ThePower business school

🇪🇸 Mini Proyecto ReactJS [2]

Introducción

Es muy habitual en React componetizar y desacoplar los elementos para así repartir tanto el detalle del elemento como su funcionalidad. Por ello vamos a desgranar en componentes el proyecto anterior.

Desacople y props

Ahora en el fichero `App.jsx` :

```
const [characterList, setCharacterList] = React.useState([]);
```

Finalmente vamos a llevarnos nuestros elementos a un componente llamado `Card.jsx` que recibirá un `character` y desde `App` simplemente llamamos a `Card`.

```
import React from "react";

export const Card = (props) => {
  const { character } = props;

  return (
    <div>
      <h2>id: {character.id}</h2>
      <h3>name: {character.name}</h3>
      <p>status: {character.status}</p>
    </div>
  );
};

export default Card;
```

Y ahora podemos usar este componente en nuestro `App.jsx` .

```
import React from "react";
import "./App.css";
import { Card } from "./Card";

const App = () => {
  const [characterList, setCharacterList] = React.useState([]);

  React.useEffect(() => {
    (async () => {
      let data = await fetch(`https://rickandmortyapi.com/api/character/`).then(
        (res) => res.json()
      );

      setCharacterList(data.results);
    })();
  }, []);

  return (
    <>
      {characterList.map((character) => (
        <Card key={character.id} character={character} />
      ))}
    </>
  );
};

export default App;
```

Pero aún podemos abstraerlo un poco más creando un componente `CharacterList.jsx` que tenga la funcionalidad que teníamos en `App` y así solo invocarlo desde `App.jsx` , de tal modo que nuestro componente `CharacterList.jsx` .

```
import React from "react";
import { Card } from "./Card";

export const CharacterList = () => {
  const [characterList, setCharacterList] = React.useState([]);

  React.useEffect(() => {
    (async () => {
      let data = await fetch(`https://rickandmortyapi.com/api/character/`).then(
        (res) => res.json()
      );
```

```
      setCharacterList(data.results);
    })();
  }, []);

  return (
    <>
      {characterList.map((character) => (
        <Card key={character.id} character={character} />
      ))}
    </>
  );
};
```

Dejando solamente nuestro componente funcional invocado desde el `App.jsx`:

```
import React from "react";
import "./App.css";
import { CharacterList } from "./CharacterList";

const App = () => {
  return <CharacterList />;
};

export default App;
```

Mini-ejercicio

Ha llegado el momento de ponerse a trabajar con ReactJS, para ello os proponemos una pequeña práctica que os ayude afianzar el concepto de children.

1. Crea una aplicación de ReactJS con vite → name: project-components-advanced.
2. Crea tu carpeta de `components` dentro de `src`.
3. Realizamos algunos componentes de ReactJS:
    a. Componente Header ⇒ Crea un componente que reciba como children el componente Title y retorne un `<header> + Children`.
    b. Componente Main ⇒ Crea un componente que reciba como children los algunos componentes creados y retorne un `<main> + Children`.

c. Componente Footer ⇒ Crea un componente que reciba como children los algunos componentes creados y retorne un `<footer> + Children`.

d. Componente CharacterList ⇒ Crea componente listado que genera un `<ul>`

e. Componente ItemList ⇒ Crea un componente que recibe un item y retorna un `<li>` por cada uno de los elementos recibidos.

f. Componente Title ⇒ Crea un componente Title que retorne un `<h1>` con un texto recibido por props.

g. Componente SubTitle ⇒ Crea un componente que retorne un `<h2>` con un texto recibido por props.

h. Componente Image ⇒ Crea un componente que retorne un `<image>` con un src y alt recibido por props || además también recibirá el with y el height.

i. Componente Paragraph ⇒Crea un componente que retorne un `<p>` con un texto recibido por props.

4. Estila cada uno de ellos haciendo uso de CSS Modules → hoja de estilo asociada al componente.

5. Exporta los componentes en un `index.js` e importalos en `App.jsx`.

   a. Composición HTML:

```
<header>
  <h1>Title</h1>
</header>
<main>
  <h2>SubTitle</h2>
  <ul>
    <li>
        <p>Name</p>
        <image>
        <p>Status</p>
        <p>Origin</p>
    </li>
  </ul>
</main>
<footer>
  <p>Created by name</p>
  <image>
  <p>Copyright</p>
</footer>
```

6. Puedes usar los componentes tantas veces como quieras pero siempre bajo la estructura de los componentes `<Header>` || `<Main>` || `<Footer>`.
7. Comprueba que la visualización es correcta.