EXPLICACIÓ JOCS DE PROVA

Júlia Amenós Dien
Víctor Mena Doz
Marc Navarro Acosta
Jordi Soley Masats

ÍNDEX

1. Author

- ## Objecte de la Prova:

L'objecte de la prova és la classe Author, de la qual volem verificar el correcte funcionament de totes les seves funcionalitats.

- ## Altres elements integrats en la Prova:

En la prova no s'integren altres elements.

- ## Drivers:

Per aquesta prova el driver que s'ha desenvolupat ha estat AuthorDriver.

- ## Stubs:

No s'ha utilitzat stubs ja que la classe Author, no conté cap element/instància de tipus de alguna de les altres classes. Exclusivament té un ArrayList de Strings.En la implementació també es tracta amb Integers i Strings.

- ## Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input determinat i l'output esperat. De tal forma que al executar-lo amb l'input, podem comprovar si l'output obtingut és diferent a l'output esperat.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures. De totes formes podríem dir que estem fent servir una Caixa gris ja que el propi Usuari pot provar les funcions amb el driver.

- ## Operativa:

Un cop s'executa, s'imprimeix per pantalla un llistat de totes les funcions numerades que es poden testejar. Demana el nombre de la funció que es vol testejar. Si es prem 0, es testejen totes les funcions per ordre una a una. Si es prem 9 finalitza el programa.

En aquest cas hi ha 8 funcions, numerades del 1 al 8.

1. testAuthorConstruct → Testeja la creació d'un Author correctament donat un nom inicial.
2. testSetName → Testeja la funcionalitat de canviar-li el nom a un Author.
3. testGetName → Testeja la funcionalitat d'obtenir el nom (String) d'un Author.
4. testGetTitles → Testeja la funcionalitat d'obtenir un ArrayList de Strings dels titols dels documents d'un Author determinat.
5. testGetNumTitles → Testeja la funcionalitat d'obtenir el número (Integer) de documents d'un Author determinat.
6. testAddTitle → Testeja la funcionalitat d'afegir-li el titol d'un document a un Author. Recordem que un Author guarda els titols dels seus documents, no els documents com a tal.
7. testDelTitle → Testeja la funcionalitat de borrar el titol d'un document del seu Author.
8. testMatchesPrefix → Testeja la funcionalitat de comprovar si el nom d'un autor comença per un prefix donat.

Per el test de cada funció, es van imprimint per pantalla diversos missatges i demana que s'introdueixin diversos camps que es van especificant.

2. Content

● Objecte de la Prova:

L'objecte de la prova és la classe Content, on emmagatzemem de forma eficient el contingut d'un document per després ser utilitzat en alguns casos d'ús com les cerques i les modificacions de documents.

● Altres elements integrats en la Prova:

Per aquesta prova s'utilitza la classe Sentence quan ha estat provada anteriorment, en el cas del Driver.
És necessari comentar que el JUnit no fa servir la mateixa dinàmica perquè entenem que els drivers permeten fer jocs de prova customitzables i per tant han d'utilitzar objectes reals mentre que els JUnit fan servir Stubs.

● Drivers:

S'ha desenvolupat un driver anomenat ContentDriver per provar aquesta classe de forma personalitzada.

La missió del driver és detectar sortides incorrectes de forma que no salti cap excepció ni generi una sortida incorrecta.

- ## Stubs:

La prova unitària mitjançant la classe JUnit fa servir SentenceStub que li retorna valors predefinits per tal de no encadenar errors d'una classe testejada prèviament.

- ## Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ## Efectes estudiats:

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ## Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els testos, en ordre.
    1. testContentConstruct → Test de la creadora.
    2. testGetLanguage → Testeja la funcionalitat d'obtenir el llenguatge del text guardat a content.
    3. testGetText → Testeja la funcionalitat d'obtenir el text ben posat.
    4. testGetSentences → Testeja la funcionalitat d'obtenir una llista de totes les Sentence que composen el Content.
    5. testWordContain → Testeja la funcionalitat de buscar si hi ha un grup de paraules al text.
    6. Exit → Acaba l'execució del driver.

3. Sentence

- ## Objecte de la Prova:

L'objecte de la prova és la classe Sentence, on s'emmagatzema una linea del content d'un document que ens serà d'utilitat després per a les cerques.

- ## Altres elements integrats en la Prova:

–

- ## Drivers:

S'ha desenvolupat un driver anomenat SentenceDriver per provar aquesta classe de forma personalitzada.

- ## Stubs:

–

- ## Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ## Efectes estudiats

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ## Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els testos, en ordre.
1. testSentenceConstruct → Test de la creadora.

2. testSearchWord → Testeja la funcionalitat de buscar si hi ha un grup de paraules al text.
3. testGetSentence → Testeja la funcionalitat d'obtenir la sentence com a string ben construit.
4. Exit → Acaba l'execució del driver.

4. Boolean Expression

- ### Objecte de la Prova:

L'objecte de prova és la classe Boolean Expression, on es comprovaran totes les funcions relacionades amb la creació i validesa de les expressions booleanes.

- ### Altres elements integrats en la Prova:

–

- ### Drivers:

S'ha desenvolupat el BooleanExpressionDriver, que permet a l'usuari testejar la validesa de les seves expressions.

- ### Stubs:

No és necessària cap classe auxiliar per provar els mètodes, per tant, no s'han fet servir stubs.

- ### Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ### Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, atès que és el creador de la classe qui fa els jocs de prova i pot introduir contingut erroni. El Driver i el JUnit un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ### Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els tests, en ordre.
1. testBooleanExpression → Comprova si una expressió booleana és vàlida i la inicialitza.
2. testGetExpression → Retorna l'expressió booleana original, sense cap simplificació, creada per l'usuari.
3. testIsDocumentValid → Comprova si una frase satisfà l'expressió booleana.
4. Exit → Acaba l'execució del driver.

5. Document

- ### Objecte de la Prova:

L'objecte de la prova és la classe Document, on comprovem totes les seves funcionalitats: creacions, modificacions i protecció.

- ### Altres elements integrats en la Prova:

Per realitzar aquesta prova s'utilitzen la classe Content quan ha estat provada anteriorment, en el cas del Driver.
És necessari comentar que el JUnit no fa servir la mateixa dinàmica perquè entenem que els drivers permeten fer jocs de prova customitzables i per tant han d'utilitzar objectes reals mentre que els JUnit fan servir Stubs.

- ### Drivers:

S'ha desenvolupat un driver anomenat DocumentDriver per provar aquesta classe de forma personalitzada.

La missió del driver és detectar sortides incorrectes de forma que no salti cap excepció ni generi una sortida incorrecta.

- ### Stubs:

La prova unitària mitjançant la classe JUnit fa servir un ContentStub que li retorna valors predefinits per tal de no encadenar errors d'una classe testejada prèviament.

- ### Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ## Efectes estudiats:

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ## Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els testos, en ordre.
1. testDocumentConstruct1 → Test de la creadora 1.
2. testDocumentConstruct2 → Test de la creadora 2.
3. testSetAuthor → Testeja la funcionalitat de canviar-li l'Author a un Document.
4. testSetTitle → Verifica el canvi el títol del document.
5. testSetContent → Verifica que s'inclou un contingut al document.
6. testSetLanguage → Comprova que s'ha canviat el llenguatge del document.
7. testProtectDocument → Comprova que s'ha protegit un document amb una determinada contrasenya.
8. testGetTitle → Valida que es retorna el títol del document.
9. testGetAuthor → Valida que es retorna el nom de l'Author d'un Document
10. testGetContent → Testeja la funcionalitat d'obtenir el Content d'un Document
11. testGetLanguage → Testeja la funcionalitat d'obtenir l'idioma d'un Document.
12. testContentSearch → Testeja la funcionalitat d'obtenir el vector de freqüència del contingut del document.
13. testIsProtected → Testeja la funcionalitat de comprovar si un document està protegit.
14. Exit → Acaba l'execució del driver.

6. Folder

- ## Objecte de la Prova:

L'objecte de la prova és la classe Folder, on comprovem que totes les funcions relacionades amb la gestió de documents (afegir/eliminar/modificar/protegir).

- ## Altres elements integrats en la Prova:

Per realitzar aquesta prova s'utilitzen les classes Document i Contingut quan han estat provades anteriorment, en el cas del Driver.
És necessari comentar que el JUnit no fa servir la mateixa dinàmica perquè entenem que els drivers permeten fer jocs de prova customitzables i per tant han d'utilitzar objectes reals mentre que els JUnit fan servir Stubs.

- ## Drivers:

S'ha desenvolupat un driver anomenat FolderDriver per provar aquesta classe de forma personalitzada.

La missió del driver és detectar sortides incorrectes de forma que no salti cap excepció ni generi una sortida incorrecta.

- ## Stubs:

La prova unitària mitjançant la classe JUnit fa servir un DocumentStub que li retorna valors predefinits per tal de no encadenar errors d'una classe testejada prèviament.

- ## Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ## Efectes estudiats:

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ## Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els tests, en ordre.
1. testFolder → Test de la creadora.
2. testSetFolderName → Testeja la funcionalitat de canviar el nom a una carpeta.
3. testCreateFolder → Testeja la funcionalitat de crear una subCarpeta.
4. testAddDocument → Testeja la funcionalitat d'afegir un Document a la carpeta amb un Objecte Document.
5. testAddNonConstructedDocument → Testeja la funcionalitat d'afegir un Document a la carpeta amb els atributs necessaris d'un Objecte Document.
6. testDelDocument → Testeja la funcionalitat d'esborrar un Document de la carpeta.
7. testModifyContent → Testeja la funcionalitat de modificar el Contingut d'un Document.
8. testModifyAuthor → Testeja la funcionalitat de modificar l'Autor d'un Document.
9. testModifyTitle → Testeja la funcionalitat de modificar el títol d'un Document.
10. testProtectDocument → Testeja la funcionalitat de protegir un Document amb contrasenya.
11. testGetName → Testeja la funcionalitat d'obtenir el nom de la carpeta.
12. testGetDocumentsName → Testeja la funcionalitat d'obtenir el nom de tots els Documents de la carpeta.
13. testGetDocumentAmount → Testeja la funcionalitat d'obtenir el nombre de Documents de la carpeta.
14. testGetDocument → Testeja la funcionalitat d'obtenir el contingut d'un Document de la carpeta.
15. testGetMapsDoc → Testeja la funcionalitat d'obtenir l'estructura interna del vector de freqüències, necessari per les cerques.
16. testDocumentContained → Testeja la funcionalitat d'esbrinar si un document hi és a la carpeta.
17. testFolderContained → Testeja la funcionalitat d'esbrinar si la carpeta indicada és una subcarpeta d'aquesta.
18. Exit → Acaba l'execució del driver.

7. AuthorsController

- ## Objecte de la Prova:

L'objecte de la prova és el controlador d'autors (AuthorsController), de la qual volem verificar el correcte funcionament de totes les seves funcionalitats. Principalment la gestió d'autors del sistema.

- ## Altres elements integrats en la Prova:

Per realitzar aquesta prova s'utilitza la classe Author quan ha estat provada anteriorment. Els drivers permeten fer jocs de prova customitzables i per tant han d'utilitzar objectes reals i no cal que utilitzin Stubs. Al ser un controlador no s'ha fet un Junit.

- ## Drivers:

Per aquesta prova el driver que s'ha desenvolupat ha estat AuthorsControllerDriver.

- ## Stubs:

No s'ha utilitzat Stubs ja que al ser un controlador, només s'ha fet un driver i no Junit. Els drivers permeten fer jocs de prova customitzables i per tant han d'utilitzar objectes reals i no cal que utilitzin Stubs.

- ## Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input determinat i l'output esperat. De tal forma que al executar-lo amb l'input, podem comprovar si l'output obtingut és diferent a l'output esperat.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ## Operativa:

Un cop s'executa, s'imprimeix per pantalla un llistat de totes les funcions numerades que es poden testejar. Demana el nombre de la funció que es vol testejar. Si es prem 0, es testejen totes les funcions per ordre una a una. Si es prem finalitza el programa.

En aquest cas hi ha 8 funcions, numerades del 1 al 8.

1- testAuthorsControllerConstruct → Testeja la creació d'un AuthorsController

2- testAddAuthor → Testeja la funcionalitat d'afegir un autor.

3- testDelAuthor → Testeja la funcionalitat d'eliminar un autor.

4- testSize → Testeja la funcionalitat d'obtenir el número d'autors que hi ha al sistema.

5- testSearchAuthorsPrefix → Testeja la funcionalitat de buscar i obtenir un ArrayList<String> dels noms dels autors que comencin per un prefix donat.

6- testSearchAuthorDocuments → Testeja la funcionalitat de buscar i obtenir un ArrayList<String> dels títols dels documents d'un autor en concret.

7- testAddTitleAuthor → Testeja la funcionalitat d'afegir el títol d'un document a un autor.

Recordem que un Author guarda els titols dels seus documents i no els documents com a tal.

8- testDelTitleAuthor → Testeja la funcionalitat de borrar el títol del document del seu autor.

Per el test de cada funció, es van imprimint per pantalla diversos missatges i demana que s'introdueixin diversos camps que es van especificant.

8. FoldersController

- ## Objecte de la Prova:

L'objecte de la prova és la classe FoldersController, el subcontrolador que gestiona les carpetes. Comprovem que totes les funcions relacionades amb la inicialització del sistema de Carpetes així com l'intercanvi d'informació entre els objectes del Domini (Carpetes i Documents en aquest cas) i la comunicació amb el DomainController per interaccionar amb altres capes.

- ## Altres elements integrats en la Prova:

Per realitzar aquesta prova s'utilitzen les classes Folder i Document quan han estat provades anteriorment.

- ## Drivers:

S'ha desenvolupat un driver anomenat FoldersControllerDriver per provar aquesta classe de forma personalitzada.

La missió del driver és detectar sortides incorrectes de forma que no es llenci cap excepció ni generi una sortida incorrecta.

- ## Stubs:

—-

- ## Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures. De totes formes podríem dir que estem fent servir una Caixa gris ja que el propi Usuari pot provar les funcions amb el driver.

- ## Efectes estudiats

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ## Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els testos, en ordre.
1. testFolder → Test de la creadora.
2. testSetFolderName → Testeja la funcionalitat de canviar el nom a una carpeta.
3. testCreateFolder → Testeja la funcionalitat de crear una subCarpeta.
4. testAddDocument → Testeja la funcionalitat d'afegir un Document a la carpeta amb un Objecte Document.
5. testAddNonConstructedDocument → Testeja la funcionalitat d'afegir un Document a la carpeta amb els atributs necessaris d'un Objecte Document.
6. testDelDocument → Testeja la funcionalitat d'esborrar un Document de la carpeta.

7. testModifyContent → Testeja la funcionalitat de modificar el Contingut d'un Document.
8. testModifyAuthor → Testeja la funcionalitat de modificar l'Autor d'un Document.
9. testModifyTitle → Testeja la funcionalitat de modificar el títol d'un Document.
10. testProtectDocument → Testeja la funcionalitat de protegir un Document amb contrasenya.
11. testGetName → Testeja la funcionalitat d'obtenir el nom de la carpeta.
12. testGetDocumentsName → Testeja la funcionalitat d'obtenir el nom de tots els Documents de la carpeta.
13. testGetDocumentAmount → Testeja la funcionalitat d'obtenir el nombre de Documents de la carpeta.
14. testGetDocument → Testeja la funcionalitat d'obtenir el contingut d'un Document de la carpeta.
15. testGetMapsDoc → Testeja la funcionalitat d'obtenir l'estructura interna del vector de freqüències, necessari per les cerques.
16. testDocumentContained → Testeja la funcionalitat d'esbrinar si un document hi és a la carpeta.
17. testFolderContained → Testeja la funcionalitat d'esbrinar si la carpeta indicada és una subcarpeta d'aquesta.
18. Exit → Acaba l'execució del driver.

9. SearchController

- ### Objecte de la Prova:

L'objecte de la prova és la classe Search Controller, on es manté un historial de les expressions booleanes inscrites anteriorment, així com la consulta de tres cerques relacionades amb el contingut del document.

- ### Altres elements integrats en la Prova:

Per realitzar aquesta prova s'utilitzen les classes Document, Contingut i Boolean Expression, que han estat provades anteriorment, en el cas del Driver.

- ### Drivers:

S'ha desenvolupat un driver anomenat SearchControllerDriver per provar aquesta classe de forma personalitzada.

La missió del driver és detectar sortides incorrectes de forma que no salti cap excepció ni generi una sortida incorrecta.

- ### Stubs:

No es fan servir Stubs, ja que no es tracta d'una classe unitària.

- ### Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ### Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ### Efectes estudiats

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ### Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els testos, en ordre.
1. testConstructor → Comprova si s'ha pogut crear una nova instància de domain controllerd
2. testAddExpression → Comprova que s'ha guardat l'expressió booleana a l'historial.
3. testModifyExpression → Comprova que una expressió booleana existent de l'historial s'ha modificat per una altra correctament.
4. testDeleteExpression → Verifica que s'ha eliminat una expressió booleana que es trobava a l'historial.

5. testGetListExp → Testeja la funcionalitat de retornar una llista amb les expressions booleanes anteriors de l'usuari.
6. testBooleanExpressionSearch → Testeja la cerca de documents que contenen una frase que satisfà l'expressió booleana indicada per l'usuari.
7. testSearchDocuments → Testeja la cerca d'un nombre específic de documents que tenen major semblança al document indicat per l'usuari.
8. testSearchDocument → Testeja la cerca dels k documents més rellevants donada una llista de paraules.
9. Exit → Acaba l'execució del driver.

10. DomainController

● Objecte de la Prova:

L'objecte de la prova és la classe Folder, on comprovem tots els casos d'ús i com aquests s'han de relacionar amb les altres capes.

● Altres elements integrats en la Prova:

Per realitzar aquesta prova s'utilitzen les classes AuthorController, FolderController i SearchController quan han estat provades anteriorment, en el cas del Driver.
És necessari comentar que el JUnit no fa servir la mateixa dinàmica perquè entenem que els drivers permeten fer jocs de prova customitzables i per tant han d'utilitzar objectes reals mentre que els JUnit fan servir Stubs.

● Drivers:

S'ha desenvolupat un driver anomenat DomainControllerDriver per provar aquesta classe de forma personalitzada.

● Stubs:

–

● Fitxers de dades necessaris:

S'ha generat un Joc de proves amb un input i un output esperat. D'aquesta forma podem redireccionar l'entrada i comprovar si la sortida és igual a l'esperada.

- ## Valors estudiats:

Hem utilitzat el mètode caixa negra però provant situacions amb alta tendència a l'error, d'aquesta forma és el creador de la classe qui s'encarrega d'elaborar els jocs de prova, el Driver i el JUnit. Tot això un cop s'ha comprovat que la classe funciona per a assegurar-nos que segueix funcionant en modificacions futures.

- ## Efectes estudiats

Funcionalitats correctes dels objectes de la capa de Domini, es comprova si els Objectes realitzen la seva funció.

- ## Operativa:

El Driver va generant sortides per indicar a l'Usuari com continuar provant les diferents funcions i ofereix les següents possibilitats:

0. All → Executa tots els testos, en ordre.
1. testDomainController → Test de la creadora.
2. testImportDocument → Testeja la funcionalitat d'importar un document.
3. testExportDocument → Testeja la funcionalitat d'exportar un document.
4. testNewDocument → Testeja la funcionalitat d'afegir un document.
5. testModifyDocument → Testeja la funcionalitat de modificar un document.
6. testAuthorDocuments → Testeja la funcionalitat de buscar la llista de documents fets per un autor concret.
7. testSearchAuthors → Testeja la funcionalitat de buscar els autors que tinguin un prefix donat al seu nom.
8. testGetDocument → Testeja la funcionalitat d'obtenir un document seleccionat.
9. testAppearanceSearch → Testeja la funcionalitat de buscar els documents més semblants a un document seleccionat.
10. testBooleanExpressionSearch → Testeja la funcionalitat de buscar els documents que continguin alguna frase que compleixi l'expressió booleana.
11. testDocumentsQuery → Testeja la funcionalitat de buscar els documents més rellevants donat una llista de paraules.
12. testSaveDocument → Testeja la funcionalitat d'emmagatzemar un document.
13. testDeleteDocument → Testeja la funcionalitat d'eliminar un document del sistema.

14. testProtectDocument → Testeja la funcionalitat de protegir un document seleccionat.
15. testNewFolder → Testeja la funcionalitat d'afegir una nova carpeta.
16. Exit → Acaba l'execució del driver.