Test DocumentsSet

Objecte de la prova

En aquesta prova es realitza un test unitari sobre la classe DocumentSet. Aquesta classe és la que rep, des del controlador de domini, les funcions de crear i esborrar documents del sistema, llista els documents de la forma demanda, llista els títols d'un autor, llista els autors segons un prefix del seu nom, obtenir el contingut d'un document o modificar aquest. Per tant, és responsable dels casos d'ús "Carregar document", "Carrega més d'un document", "Alta document buit", "Seleccionar un document", "Llistar k més semblants", "Modificar un document", "Baixa d'un document", "Guardar el contingut d'un document", "Llistar documents", "Obtenir document per títol i autor", "Llistar els títols d'un autor", "Llistar autors per prefix", "Llistat del k documents més rellevants d'un query" i "llistar per expressió booleana".

Altres elements integrats a la prova

En aquest test hem integrat la classe del domini Pair. En el diagrama de classes del sistema ja hem esmentat que DocumentsSet té una dependència cap a la classe Pair. Aquesta classe és molt bàsica i ja hem comprovat de manera unitària, en test a part, que la seva funcionalitat és adequada. Per aquest motiu, hem decidit no realitzar un stub de Pair ni mockejar-la, sinó que emprar la que hem implementat nosaltres directament.

Drivers

No s'usa cap driver en aquest test.

Stubs

Per tal de realitzar aquest test, hem usat dos stubs. Com observem en el diagrama de classes del nostre sistema, la classe DocumentsSet té una associació amb Document i dependències de les classes d'Expression i de Pair. Pel que fa a la classe Pair, ja hem esmentat que hem justificat que hem fet servir la classe que hem implementat i no cap stub. Ara bé, per Document i per Expression, que són classes força més complexes, sí que hem decidit realitzar stubs per assegurar que les proves realitzades

se centren en les funcions pròpies de DocumentsSet i no influeixen la resta de classes de què depèn.

- Stub de Document: Document té diverses funcions complexes que hem implementat en aquest stub de manera molt simplificada per minimitzar possibles errors en càlculs que a DocumentsSet no li influeixen.

Pel que fa a les constructores, les hem declarat totes, i hem comprovat les excepcions que es poden donar.

Sobre la funció getRelevantWords(), aquesta funció és complexa en tant que és informació que no s'obté directament del Document sinó que de la seva representació interna. Per aquest motiu, per simplificar, hem suposat que tot contingut dels documents que usarem de proves estarà format per caràcters separats per espais entre ells. Això no influeix a DocumentsSet, que no analitza el contingut que rep, i ajuda que puguem retornar informació en aquesta funció de manera molt senzilla. Per aquest motiu i a partir d'aquesta decisió, en el test tots els documents que creem tenen un contingut que compleix aquesta restricció (per exemple, el contingut "aaa" no l'usem, faríem servir "a a a").

Pel que fa a la resta de funcions, totes complexes, que comparen el contingut amb diferents estratègies com tf-idf, tf-boolean o a partir d'una query (compare_tf_idf(), compare_tf_boolean() i queryRelevance()), hem decidit que dos strings s'assemblen més o menys en funció de si la llargada d'aquests strings és major o menor, respectivament. Això ens permet crear funcions molt senzilles per sobreescriure operacions complexes que no afecten DocumentsSet i en permeten el seu testing de manera fàcil.

- Stub de Expression: Aquesta també és una classe algorísmicament complexa en les dues funcions públiques que ofereix, la de crear i la d'avaluar (create() i evaluate()).

Pel que fa a create(), la capçalera d'aquest mètode a la classe Expression informa que pot llençar una excepció en cas de mal format (ExceptionInvalidExpression). Per imitar aquesta funció, doncs, el stub retorna sempre una "Expression", excepte si el paràmetre que rep com a identificador de l'expressió és "invalid", que aleshores llença aquesta excepció. Ho hem dissenyat d'aquesta manera perquè la finalitat de DocumentsSet no és comprovar que una expressió sigui vàlida o no, això és tasca de la mateixa classe Expression, i d'aquesta manera podem provar què passaria en cas que una expressió fos invàlida.

Sobre la funció d'evaluate(), aquesta retorna un valor cert en cas que la llargada del contingut a avaluar sigui major a 5, una implementació molt senzilla que segur que no és errònia, però ja ens serveix per testejar i fer proves adientment sobre el que realment ens interessa a DocumentsSet.

Fitxers de dades necessaris

No es requereix cap fitxer addicional.

Valors estudiats

Aquest test l'hem desenvolupat seguint l'estratègia de caixa blanca, ja que els desenvolupadors del test coneixíem el codi de la classe ExpressionsSet. En aquest sentit, doncs, hem intentat provar totes les funcions públiques de la classe i els diferents casos (amb les possibles excepcions) que es poden donar. Abans de tots els tests, s'executa una inicialització de valors de prova (testValues) què ens serveixen com a conjunt de documents de prova per inicialitzar la instància, en alguns tests.

- test de getInstance()

La classe DocumentSet és singleton, així que té el mètode getInstance. El que comprovem és que les instàncies que ens retornen diferents crides a aquesta funció són, efectivament, la mateixa.

- test dels getters i setters

Simplement comprovem amb un conjunt de documents de prova que si fem set i posteriorment get obtenim el mateix resultat.

- test de get numDocs

Simplement comprovem amb un conjunt de documents de prova que si fem set i posteriorment get de numDocs aconseguim el nombre de documents correcte.

- test de get presence

Simplement comprovem amb un conjunt de documents de prova que si fem set i posteriorment get de presence obtenim el mapa de presència correcte.

-test de create Document

En aquest test volem provar diferents coses. Primerament, el cas en què cridem a crear un document que ja existeix fent saltar la seva excepció i el cas en què fem crear un document amb un llenguatge invàlid. Per altra banda, el cas en què creem el document hem de comprovar que s'ha afegit el document amb el contingut adequat, però no només això sinó també que s'ha actualitzat el mapa de presència i el nombre de documents.

-test de delete Document

Aquest test és molt semblant al de create Document. Primerament, el cas en què cridem a esborrar un document que no existeix fent saltar la seva excepció. Per altra banda, el cas en què esborrem el document hem de comprovar que s'ha esborrat el document, però no només això sinó també que s'ha actualitzat el mapa de presència i el nombre de documents.

-test de exist document

Simplement comprovem amb un conjunt d'expressions de prova que si fem exist d'un document existent ens torna true i si no existeix serà fals.

-test de get content d'un document

Simplement comprovem amb un conjunt de documents de prova que si fem set i posteriorment get del contingut d'un document obtenim el contingut correcte i finalment si no existeix el document, torna l'excepció de no existència del document.

-test update content d'un document

Simplement comprovem amb un conjunt de documents de prova que si fem set i posteriorment update del contingut d'un dels documents i posteriorment un get del contingut aconseguim el nou contingut correcte i finalment si no existeix el document, torna l'excepció de no existència del document.

-test list similars

Aquest test comprèn moltes possibilitats, la primera que hem de comprovar és el cas en què k (nombre de documents que volem que torni) és negatiu i retorni l'excepció. En el segon cas, proven si cridem a una estratègia inexistent si ens retorna l'exececpció pròpia del cas. Per posar fi a les excepcions, també hem de comprovar el cas que busquem un document similar d'un document que no existeix. Finalment, en el cas base comprovar que surten els documents correctes ordenats adequadament i per últim comprovar que no s'ha modificat res del DocumentSet.

-test list by expression

Simplement comprovem amb un conjunt de documents de prova que si fem set i més tard el list by expression d'una expressió a ens retorna la llista de documents que compleixen l'expressió i comprovar que res de document set ha sigut modificat.

-test list titles of author

Simplement comprovem amb un conjunt de documents de prova que si fem set i fem list d'un autor qualsevol ens retorna tots els seus títols dels seus documents i no s'ha modificat res de document set.

-test list titles autors by prefix

Simplement comprovem amb un conjunt de documents de prova que si fem set i fem list per prefix de nom d'autors qualsevol ens retorna tots els noms que tenen com prefix aquell mot i finalment que no s'ha modificat res de document set.

-test list by query
Té una casuística més complicada, per una banda, tenim el cas en què afegim una k (nombre de documents que volem que torni) negatiu provant si retorna l'excepció. En conclusió, el cas base d'una query i comprovar si retorna els documents correctes i no modifica document set.

Efectes estudiats

No aplica en aquest test, que no és de la capa de presentació.

Operativa

Per executar aquest test, cal introduir per consola l'ordre d'execució de java usant les llibreries de junit i hamcrest. Si es realitza des del directori arrel del projecte, és:
java -cp ./EXE/DocumentsSet:./lib/junit-4.12.jar:./lib/hamcrest-core-1.3.jar org.junit.runner.JUnitCore test.domain.documents.TestDocumentsSet