Análisis de pruebas de Etherpad

Susana Rodríguez Lado

s.lado@udc.es

EP Lite Connection Test: Pruebas unitarias de conexión.

La mayoría son de caja negra, positivas, a excepción de la nº5, que es negativa, y las "handle", que son de caja blanca, y mayoritariamente negativas. Todas son dinámicas y funcionales.

1. domain_with_trailing_slash_when_construction_an_api_path()

   Comprueba la correcta generación de la URL de conexión, partiendo de una URL terminada en "/". Caja negra, positiva.

2. domain_without_trailing_slash_when_construction_an_api_path()

   Comprueba la correcta generación de la URL de conexión, partiendo de una URL que no termina en "/". Caja negra, positiva.

3. query_string_from_map()

   Comprueba que se añaden correctamente argumentos de un mapa a la URL. Caja negra, positiva.

4. url_encoded_query_string_from_map()

   Comprueba que se añaden correctamente argumentos de un mapa a la URL, incluyendo caracteres extraños. Caja negra, positiva.

5. api_url_need_to_be_absolute()

   Comprueba la excepción de recibir una query nula al generar el path URL. Es de caja negra, negativa, ya que fuerza un caso de error.

6. handle_valid_response_from_server()

   Comprueba que el método handleResponse(), enviándole una lista de pads, recupera correctamente únicamente el primero de la lista. Caja negra, positiva, ya que no busca un caso de error.

7. handle_invalid_parameter_error_from_server()

   Comprueba el método handleResponse(), esta vez recibiendo un parámetro incorrecto. Caja blanca, negativa, ya que fuerza un caso de error y que se maneje correctamente.

8. handle_internal_error_from_server()

   Comprueba el método handleResponse(), esta vez forzando un error interno del servidor. Caja blanca, negativa, ya que fuerza un caso de error y que se maneje correctamente.

9. handle_no_such_function_error_from_server()

   Comprueba el método handleResponse(), esta vez recibiendo una función inexistente. Caja blanca, negativa, ya que fuerza un caso de error y que se maneje correctamente.

10. handle_invalid_key_error_from_server()

    Comprueba el método handleResponse(), esta vez recibiendo una key inválida. Caja blanca, negativa, ya que fuerza un caso de error y que se maneje correctamente.

11. unparsable_response_from_the_server()

    Comprueba el método handleResponse(), esta vez recibiendo un parámetro no parseable. Caja negra, negativa, ya que fuerza un caso de error y que se maneje correctamente.

12. unexpected_response_from_the_server()

    Comprueba el método handleResponse(), esta vez recibiendo un parámetro vacío, que da lugar a una respuesta inesperada del servidor. Caja negra, negativa, ya que fuerza un caso de error y que se maneje correctamente.

13. valid_response_with_null_data()

    Comprueba el método handleResponse(), esta vez recibiendo un parámetro correcto, con cuerpo vacío, por lo que la respuesta esperada será nula. Caja negra, positiva.

EP Lite Client Integration Test: Pruebas de integración del cliente.

Son todas de caja negra, positivas (no se buscan los casos de error ni los valores límite, solo las ejecuciones correctas y el funcionamiento esperado), dinámicas y funcionales.

14. validate_token()

    Comprueba la correcta validación de un token (no se prueban casos de token incorrectos).

15. create_and_delete_group()

    Comprueba que tras crear un grupo, se devuelva un elemento no-nulo que comienza por g (indicador de grupo), pero no comprueba ni su contenido ni su correcta eliminación.

16. create_group_if_not_exists_for_and_list_all_groups()

    Crea un grupo que no existía, y comprueba que se devuelve un groupID (sin comprobar que el código de grupo comience por g), y comprueba el tamaño de la lista resultado de la búsqueda de grupos, sin comprobar su contenido ni su correcta eliminación.

17. create_group_pads_and_list_them()

    Comprueba el contenido de los parámetros "public status" e "is password protected" del grupo, la correcta creación de un pad y su texto inicial, y el tamaño de la lista de pads (sin comprobar su contenido salvo por comprobar que contiene el pad creado). No comprueba la correcta eliminación del grupo.

18. create_author()

    Comprueba, tras crear un autor, que su id no es nula y su nombre es correcto, sin comprobar el resto de parámetros.

19. create_author_with_author_mapper()

    Crea dos autores diferentes nuevos y comprueba que sus nombres son diferentes, y luego trata de crear un tercer autor que ya existe, por lo que debe ser igual al primero, comprobando que los nombres sean iguales.

20. create_and_delete_session()

    Crea dos sesiones, comprobando que sus Ids no coinciden, también verifica sus autores y grupos, y las borra, pero sin comprobar su correcta eliminación.

21. create_pad_set_and_get_content()

    Crea pads y comprueba sus atributos y cantidad de usuarios, propiedad “read only”, autores y propiedad “last edited”, pero no comprueba su correcta eliminación.

22. create_pad_move_and_copy()

    Crea, mueve y copia un pad, comprobando el contenido del texto copiado y movido.

23. create_pads_and_list_them()

    Crea dos pads y los lista, comprobando que la lista tiene dos o más elementos (no exactamente dos), y que contiene los pads creados.

24. create_pad_and_chat_about_it()

    Crea mensajes en pads, comprobando la cantidad de mensajes y su contenido, pero no comprueba el correcto borrado del pad.

Tabla de tipos:

Nº	Tipo de caja	+/-	Estática/Dinámica	Funcional?	Unitaria/Integración
1	Caja negra	Positiva	Dinámicas	Si	Unitarias
2	Caja negra	Positiva			
3	Caja negra	Positiva			
4	Caja negra	Positiva			
5	Caja negra	Negativa			
6	Caja negra	Positiva			
7	Caja blanca	Negativa			
8	Caja blanca	Negativa			
9	Caja blanca	Negativa			
10	Caja blanca	Negativa			
11	Caja negra	Negativa			
12	Caja negra	Negativa			
13	Caja negra	Positiva			
14	Caja negra	Positivas	Dinámicas	Si	De integración
15					
16					
17					
18					
19					
20					
21					
22					
23					
24					

Conclusión:

Hay excepciones sin comprobar tanto en las pruebas unitarias como en las de integración.

Se necesitan más pruebas negativas, que fuercen la aparición de errores y comprueben los valores límite, ya que no se están probando muchos casos que podrían resultar problemáticos.

La cobertura de las pruebas ronda el 94%, pero en muchos casos el código cubierto sólo está siendo ejecutado sin buscar los posibles errores, por lo que no se garantiza su correcto funcionamiento.