Análisis de los test disponibles para la aplicación etherpad-client-lite

Autor: Luis Ogea Borbón

Correo: luis.ogea.borbon@udc.es

Indice:

Test del archivo de EPLiteConnectionTest.java:

Tipo de test	Total de test
Dinámica, caja blanca, positiva, estructural	5
Dinámica, caja blanca, negativa, estructural	6
Dinámica, caja negra, negativa, funcional	2

domain_with_trailing_slash_when_construction_an_api_path :

Tipo: Dinámica, caja blanca, positiva, estructural

domain_without_trailing_slash_when_construction_an_api_path:

Tipo: Dinámica, caja blanca, positiva, estructural

Estos dos tests son de caja blanca ya que prueban a introducir la url con y sin el / para comprobar que el método EPLiteConnection pase por los distintos “caminos” que hay.

query_string_from_map:

Tipo: Dinámica, caja blanca, positiva, estructural

url_encoded_query_string_from_map:

Tipo: Dinámica, caja blanca, positiva, estructural

Estos dos tests son de caja blanca ya que hay conocimiento del código a la hora de crearlos, tienen un boolean el cual si esta en false genera solo el query y si es true realiza un encode.

api_url_need_to_be_absolute:

Tipo: Dinámica, caja negra, negativa, funcional

En mi opinión es de caja negra ya que no es necesario que conozca el código del método que cree la URI (método que no es suyo, esta importado), solo hace falta saber que ese método falla si se introduce una de las opciones.

handle_valid_response_from_server:

Tipo: Dinámica, caja blanca, positiva, estructural

handle_invalid_parameter_error_from_server:

Tipo: Dinámica, caja blanca, negativa, estructural

handle_internal_error_from_server:

Tipo: Dinámica, caja blanca, negativa, estructural

handle_no_such_function_error_from_server:

Tipo: Dinámica, caja blanca, negativa, estructural

handle_invalid_key_error_from_server:

Tipo: Dinámica, caja blanca, negativa, estructural

unparsable_response_from_the_server:

Tipo: Dinámica, caja blanca, negativa, estructural

unexpected_response_from_the_server:

Tipo: Dinámica, caja blanca, negativa, estructural

En el caso de estos últimos 7 test, son de caja blanca ya que se comprueba en cada uno de ellos una parte del case del que disponen para diferenciar cada tipo de código que reciben del servidor y cada una de las excepciones que se pueden dar, por lo que se conoce que esos códigos recorren distintos caminos y en estos test los prueban todos. A excepción de la primera el resto considero que son negativas ya que se prueba lo que pasaría al recibir un mensaje de error del servidor.

valid_response_with_null_data:

Tipo: Dinámica, caja negra, negativa, funcional

A diferencia de las 7 pruebas anteriores, considero que estando en estas todos los casos probados esta no prueba nada nuevo y es básicamente una variación de handle_valid_response_from_server, no requiere de conocimiento sobre el código para saber que si no hay datos recibirás un null (valdrá con leer la documentación de etherpad-lite).

Test del archivo de EPLiteClientIntegrationTest.java:

Tipo de test	Total de test
Dinámica, caja negra, positiva, funcional	11

validate_token:

Tipo: Dinámica, caja negra, positiva, funcional

Comprueba que pasando un token válido el método no falla, no requiere de conocimiento sobre el código.

create_and_delete_group:

Tipo: Dinámica, caja negra, positiva, funcional

Simplemente crea un grupo con valores válidos y comprueba que funciona bien el método. No veo que compruebe que el deleteGroup() realmente a borrado el grupo, por lo que considero que este test que debería probar el método create y delete esta incompleto.

create_group_if_not_exists_for_and_list_all_groups:

Tipo: Dinámica, caja negra, positiva, funcional

Simplemente crea un grupo con valores válidos y comprueba que funciona bien el método. Hace una segunda llamada al createGroupIfNotExist que no debe de hacer nada en el servidor ya que ya existe un grupo y prueba el método que los lista y compara los resultados para comprobar que la segunda llamada no a creado un grupo.

create_group_pads_and_list_them:

Tipo: Dinámica, caja negra, positiva, funcional

Similar al test anterior, aunque comprueba varios otros métodos relacionados a la creación y uso de los pads.

create_author:

Tipo: Dinámica, caja negra, positiva, funcional

Crea un autor (con valores válidos) y obtiene su nombre, no requiere de conocimiento del código.

create_author_with_author_mapper:

Tipo: Dinámica, caja negra, positiva, funcional

Similar al test de create_group_if_not_exists_for_and_list_all_groups sin la parte de listar.

create_and_delete_session:

Tipo: Dinámica, caja negra, positiva, funcional

Comprobación de la creación de una sesión (con valores válidos), al igual que en el test de creación de grupos, no se comprueba el delete, solo se hace la llamada al método y ya está.

create_pad_set_and_get_content:

Tipo: Dinámica, caja negra, positiva, funcional

Comprueba en esta prueba la creación de pads, los métodos de set y los diferentes gets relativos a los pads. Además, aprovecha el test para meter caracteres especiales y probar que se realiza bien su codificación y descodificación. También realiza el delete, pero no lo prueba ni en este método ni en los restantes.

create_pad_move_and_copy:

Tipo: Dinámica, caja negra, positiva, funcional

En este test prueba el método de creación de pads con texto, el move y copy de los pads (con valores válidos).

create_pads_and_list_them:

Tipo: Dinámica, caja negra, positiva, funcional

Crea un 2 pads y comprueba que al listarlos se devuelven los dos.

create_pad_and_chat_about_it:

Tipo: Dinámica, caja negra, positiva, funcional

Prueba las opciones de chat de los pads.

Opinión:

En general se realizan pruebas de todas las funcionalidades implementadas en esta aplicación. La cobertura resultante de ejecutar todos los test es del 93%. Esto en un principio es una cobertura bastante aceptable. El problema es si los test están bien implementados o no, en este documento se realiza un breve análisis de los test, comprobando lo que hacen y clasificándolos en los distintos tipos de test.

Tras el análisis de los test disponibles, me he encontrado con lo siguiente:

- EPLiteConnectionTest.java: En este fichero puede observarse que hay una gran variedad de tipos de test, en conjunto prueba cada uno de los caminos de los métodos, probando tanto casos donde el resultado esperado es correcto como los casos donde se pueden producir las distintas excepciones disponibles.

  En este test se prueba la conexión al servidor y la recepción de los diferentes tipos de código que pueden tener los mensajes.

- EPLiteClientIntegrationTest.java: En este fichero se prueban las distintas funcionalidades de la aplicación. Lo que he podido observar en este fichero es distinto a lo que encontré en el anterior. Todos los test de este fichero son del mismo tipo (dinámicos, de caja negra, positivos y funcionales). Todos los test son probados con valores válidos para así recibir una respuesta correcta, esto en principio puede ser normal ya que los casos donde el mensaje recibido por el servidor es de error ya los prueba en el fichero anterior, por lo que ya que no se trata de probar que el método del servidor funciona como debe, con eso es suficiente. Aunque, podrían haberlos hecho perfectamente, para comprobar que una petición incorrecta devuelve el mensaje de error que se espera.

En general, considero que los test (sobre todo en el último archivo) no están todo lo bien que deberían. Uno de los problemas que encontré es que algunos de los test son un popurrí, es decir, prueban una gran cantidad de métodos en un mismo test lo que hace que estos sean un poco más difíciles de leer. Tampoco ayuda los nombres que han decidido darles a estos, podemos encontrar test que prueban métodos que no aparecen en su nombre, cuatro test que empiezan con create_pad, cuando solo es necesario probarlo una vez (dos si se tiene en cuenta la creación con texto, pero eso iría en dos test separados y no junto a la prueba de otros métodos como el de set html).

Otro defecto es el de no comprobar lo que devuelve los métodos delete, ya no digo que compruebe que se ha borrado efectivamente el objeto (que podría), si no que los pruebe al menos como hizo con el create, que se asegure que el mensaje devuelto sea el de “ok” y no un mensaje de error ya que según la

documentación de etherpad-lite se puede devolver dos tipos de mensaje (uno correcto y otro de error).

Hecho en falta alguna prueba con valores aleatorios, sobre todo en funcionalidades como las de chat, la escritura en los pads, la creación de autores (para el nombre) y la de grupos (también para el nombre).

En conclusión, creo que se prueba lo básico aunque el conjunto de pruebas se podría mejorar.