      jsx';
7
```

Listing 1: Código del componente App.jsx

3.1.2. Explicación

Se importan todos los componentes implementados junto con el estilo de la página.

3.1.3. Declaración de variables

```
1     const [vista, setVista] = useState('recent');
2     const [favorites, setFavorites] = useState(() => {
3       const savedFavorties = localStorage.getItem('favorites
      ');
4       return savedFavorties ? JSON.parse(savedFavorties) :
      [];
5     });
6     const [filters, setFilters] = useState({});
7     const [selectedComic, setSelectedComic] = useState(null);
8     const [characters, setCharacters] = useState([]);
9     const [isPopUpVisible, setIsPopUpVisible] = useState(
      false);
10    const [characterIndex, setCharacterIndex] = useState(0);
11
```

Listing 2: Código del componente App.jsx

3.1.4. Explicación

Se declaran todas las variables necesarias en este componente:

- `vista`: vista de la plataforma, favoritos o recientes.
- `favorites`: cómics marcados como favoritos, utilizando localStorage para los cambios de sesión
  - Usa `localStorage.getItem('favorites')` para obtener los favoritos guardados.
  - Si existen favoritos guardados, los convierte de texto a un array de objetos usando `JSON.parse`.
  - Si no hay favoritos guardados, devuelve un array vacío.
- `filters`: filtros de búsqueda aplicados.
- `selectedComic`: cómic seleccionado para mostrarlo en el popup.
- `characters`: carrousel de personajes.
- `isPopUpVisible`: visibilidad del popup.
- `characterIndex`: índice actual del carrousel de personajes.

3.1.5. Funciones

```
1    useEffect(() => {
2      localStorage.setItem('favorites', JSON.stringify(
     favorites));
3    }, [favorites]);
4    const toggleFavorite = (comic) => {
5      if (favorites.some(fav => fav.id === comic.id)) {
6        setFavorites(favorites.filter(fav => fav.id !== comic
     .id));
7      } else {
8        setFavorites([...favorites, comic]);
9      }
10   };
11   const openPopup = (comic) => {
12     const onSaleDate = comic.dates.find(date => date.type
     === 'onsaleDate');
13     const releaseDate = onSaleDate ? new Date(onSaleDate.
     date).toLocaleDateString() : 'Fecha no disponible';
14     setSelectedComic({ ...comic, releaseDate });
15     setIsPopUpVisible(true);
```

```
16      const hash = 'd100e38c62fc4dcaea9841688cac1964';
17      const publicKey = 'bebb3c74579f7d835e2ecf6bd8734e70';
18      const ts = '2';
19      fetch(`http://gateway.marvel.com/v1/public/comics/${
    comic.id}/characters?ts=${ts}&apikey=${publicKey}&hash=${
    hash}`)
20      .then((results) => results.json())
21      .then((data) => {
22        setCharacters(data.data.results)
23        setCharacterIndex(0);
24      })
25      .catch((error) => console.log("Error en el fetch de los
     personajes:", error));
26    };
27    const closePopup = () => {
28      setIsPopUpVisible(false);
29      setSelectedComic(null);
30      setCharacters([]);
31    };
32    const handleNextCharacter = () => {
33      setCharacterIndex((prevIndex) => (prevIndex + 1) %
    characters.length);
34    }
35    const handlePreviousCharacter = () => {
36      setCharacterIndex((prevIndex) => (prevIndex - 1 +
    characters.length) % characters.length);
37    }
38
```

Listing 3: Código del componente App.jsx

3.1.6. Explicación

- `useEffect`:
  - Almacena los cómics favoritos utilizando localstorage.
- `toggleFavorite(comic)`:
  - Gestiona el almacenamiento de los cómics favoritos, si no esta lo añade, si esta lo elimina
- `openPopup(comic)`:
  - Muestra los detalles sobre el cómic seleccionado, portada, título, fecha de lanzamiento, descripción y carrousel de personajes.

- `closePopup`:
  - Cierra el popup.
- `handleNextCharacter`:
  - Avanzar al siguiente personaje del carrousel
- `handlePreviousCharacter`:
  - Retroceder al personaje anterior

3.1.7. Código html

```
1    return (
2    <div id="container"> {/*Contenedor padre*/}
3    <div className="menu-advanced-container"> {/*Menu
     recientes(por defecto)/favoritos y a adir filtros*/}
4    <Menu setVista={setVista} vista={vista} />
5    {vista === 'recent' && <AdvancedSearch setFilters={
     setFilters} />}
6    </div>
7    <div className="content"> {/*Contenido dependiendo de la
     vista con las funciones respectivas*/}
8    {vista === 'recent' && <ComicCarousel favorites={
     favorites} toggleFavorite={toggleFavorite} filters={
     filters} openPopup={openPopup} />}
9    {vista === 'favorites' && <FavoriteComics favorites={
     favorites} toggleFavorite={toggleFavorite} openPopup={
     openPopup} />}
10   </div>
11   {/*Cuando se seleciona un comic y se abre el popup*/}
12   {isPopUpVisible && selectedComic && (
13     <div className="popup"> {/*Padre del popup*/}
14     <div className="popup-content">
15     <button className="close-button" onClick={closePopup}>X
     </button>
16     <div className="popup-header"> {/*Imagen y t tulo*/}
17     <img
18     //obtener la ruta y luego la extension de la imagen
19     src={`${selectedComic.thumbnail.path}.${selectedComic.
     thumbnail.extension}`}
20     alt={selectedComic.title}
21     className="popup-comic-image"
22     />
23     <h2 className="popup-title">{selectedComic.title}</h2>
24     </div>
```

```
25      <p className="popup-release-date">{selectedComic.
    releaseDate}</p> {/*Fecha de lanzamiento*/}
26      <p className="popup-description">{selectedComic.
    description || "No hay descripci n disponible"}</p> {/*
    Descripci n*/}
27      <h3>Personajes:</h3>
28      {characters.length > 0 ? ( //si hay m s de 0
    personajes
29        <div className="character-carousel">
30        <button className="carousel-arrow" onClick={
    handlePreviousCharacter}>   </button> {/*Despalzamiento
    izquierda*/}
31        <div className="character-image-container">
32        <img
33        src={`${characters[characterIndex].thumbnail.path}.${
    characters[characterIndex].thumbnail.extension}`}
34        alt={characters[characterIndex].name}
35        className="character-image"
36        />
37        <p>{characters[characterIndex].name}</p>
38        </div>
39        <button className="carousel-arrow" onClick={
    handleNextCharacter}>   </button> {/*Desplazamiento
    derecha*/}
40        </div>
41        ) : (
42        <p>No hay personajes disponibles</p> //Si no, mostrar
     que no hay
43        )}
44      </div>
45      </div>
46      )}
47    </div>
48    );
49
```

Listing 4: Código del componente App.jsx

3.1.8. Explicación

Este es el código que estará visible en la pantalla.
La estructura es la siguiente:
Contenedor padre
-Contenedor Menú
-Menú y búsqueda avanzada
-Contenedor contenido (carrousel o favoritos)

Si se abre el popup su estructura es:
-Contenedor padre
-Botón en la esquina derecha
-Imagen y titulo
-Fecha de lanzamiento
-Descripción
-Carrousel de personajes

3.2. ComicCarrousel.jsx

`ComicCarrousel.jsx` componente para hacer el carrousel de comics y la logica

3.2.1. Variables

```
1     const [comics, setComics] = useState([]);
2     const [currentIndex, setCurrentIndex] = useState(2); //
       ndice  para el carrousel, empezar en el tercero
3     const [isLoading, setIsLoading] = useState(false); //
     Mostrar mensaje de carga
4     const hash = 'd100e38c62fc4dcaea9841688cac1964';
5     const publicKey = 'bebb3c74579f7d835e2ecf6bd8734e70';
6     const ts = '2';
7
```

Listing 5: Código del componente ComicCarrousel.jsx

3.2.2. Explicación

- `comics`: lista de cómics obtenidos de la API.
- `currentIndex`: índice de los cómics mostrados en el carrousel
- `isLoading`: indica si los datos están cargando desde la API
- `hash`: solicitud a la API de Marvel.
- `publicKey`: clave pública utilizada para acceder a la API de Marvel.
- `ts`: marca de tiempo (*timestamp*) requerida por la API de Marvel para verificar la autenticidad de la solicitud.

3.2.3. Funciones

```
1     const generarApiUrl = () => {
2       let url = `http://gateway.marvel.com/v1/public/comics?
    ts=${ts}&apikey=${publicKey}&hash=${hash}&orderBy=-
    modified`;
3       if(filters.year){
4         url+=`&startYear=${filters.year}`;
5       }
6       if(filters.genre){
7         url+=`&format=${filters.genre}`;
8       }
9       if(filters.characterId){
10        url+=`&characters=${filters.characterId}`;
11      }
12      return url;
13    }
14    useEffect(() => {
15      setIsLoading(true);
16      fetch(generarApiUrl())
17      .then((results) => results.json())
18      .then((data) => {
19        setComics(data.data.results);
20        setIsLoading(false);
21      })
22      .catch((error) => {
23        console.log("Error en el fetch de los comics:", error
    );
24        setIsLoading(false);
25      });
26    }, [filters]);
27    const handleMoveLeft = () => {
28      setCurrentIndex((prevIndex) => {return (prevIndex - 1 +
     comics.length) % comics.length;});
29    };
30    const handleMoveRight = () => {
31      setCurrentIndex((prevIndex) => {return (prevIndex + 1)
    % comics.length;});
32    };
33    const getCircularIndex = (index) => {
34      return (index + comics.length) % comics.length;
35    }
36
```

Listing 6: Código del componente ComicCarrousel.jsx

3.2.4. Explicación

- `generarApiUrl`: función para generar una url de la api de marvel específica en función de los filtros aplicados.
- `useEffect`: Cargar los cómics
- `handleMoveLeft`: Navegar hacia el cómic anterior
- `handleMoveRight`: Navegar al siguiente cómic
- `getCircularIndex(index)`: Esta función permite que cuando se llegue al final de la lista de comics se vuelva a empezar por el otro lado de forma circular

3.2.5. Código html

```
1  <div className="carousel-container"> {/*Contenedor padre carrousel*/}
2  {isLoading && <div className="loading-spinner">Cargando...</div>} {/*Mostrar mensaje de carga*/}
3  <button onClick={handleMoveLeft} className="carousel-button">  </button> {/*Desplazamiento*/}
4  <div className="carousel"> {/*Carrousel de c mics*/}
5  {comics.length > 0 && [...Array(5)].map((_, i) => { {/*Array temporal de 5 comics e iteraci n sobre todos*/}
6      const comicIndex = getCircularIndex(currentIndex - 2 + i);
7      const comic = comics[comicIndex];
8      const isCenter = i === 2; //Centro de los 5 c mics
9      return (
10     //Renderizar cada comic, si es el central se le a ade la clase center, sino side
11     <div
12     key={comic.id}
13     className={`carousel-item ${isCenter ? 'center' : 'side'}`}
14     style={{ position: "relative" }}
15     onClick={() => openPopup(comic)}
16     >
17     {/*Renderizar imagen del comic, con filtro opaco si no es el centrar*/}
18     <img
19     src={`${comic.thumbnail.path}.${comic.thumbnail.extension}`}
20     alt={comic.title}
21     className="comic-image"
```

```
22          style={{ filter: isCenter ? 'none' : 'brightness(0.5)
    ' }}
23          />
24          {/*Icono de favorito*/}
25          <span
26          className="favorite-icon"
27          onClick={(e) => {
28              e.stopPropagation(); //Evita openpopup
29              toggleFavorite(comic);
30          }}
31          //Cambiar color si es favorito
32          style={{ color: favorites.some(fav => fav.id ===
    comic.id) ? '#ffcc00' : '#ffffff' }}
33          >   </span>
34          {/*Si es el central mostrar titulo*/}
35          {isCenter && <h2 className="comic-title">{comic.title
    }</h2>}
36          </div>
37          );
38        })
39       }
40       </div>
41       <button onClick={handleMoveRight} className="carousel-
    button">   </button>
42       </div>
43
```

Listing 7: Código del componente ComicCarrousel.jsx

3.2.6. Explicación

Este es el código html que devuelve el componente.
La estructura es la siguiente:
-Contenedor padre
-Si está cargando mostrar mensaje de carga
-Carrousel de cómics con la portada y el título en el centro
-Estrella para añadir a favoritos en cada cómic
-Botones para desplazarse entre los cómics

3.3. FavoriteComics.jsx

`ComicCarrousel.jsx` componente en el que se muestran los cómics marcados como favoritos

3.3.1. Código html

```
1    <div className="contenedor-principal"> {/*Contenedor
     padre*/}
2    <h1 className="titulo">Comics Favoritos</h1>
3    <div className="grid-container">
4    {favorites.length > 0 ? ( //Si hay favoritos renderiza
     cada uno
5      favorites.map((comic) => (
6      <div key={comic.id} className="comic-card" onClick={()
     => openPopup(comic)}>
7      <div className="comic-content"> {/*Contenedor de cada
     comic*/}
8      <img
9      src={`${comic.thumbnail.path}.${comic.thumbnail.
     extension}`}
10     alt={comic.title}
11     className="comic-image"
12     />
13     <div className="comic-info">
14     <h2 className="comic-title">{comic.title}</h2>
15     <button className="remove-favorite-button" onClick={(e)
     => {
16         e.stopPropagation(); //Evita openpopup
17         toggleFavorite(comic);
18     }}>
19     Quitar de favoritos
20     </button>
21     </div>
22     </div>
23     </div>
24     ))
25     ) : (
26     <p>No hay comics favoritos</p>
27     )}
28   </div>
29   </div>
30
```

Listing 8: Código del componente FavoriteComics.jsx

3.3.2. Explicación

Este es el código html que devuelve el componente
La estructura es la siguiente:
-Contenedor padre
-Titulo
-Contenedor de cómics
-Contenedor de cada cómic que se haya añadido a favoritos
-En cada contenedor aparece la portada, titulo y un botón para eliminar de favoritos

3.4. AdvancedSearch.jsx

`ComicCarrousel.jsx` este componente implementa la búsqueda avanzada mediante filtros

3.4.1. Declaración de variables

```
    const [year, setYear] = useState('');
    const [genre, setGenre] = useState('');
    const [character, setCharacter] = useState('');

```

Listing 9: Código del componente AdvancedSearch.jsx

3.4.2. Explicación

Cada variable representa cada uno de los filtros posibles, año, género y personaje.

3.4.3. Funciones

```
    const handleSearch = () => {
      setFilters({year, genre, characterId: character});
    };

```

Listing 10: Código del componente AdvancedSearch.jsx

3.4.4. Explicación

Con esta función se actualizan los filtros y permite a otros componentes acceder a estos

3.4.5. Código html

```
    <div className="advanced-search">
    <input //Filtro de a o
    type="number"
    placeholder="A o"
    value={year}
    onChange={(e) => setYear(e.target.value)}
    />
    <select value={genre} onChange={(e) => setGenre(e.target.
   value)}> {/*Seleccionar tipo*/}
    <option value="">Seleccionar g nero</option>
    <option value="comic">C mic</option>
    <option value="digital comic">C mic digital</option>
    <option value="hardcover">Hardcover</option>
    </select>
    <input //Filtrar por id de personaje que aparece
    type="text"
    placeholder="ID del personaje"
    value={character}
    onChange={(e) => setCharacter(e.target.value)}
    />
    <button onClick={handleSearch}>Buscar</button>
    </div>

```

Listing 11: Código del componente FavoriteComics.jsx

3.4.6. Explicación

Este es el código html que devuelve el componente.
La estructura es la siguiente:
-Contenedor padre
-Input para introducir el filtro del año
-Select para seleccionar el filtro por tipo de cómic
-Input para introducir el id del personaje que se busque en el cómic
-Botón de buscar para aplicar los filtros

3.5. Menu.jsx

`ComicCarrousel.jsx` este componente sirve para alternar entre la vista de comics recientes y favoritos

3.5.1. Código html

```
1     <nav className="menu">
2     <button onClick={() => setVista('recent')} className={`
      boton-menu ${vista === 'recent' ? 'active' : ''}`}>
3     Comics Recientes
4     </button>
5     <button onClick={() => setVista('favorites')} className
      ={`boton-menu ${vista === 'favorites' ? 'active' : ''}`}>
6     Comics Favoritos
7     </button>
8     </nav>
9
```

Listing 12: Código del componente Menu.jsx

3.5.2. Explicación

Simplemente devuelve un navegador con dos botones, uno asociado al carrousel de cómics recientes y el otro a la lista de comics favoritos

3.6. App.css

Este archivo controla el estilo de todos los componentes utilizando CSS. No se incluye el código en ese archivo por mantener orden ya que es muy extenso, no obstante el archivo está comentado con cada funcionalidad.

4. Conclusión

En este proyecto se utiliza React para construir una plataforma interactiva sobre cómics de Marvel a través de varios componentes, obteniendo los datos a través de la propia API de Marvel.
Se muestra un feed de cómics recientes en forma de carrousel con la posibilidad de realizar una búsqueda con filtros por año, género y personajes. Además se puede añadir a favoritos cualquier cómic, y este aparecerá en otra pantalla a la que se puede acceder a través de un menú.
Por último, tanto en la pantalla de cómics recientes como en la de cómics

favoritos se puede acceder a más detalles de cada cómic como su fecha de lanzamiento, descripción y personajes clickando sobre ellos.