CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

ACTIVIDAD FINAL
PRIMERA EVALUACIÓN

Desarrollo Web en entorno cliente
CFGS DAW

Álvaro Maceda Arranz

alvaro.maceda@ceedcv.es

2021/2022

Versión:220123.1242

Licencia

ACTIVIDAD INTERMEDIA EVALUABLE - 2ª EVALUACIÓN

Revisa los criterios de evaluación para saber que es lo que debes tener en cuenta a la hora de realizar el ejercicio: no es suficiente que el programa cumpla la función, debe cumplir además otros criterios para ser considerado un buen código JavaScript.

1. ENUNCIADO

Crea una función llamada `searchCharacter` que, dado un nombre de personaje de Rick and Morty, devuelva los datos de ese personaje utilizando el API de Rick and Morty:

https://rickandmortyapi.com/

En caso de que haya más de un personaje con ese nombre debes tomar el primero que devuelva la consulta.

Para buscar un personaje puedes utilizar la siguiente URL:

https://rickandmortyapi.com/api/character/?name=[NOMBRE DEL PERSONAJE]

Crea otra función llamada `getCompanions` que admita un nombre de personaje y devuelva de forma asíncrona el id y el nombre de todos los personajes con los que el personaje buscado haya coincidido alguna vez en algún episodio, ordenados alfabéticamente y sin contener al personaje original.

Por ejemplo, si ejecutamos `getCompanions('Armagheadon')` cuando se obtenga el resultado este debe ser:

```
[
  { id: 454, name: 'Arbolian Mentirososian' },
  { id: 4, name: 'Beth Smith' },
  { id: 47, name: 'Birdperson' },
  { id: 115, name: 'Ethan' },
  { id: 124, name: 'Father Bob' },
  ...
  { id: 344, name: 'Tammy Guetermann' },
  { id: 346, name: 'Terry' }
]
```

Debes obtener los datos lo más rápidamente posible. Es decir, lo que debes hacer es:

- Obtener los datos del personaje. En los datos del personaje se encuentra la lista de episodios donde ha participado
- Obtener en paralelo los datos de cada uno de los episodios donde ha participado. En los datos de cada uno de los episodios viene una lista de personajes de ese episodio.
- Tan pronto como tengas la lista de personajes de un episodio debes consultar en paralelo los datos de esos personajes. Ten en cuenta que, si hubieses lanzado ya una petición para obtener los datos de ese personaje, no debes lanzarla de nuevo.

- Una vez tengas los datos de todos los personajes de todos los episodios debes ordenarlos por su nombre alfabéticamente y devolver la lista.
- El personaje original no debe estar en la lista

Ambas funciones han de cumplir lo siguiente:

- Cuando no se encuentre ningún personaje con ese nombre debe devolver una promesa rechazada con el valor: `"Personaje no encontrado"`
- Si la respuesta no tiene status 200 debe devolver una promesa rechazada con el valor `"Error: [STATUS DE LA RESPUESTA]"`, sustituyendo `[STATUS DE LA RESPUESTA]` por el valor devuelto.
- Cuando ocurra algún error de HTTP debe devolver una promesa rechazada con el error devuelto por `fetch`.

2. Entorno de desarrollo

Debes usar una plantilla para el proyecto. Puedes obtenerla en el siguiente repositorio de github: https://github.com/CEED-2021/actividad-intermedia-segunda-evaluacion

La plantilla admite los comandos `yarn test` para ejecutar las pruebas y `yarn lint` para ejecutar el linter.

3. Pruebas

Sólo debes crear los tests para la función `searchCharacter`. Debes rellenar los tests que hay en el fichero `tests/searchCharacter.test.js`. Puedes añadir otros tests si los necesitas, pero no se evaluarán (ni para bien ni para mal).

4. Linter

El comando `yarn eslint` ejecuta el linter sobre los directorios `src` y `test`. **Para que el ejercicio se corrija el linter no debe devolver ningún error**. En caso de que el linter devuelva un error, la máxima nota del trabajo será de 1.

Las reglas del linter son las que están especificadas en `.eslintrc.json`. No puede modificarse este fichero. Asimismo no se admitirá deshabilitar ninguna de las reglas de eslint por ningún medio: en ese caso se procederá como si el programa hubiese fallado el linter.

5. Restricciones

- **NO** se pueden obtener múltiples ítems a la vez. Por ejemplo, no podrías utilizar la petición https://rickandmortyapi.com/api/character/1,2,3 para obtener los personajes con ids 1,2 y 3. Tendrías que hacer tres peticiones individuales.

- **NO** puedes utilizar el API GraphQL
- **NO** puedes utilizar ninguna librería. Todo el código debe ser código original.
- Todo el código debe estar en el directorio `src`.
- Todas las pruebas deben estar en el directorio `tests`.

6. Entrega

Para la entrega debes eliminar los directorios `node_modules` y comprimir el directorio de tu proyecto en un único fichero `.zip` o `.gz`.

Debes entregar el ejercicio como un único fichero con el siguiente nombre: `NOMBRE_APELLIDO1.[zip|gz]` (sustituye *NOMBRE* y *APELLIDO1* por tu nombre y tu primer apellido) El fichero se entregará en la tarea del curso habilitada a tal efecto.

No se admitirán entregas pasada la fecha límite.

7. Criterios de evaluación

- Se cumple todo lo especificado en el apartado de restricciones
- El linter no devuelve ningún error
- El programa es correcto, realiza la función que se solicita en el enunciado
- Se han utilizado estructuras del lenguaje adecuadas: bucles, condicionales, operadores, etc.
- Se han utilizado variables y constantes de forma adecuada
- Se utilizan correctamente y cuando corresponda los tipos de datos y objetos predefinidos del lenguaje (Arrays, objetos planos, Map, Set, etc.)
- Se han utilizado funciones para estructurar el código, definiendo y utilizando parámetros y valores de respuesta de forma adecuada
- El programa es lo más sencillo posible para realizar su función.
- No existe código repetido: se han extraído los comportamientos comunes a funciones y se ha intentado hacer el código genérico.
- Se han creado pruebas para validar la funcionalidad del código utilizando las validaciones adecuadas.
- Se han estructurado las pruebas de forma correcta.