File2DDL Converter

Testing Plan

Hassan Ndow

Date: 02-24-2017

Version: 1.0

# **Testing Plan**

The objective of this test plan is to create test cases and run test code to verify that the current application can read in a file created by Edge Diagrammer and be able to generate the appropriate SQL statements needed to create tables in a MySQL database.

Once this phase is completed the application will be refactored to become an extensible and reusable application with expanded functionality. It will be able read in schema description files of different formats as well as produce the expected DDL file with respect to its required database. For all intents and purposes, specific file formats and databases left ambiguous in the original specifications document shall be referred to using the ‘Generic’ descriptor.

The application system’s refactoring will be developed using the Factory, Strategy and Singleton patterns. This is so that the system will be able to readily adapt to the various I/O formats as well as provide a simple interface to handle these functions.

# 

# **Test Cases**

## EdgeReader

Test readFile(in File file):void

|  |  |
| --- | --- |
| Title | Case 1: Read and parse a valid Edge diagramming file |
| Description | EdgeReader receives a .edge file passed in as parameter and reads and parses the file into prepared tables and fields to be stored in its respective ORM classes |
| Pre-Conditions | * \*.edg file exists * \*.edg file is valid * contains one or more tables |
| Test Steps | User inputs \*.edg file |
| Expected Results | \*.edg file is read, parsed, and stored in ORM Table classes |

|  |  |
| --- | --- |
| Title | Case 2: Read and parse an Edge diagramming file containing invalid contents |
| Description | EdgeReader receives an invalid .edge file passed in as parameter and throws an exception error |
| Pre-Conditions | * \*.edg file exists * \*.edg file is invalid |
| Test Steps | User inputs \*.edg file |
| Expected Results | \*.edg file is read, parsed, and throws a MalformedInputException |

|  |  |
| --- | --- |
| Title | Case 3: Read and parse a non Edge diagramming file |
| Description | EdgeReader receives a file whose extension is not .edg but passed in as parameter and throws an exception error |
| Pre-Conditions | * \*.edg file does not exists |
| Test Steps | User inputs non .edg file |
| Expected Results | FileNotFoundException error is thrown |

## XMLReader

Test readFile(in File file):void

|  |  |
| --- | --- |
| Title | Case 1: Read and parse a valid XML file |
| Description | XMLReader receives a .xml file passed in as parameter and reads and parses the file into prepared data to be stored in the ORM Table classes |
| Pre-Conditions | * \*.xml file exists * \*.xml file is valid * contains one or more tables |
| Test Steps | User inputs \*.xml file |
| Expected Results | \*.xml file is read, parsed, and stored in ORM Table classes |

|  |  |
| --- | --- |
| Title | Case 2: Read and parse an XML file containing invalid contents |
| Description | XMLReader receives an invalid .xml file passed in as parameter and throws an exception error |
| Pre-Conditions | * \*.xml file exists * \*.xml file is invalid |
| Test Steps | User inputs \*.xml file |
| Expected Results | \*.xml file is read, parsed, and throws a MalformedInputException |

|  |  |
| --- | --- |
| Title | Case 3: Read and parse a non XML file |
| Description | XMLReader receives a file whose extension is not .xml but passed in as parameter and throws an exception error |
| Pre-Conditions | * \*.xml file does not exists |
| Test Steps | User inputs non .xml file |
| Expected Results | FileNotFoundException error is thrown |

## GenericReader

Test readFile(in File file):void

|  |  |
| --- | --- |
| Title | Case 1: Read and parse a valid ‘Generic’ file |
| Description | GenericReader receives a .generic file passed in as parameter and reads and parses the file into prepared tables and fields to be stored in its respective ORM classes |
| Pre-Conditions | * \*.generic file exists * \*.generic file is valid * contains one or more tables |
| Test Steps | User inputs \*.generic file |
| Expected Results | \*.generic file is read, parsed, and stored in ORM Table classes |

|  |  |
| --- | --- |
| Title | Case 2: Read and parse a ‘Generic’ file containing invalid contents |
| Description | GenericReader receives an invalid .generic file passed in as parameter and throws an exception error |
| Pre-Conditions | * \*.generic file exists * \*.generic file is invalid |
| Test Steps | User inputs \*.generic file |
| Expected Results | \*.generic file is read, parsed, and throws a MalformedInputException |

|  |  |
| --- | --- |
| Title | Case 3: Read and parse a non ‘Generic’ file |
| Description | GenericReader receives a file whose extension is not .generic but passed in as parameter and throws an exception error |
| Pre-Conditions | * \*.generic file does not exists |
| Test Steps | User inputs non .generic file |
| Expected Results | FileNotFoundException error is thrown |

## MySQLDDLWriter

Test createDDL():String

|  |  |
| --- | --- |
| Title | Case 1: Generate MySQL statements from ORM Table objects passed to MySQLDDLWriter’s constructor |
| Description | MySQLDDLWriter receives a list of ORM table objects passed in as a parameter in its constructor and generates MySQL DDL statements from it. |
| Pre-Conditions | * ORM Table object exists |
| Test Steps | MySQLDDLWriter has an ORM Table object passed in its constructor |
| Expected Results | Appropriate MySQL DDL statements are written and returned as a String |

|  |  |
| --- | --- |
| Title | Case 2: Generate MySQL statements when no ORM Table objects are passed to MySQLDDLWriter’s constructor |
| Description | MySQLDDLWriter does not receive a list of ORM table objects passed in as a parameter in its constructor but attempts to generate MySQL DDL statements regardless. |
| Pre-Conditions | * ORM Table object does not exists |
| Test Steps | MySQLDDLWriter has no ORM Table object passed in its constructor and relies on its default constructor. |
| Expected Results | IllegalArgumentException error is thrown |

## OracleDDLWriter

Test createDDL():String

|  |  |
| --- | --- |
| Title | Case 1: Generate Oracle statements from ORM Table objects passed to OracleDDLWriter constructor |
| Description | OracleDDLWriter receives a list of ORM table objects passed in as a parameter in its constructor and generates Oracle DDL statements from it. |
| Pre-Conditions | * ORM Table object exists |
| Test Steps | OracleDDLWriter has an ORM Table object passed in its constructor |
| Expected Results | Appropriate Oracle DDL statements are written and returned as a String |

|  |  |
| --- | --- |
| Title | Case 2: Generate Oracle statements when no ORM Table objects are passed to OracleDDLWriter’s constructor |
| Description | OracleDDLWriter does not receive a list of ORM table objects passed in as a parameter in its constructor but attempts to generate Oracle DDL statements regardless. |
| Pre-Conditions | * ORM Table object does not exists |
| Test Steps | OracleDDLWriter has no ORM Table object passed in its constructor and relies on its default constructor. |
| Expected Results | IllegalArgumentException error is thrown |

## GenericDDLWriter

Test createDDL():String

|  |  |
| --- | --- |
| Title | Case 1: Generate ‘Generic’ statements from ORM Table objects passed to GenericDDLWriter constructor |
| Description | GenericDDLWriter receives a list of ORM table objects passed in as a parameter in its constructor and generates ‘Generic’ DDL statements from it. |
| Pre-Conditions | * ORM Table object exists |
| Test Steps | GenericDDLWriter has an ORM Table object passed in its constructor |
| Expected Results | Appropriate ‘Generic’ DDL statements are written and returned as a String |

|  |  |
| --- | --- |
| Title | Case 2: Generate ‘Generic’ statements when no ORM Table objects are passed to GenericDDLWriter’s constructor |
| Description | GenericDDLWriter does not receive a list of ORM table objects passed in as a parameter in its constructor but attempts to generate ‘Generic’ DDL statements regardless. |
| Pre-Conditions | * ORM Table object does not exists * GenericDDLWriter default constructor used |
| Test Steps |  |
| Expected Results | IllegalArgumentException error is thrown |

# **Test Code**

# **Diagrams**

