Tesztelés jegyzőkönyv

# Tesztelés során felhasznált eszközök

Az egységtesztelést a JUnit egységtesztelő keretrendszer segítségével végeztem. A JUnit az xUnit egységtesztelő keretrendszerek családjába tartozik. A nevében a „J” utal arra, hogy Java programozási nyelven íródott forráskódok teszteléséhez készült, és a teszteket szintén Java programozási nyelven kell implementálni. A JUnit támogatja az iteratív és inkrementális tesztelési és fejlesztési irányelveket. Ilyen például a teszt vezérelt fejlesztést (TDD – Test Driven Development), amely azt jelenti, hogy az új funkció lefejlesztése előtt, először a kódot tesztelő osztályokat implementáljuk, és csak ezután fejlesztjük le a tényleges funkciót. A JUnit-nak már több verziója is elérhető. A tesztelést a legújabb verzióval (JUnit 5) valósítottam meg.

A tesztek determinisztikus lefutását Mockito segítségével biztosítottam. Mockito egy open source tesztelő keretrendszer, amely lehetővé teszi a Java programozási nyelven íródott egységtesztekhez szükséges objektumok mock megfelelőinek létrehozását. Ezen kívül Mockito használatával előre meg tudjuk határozni az egyes külső, független metódusok viselkedését. Ezáltal az automatizált teszt minden lefutás során ugyanazt az eredményt produkálja.

# Tesztelt osztályok

*org.jkiss.dbeaver.utils.GeneralUtils.java*: A GeneralUtils.java osztály tesztelését választottam. Ez az osztály definiál nagyon sok általános metódust és éppen ezért fontosak mert sokat használtak a progam működése során.

# Létrehozott tesztcsomag bemutatása

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java: A GeneralUtils osztalyban talalható convetToString(), makeDisplayString(), convertToBytes(), varriablePattern(), GetRootCause(), isVarriablePattern(), metódusokat teszteli. A convertToString() egy byte tömböt alakít szöveggé, úgy hogy kap egy offsetet és egy hosszat is hozzá hogy hol kezdje és mekkora részt alakítson át belőle. A makeDisplayString() felel azért hogy bármely obijektumot ki lehessen íratni. A convertToBytes() egy Stringből készíti el a hozzá tartozó byte tömböt. A varriablePattern() egy nevet kap és ahhoz létrehozza a pattern stringet.. A getRootCause() visszadja egy dobott hiba eredetét. Az Is VarriablePattern() elnevezési konvenciót ellenőriz.*

# Létrehozott tesztek pontos megadása

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testCovertToString2 ()*:
   * Arrange: 18-22. sorok
   * Act: 23. sor
   * Assert: 23. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testMakeDisplayString()*:
   * Arrange: 59. sor
   * Act: 60. sor
   * Assert: 60. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testConvertToBytes2()*:
   * Arrange: 34-35. sorok
   * Act: 36. sor
   * Assert: 37. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testConvertToString()*:
   * Arrange: 11-14. sor
   * Act: 15. sor
   * Assert: 15. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testConvertToBytes()*:
   * Arrange: 27-28. sor
   * Act: 29. sor
   * Assert: 30. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testVarriablePattern ()*:
   * Arrange: 41. sor
   * Act: 42. sor
   * Assert: 43. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testGetRootCause ()*:
   * Arrange: 70. sorok
   * Act: 71. sor
   * Assert: 72. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testMakeDisplayString2()*:
   * Arrange: 64. sorok
   * Act: 65. sor
   * Assert: 66. sor

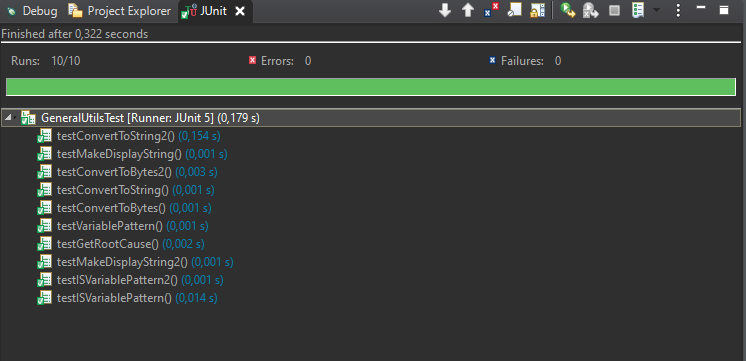
*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testIsVarriablePattern2 ()*:
   * Arrange: 53. sorok
   * Act 54. sor
   * Assert: 55. sor

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*:

1. *testIsVarriablePattern ()*:
   * Arrange: 47. sorok
   * Act: 48. sor
   * Assert: 49. sor

# Végrehajtott tesztek sikerességének megállapítása

*org.jkiss.dbeaver.utils.GeneralUtilsTest.java*: Mind a tíz teszten sikeresen átment. 

Lefedettségi jelentés

# Lefedettség mérés során felhasznált eszközök

A kód lefedettséget SonarQube segítségével mértem. A SonarQube egy open source szoftver, amely a forráskód minőségének ellenőrzésére és követésére szolgál. A SonarQube képes a Java programozási nyelven íródott forráskódok feldolgozására. Különböző grafikonokat készít a forráskód minőségi kritériumairól. Ilyen kritérium lehet például a duplikált kód mennyisége, a kód szabványok betartása, unit tesztek mennyisége és az általuk lefedett forráskód, a forráskód komplexitása, esetleges hibák száma, dokumentáltság.

# Lefedettségi érték indoklása

Az én választásom az *org.jkiss.dbeaver.model* csomagra esett. A csomag lefedettsége tesztek hiányában 0.0%

*org.jkiss.dbeaver.model csomag méret metrikái:*

1. Lines of Code: 21,411
2. Lines: 79,317
3. Statements: 17,311
4. Functions: 6,407
5. Classes: 784
6. Files: 710
7. Comment Lines: 3641
8. Comments (%): 6.6%

*org.jkiss.dbeaver.core* csomag tényleges lefedettsége:

1. Coverage: 0.0%
2. Lines to Cover: 21,411
3. Uncovered Lines: 21,411
4. Lines Coverage: 0.0%

A model felel a legtöbb alapműveletér és ez a lefedettség nagyon rosszat jelent a megbízhatóságáról.

# Erősen és kevésbe lefedett kódelemek bemutatása

*org.jkiss.dbeaver.model.DBUtils*.java méret metrikái:

1. Lines of Code: 1,783
2. Lines: 2,086
3. Statements: 959
4. Functions: 125
5. Classes:
6. Files: 1
7. Comment Lines: 96
8. Comments (%): 5.1%

*org.jkiss.dbeaver.model.DBUtils*.java tényleges lefedettsége:

1. Coverage: 0.0%
2. Lines to Cover: 1,022
3. Uncovered Lines: 1,022
4. Lines Coverage: 0.0%

*org.jkiss.dbeaver.model.virtual.DBVEntity*.java méret metrikái:

1. Lines of Code: 603
2. Lines: 704
3. Statements: 311
4. Functions: 57
5. Classes: 1
6. Files: 1
7. Comment Lines: 10
8. Comments (%): 1.6%

*org.jkiss.dbeaver.core.ui.dialogs.driver.DriverManagerDialog*.java tényleges lefedettsége:

1. Coverage: 0.0%
2. Lines to Cover: 370
3. Uncovered Lines: 370
4. Lines Coverage: 0.0%

# Összegzés

Az egész projekt lefedettsége szörnyű elsőnek ezen kellene javítani. A projektben szinte semmi sincs kommentálva vagy dokumentálva így nehéz dolgozni vele nagy eséllyel emiatt van ilyen állapotban. A dokumentációk alapján lehetne javítani a projekt minőségét és azzal a tesztelést is elérhetőbbé tenni. Így ezen állapotában túl nehezen tesztelehető a projekt.