HW 13: React-Hooks | Integración

Duración estimada 🕒

2 horas

Rick & Morty App

INTRO

En esta homework crearemos dos cosas que harán más completa nuestra aplicación: 😄

- Por un lado, haremos un **filtrado** para nuestros personajes favoritos. Vamos a filtrar todos los personajes por su género. En total hay cuatro géneros:

```
["Male", "Female", "unknown", "Genderless"];
```

- Por otro lado, haremos un **ordenamiento** también para nuestros personajes favoritos. Vamos a ordenar todos los personajes por su id (de mayor a menor y viceversa).

🦍 EJERCICIO 1

ACTIONS

- Dirígete al archivo en el que se encuentran tus **actions**.
- Lo que hay que hacer:

1. Crear una action-creator con el nombre "***filterCards***", esta action-creator debe:

   a. Recibir por parámetro un **gender**.

   b. Retornar un ***type*** llamado "**FILTER**" y un ***payload*** donde su valor sea el parámetro recibido en la action.

2. Crear una segunda action-creator con el nombre "***orderCards***". Esta action-creator debe:

   a. Recibir por parámetro un **id**.

   b. Retornar un ***type*** llamado "**ORDER**", y un ***payload*** donde su valor sea el parámetro recibido en la action.

🦍 EJERCICIO 2

REDUCER

◇ Seguiremos trabajando nuestro reducer, el objetivo de este ejercicio es añadir una nueva propiedad de estado y con la propiedad que ya teníamos creada de la homework anterior (React-Redux), lograremos paso a paso filtrar y ordenar nuestros personajes favoritos.

◇ Lo que hay que hacer:

1. Dirígete al archivo en el que se encuentra tu reducer:

   a. Crea una nueva propiedad de estado llamada ***allCharacters*** a nuestro estado global ***initialState***.

   b. **_allCharacters** debe ser un arreglo vacío.

2. Modifiquemos el caso **ADD_FAV**:

   a. Actualmente tenemos en el return de este case:

   - El state.
   - Una propiedad llamada ***myFavorites*** donde su valor es una copia del estado ***myFavorites*** y el payload.

   ♦ Lo que debes es reemplazar la copia de ***myFavorites*** por una copia del estado creado en el punto anterior: ***allCharacters***.

   b. Debajo de la propiedad ***myFavorites*** agrega la propiedad de estado ***allCharacters*** donde su valor sea una copia de este estado y el payload.

3. Crea un nuevo caso con el nombre "*FILTER*", en él vamos a filtrar nuestros personajes favoritos para ello debes hacer lo siguiente:

   a. Mediante destructuring trae la propiedad de estado "***allCharacters***".

   b. Filtra aquellos personajes que tengan el mismo género que recibes por payload.

   c. Retorna tu estado global y la propiedad ***myFavorites*** ésta última debe ser igual al filtrado que haz hecho en el punto b.

   > **Hint**: Recuerda cuando desarrollamos la homework 08-React-Estado-LifeCycle 01-Exercises: en Zoo app, creamos también copias de estado 😉.

4. Crea un caso con el nombre "*ORDER*", en él vamos a ordenar nuestros personajes favoritos de forma ascendente y descendente, para ello debes hacer los siguientes pasos:

   a. Mediante destructuring trae la propiedad de estado "***allCharacters***".

   b. Utilizar el método **sort** para ordenar tus personajes de acuerdo a su ID.

   c. Si el *payload* es igual a "**Ascendente**", los personajes deben ordenarse de menor a mayor.

   d. Si el *payload* es "**Descendente**, los personajes deben ordenarse de mayor a menor.

e. Retornar tu estado global y la propiedad ***myFavorites***, ésta última debe ser igual al ordenamiento que acabas de hacer.

> **NOTA:** investiga en la web sobre cómo funciona el método sort.

EJERCICIO 3

Filtros y ordenamientos en el componente Favorites

- Dirígete al archivo en el que se encuentra tu componente **Favorites**.
- Lo que hay que hacer:

1. Crea una etiqueta `div`.
2. Dentro del div crea una etiqueta `select` con el atributo **name**, para el ordenamiento, dentro de esta etiqueta:

   a. Crea una etiqueta `option` con el atributo **value**, el valor del atributo debe ser "*Ascendente*" y su texto puede ser *Ascendente*.

   b. Crea una segunda etiqueta **option** con el atributo **value**, el valor del atributo debe ser "*Descendente*" y su texto puede ser *Descendente*.

   Por ejemplo:

   ```
   <option value="Ascendente">Ascendente</option>
   ```

3. Crea una segunda etiqueta **select** con el atributo name, para el filtrado, dentro de esta etiqueta:

   a. Crea 4 etiquetas **option** cada una con su atributo **value** con los siguientes valores: **Male**, **Female**, **Genderless** y **unknown**. Cada texto de cada etiqueta option puede ser igual a cada valor.

   ```
   <option value="Male">Male</option>
   ```

4. Crea una función que reciba como parámetro un evento y despache la acción "**orderCards**" con el hook **useDispatch**; la acción recibe como argumento `e.target.value`.
5. Crea una función que reciba como parámetro un evento y despache la acción "**filterCards**" con el hook **useDispatch**; la acción recibe como argumento **e.target.value**.
6. Agrega el atributo `onChange` a las etiquetas **select** y que su valor sea el nombre de la función correspondiente a cada select.

Extra Credit

Agrega una opción adicional en el select del filtro para que muestre todos los personajes. Desarrolla la lógica para que ello ocurra.

A esta altura, tu filtro y ordenamiento debería estar funcionando de la siguiente manera!