**React**

**Pruebas en el archivo 09-promesas.js - Tareas asíncronas.**

Documentación: <https://es.reactjs.org/>

<https://es.reactjs.org/docs/getting-started.html>

<https://jestjs.io/>

<https://jestjs.io/docs/snapshot-testing>

<https://jestjs.io/docs/expect>

Al realizar pruebas con funciones que son tareas asíncronas (promesas), Jest empieza a ejecutar todo el código en secuencia y cuando encuentra la promesa, solamente la ejecuta y se queda ejecutándose en el fondo, pero este después sigue ejecutando todo el código restante, llega al final y pues si no existe ningún expect, quiere decir que no hay ningún error en la prueba. Entonces nosotros tenemos que esperar la respuesta de la promesa para después validar lo esperado....

Para esto existen 2 formas de esperar la respuesta, 1 es utilizando el async await que lo veremos en la siguiente sesión o la forma 2 que es utilizar la función done() en la función de test, esta es una función que mandamos a llamar cuando se termina de ejecutar el código de la promesa. (done) le avisa a Jest cuando termina esa respuesta, pero si nosotros no utilizamos done() y usamos su referencia en test, Jest se va quedar esperando esa respuesta y después de un tiempo Jest determinará que habrá un error. Ejemplo:

**Código de archivo 09-promesas.js**

import { getHeroeById } from './08-imp-exp'

export const getHeroeByIdAsync = ( id ) => {

    return new Promise( (resolve, reject) => {

        setTimeout( () =>  {

            const p1 = getHeroeById( id );

            if ( p1 ) {

                resolve( p1 );

            } else {

                reject( 'No se pudo encontrar el héroe: ' + id );

            }

        }, 1000 )

    });

}

**Código de programación de prueba**

import {describe, expect, test} from '@jest/globals';

import { getHeroeByIdAsync } from '../../base-pruebas/09-promesas'

describe('Pruebas en 09-promesas', () => {

    test('getHeroeByIdAsync debe de retornar un objeto por id de heroe', (done) => {

        const HeroeId = 1;

        getHeroeByIdAsync( HeroeId ).then( Heroe =>{

            // Aquí ya podemos realizar el expect y validar lo esperado...

            // expect( true ).toBe( false ); // Error

            expect( Heroe ).toStrictEqual({

                id: 1,

                name: 'Batman',

                owner: 'DC'

            });

            done(); // Avisamos a Jest que terminamos de llamar nuestra promesa

        });

    })

    test('getHeroeByIdAsync debe de obtener un error si el heroe no existe', (done) => {

        const HeroeId = 100;

        getHeroeByIdAsync( HeroeId )

        .then( Heroe =>{

            expect( Heroe ).toBeFalsy(); // Si existe el heroe mandamos error, para validar un id heroe que no exista...

            done();

        })

        .catch( Error => {

            console.log( Error );

            expect( Error ).toBe( `No se pudo encontrar el héroe: ${HeroeId}` );

            done();

        });

    })

})