Certified Tech Developer
The Ultimate Degree

Testing I

API Testing- Creación de pruebas (JS)

Pasos

1. Comencemos con algunos conceptos a tener en cuenta:

- Para codificar los test con Postman debemos de conocer un poco el API que nos ofrecen. Cada uno de los test es ejecutado con el objeto `pm` y en concreto con el método `.test()`. Así, para cada uno, tendremos la siguiente estructura:

```
pm.test("Descripción Funcionalidad a Probar", function(){
  // Código que valida la prueba del test
});
```

- Para poder acceder al contenido de la respuesta de las invocaciones tenemos el objeto `pm.response` y su método `.json()` que nos permitirán acceder a los elementos de la respuesta en JSON.

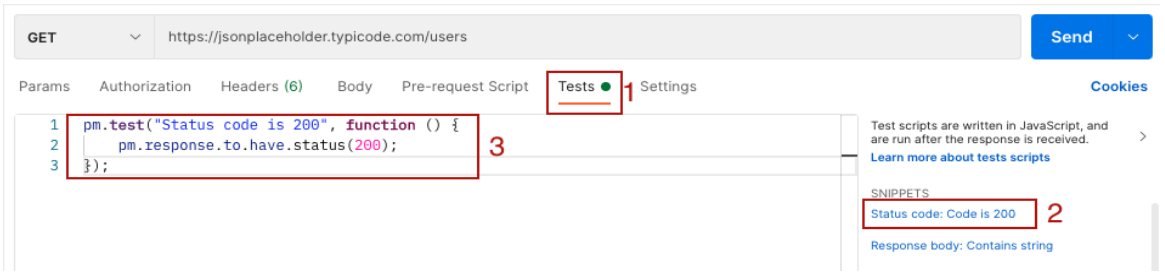
```
pm.test("Obteniendo contenido de la Response", function(){
   pm.response.json();
});
```

- Otro método importante es el que nos permite realizar una comprobación de contenido, este es pm.expect.

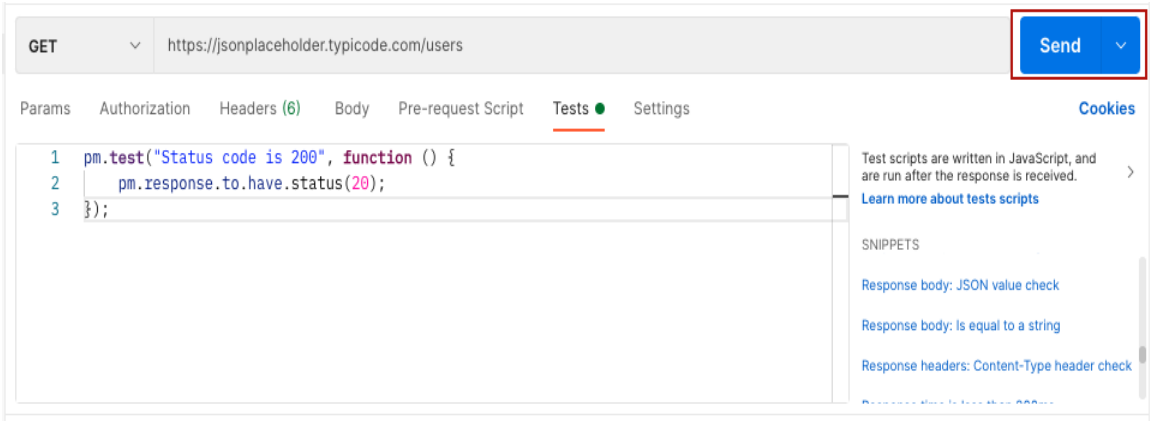
```
pm.test("Comparando el valor devuelto con el esperado", function(){
    pm.expect(valor).to.equal("Valor esperado")<
});
```

2. Teniendo en cuenta los conceptos ya definidos, veremos dos de las pruebas más utilizadas en el Testing de APIs. Postman nos brinda una serie de fragmentos por defecto que nos guía a la hora de construir nuestras pruebas:

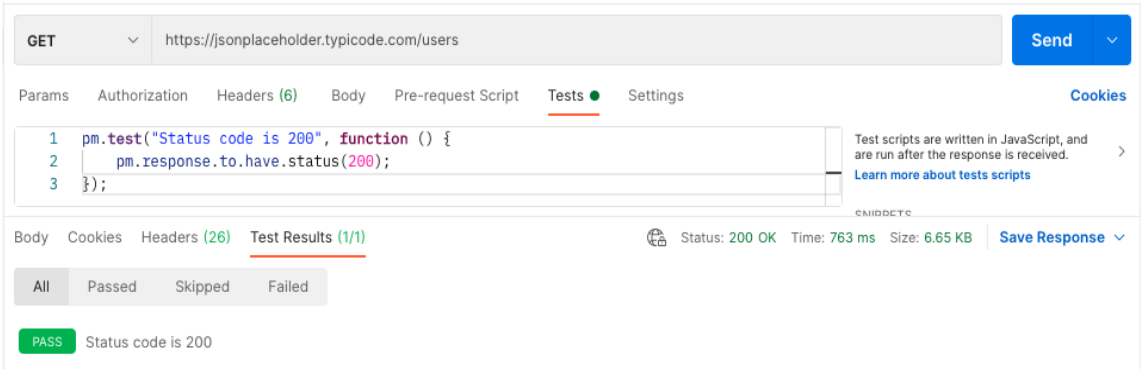
- Para comenzar, vamos a la solicitud GET que creamos anteriormente y seleccionamos la pestaña Pruebas (1) . En esta sección escribiremos nuestro set de pruebas relacionados con esa API. En la subsección de fragmentos, haremos clic en "Código de estado: el código es 200" (2) para generar una de las pruebas por defecto. La secuencia de comandos se completará automáticamente (3).



- Al hacer clic en ENVIAR se mostrará el resultado del test.

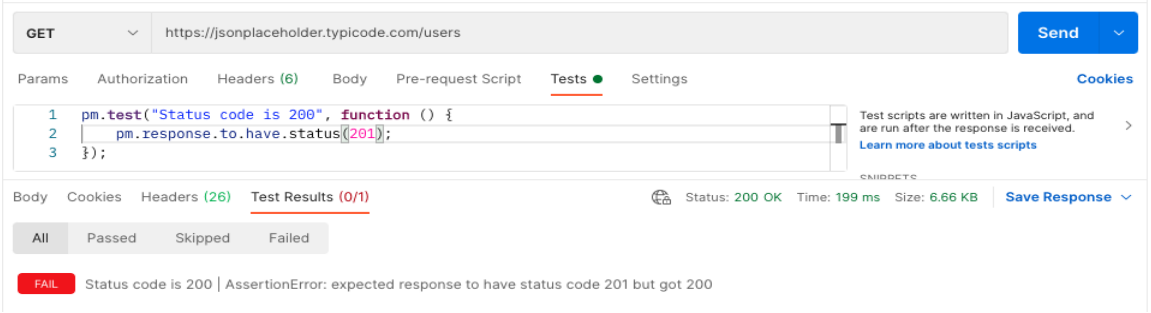


- Con esta prueba estamos validando que el código de respuesta de la API sea 200. Si esto es correcto el test devolverá PASS: eso significa que el servicio está respondiendo según lo esperado.



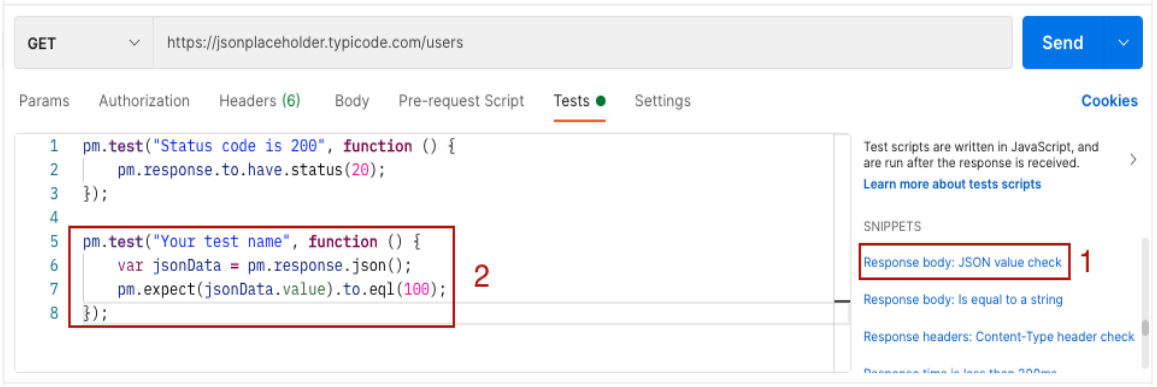
- Si el servicio falla nos mostrará el estado FAIL, y el código de error relacionado con este estado.

  En esta prueba estamos reutilizando fragmentos de código que nos brinda Postman para validar si la petición se realizó correctamente. Podemos editar esta consulta a nuestro gusto utilizando código javascript.



3. Agreguemos otra de las pruebas más usadas. En esta compararemos el resultado esperado con el resultado real.

- Para ello, en la subsección de fragmentos, haremos clic en "Cuerpo de respuesta: Verificación del valor del JSON" (1). La secuencia de comandos se completará automáticamente (2).

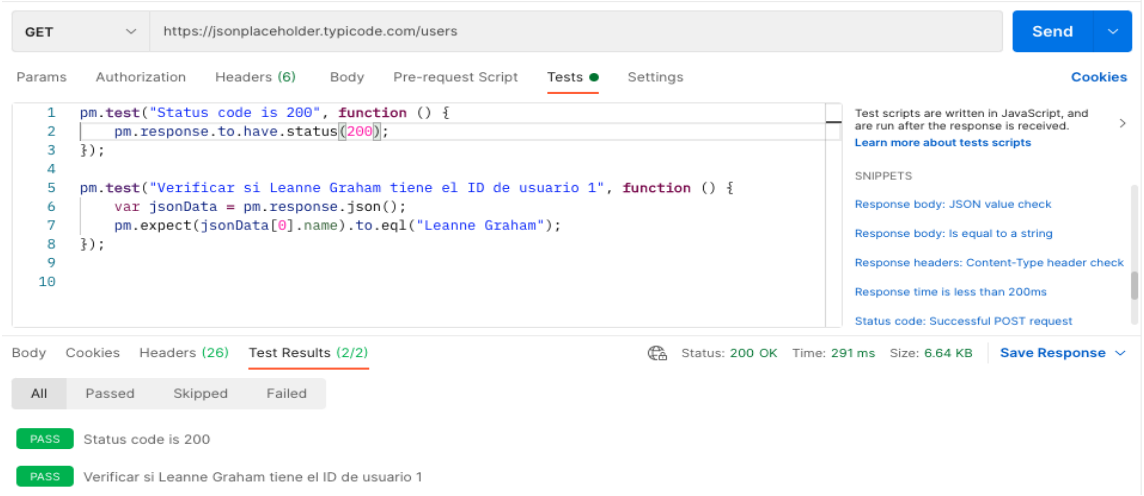


- Podemos cambiar el nombre de la prueba por defecto por el que más nos guste. En este caso lo reemplazamos por “Verificar si Leanne Graham tiene el ID de usuario 1”, dado que este es el primero usuario de la lista devuelta por el API. También debemos actualizar el cuerpo de la función reemplazando jsonData.value con jsonData[0].name; así obtendremos el primer elemento de la lista.

```
pm.test("Verificar si Leanne Graham tiene el ID de usuario 1", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData[0].name).to.eql("Leanne Graham");
});
```

- Al hacer clic en ENVIAR se mostrará el resultado.

- Se observa que nuestro test nos devolvió un estado “PASS”. De esta manera validamos que el contenido de la *response* es el esperado. Así, podremos ir validando diferentes datos y viendo si nuestra petición nos devuelve los datos deseados.



- Por último, se observa que al momento de enviar la petición se ejecutan todos los test relacionados a esta. De esta manera, podemos crear un set de test vinculados a cada petición y verificar rápidamente su estado.