

TestResultDBAccess

v. 0.1.0

Tran Duy Ngoan

25.04.2024

Contents

1	Introduction	1
2	Description	2
2.0.1	DBAccessFactory	2
2.0.2	DBAccess Interface	2
2.0.3	DirectDBAccess	2
2.0.4	RestApiDBAccess	2
3	db_accesss_interface.py	3
3.1	Class: DBAccess	3
3.1.1	Method: connect	4
3.1.2	Method: disconnect	4
3.1.3	Method: commit	4
3.1.4	Method: arGetCategories	4
3.1.5	Method: bExistingResultID	4
3.1.6	Method: sGetLatestFileID	4
3.1.7	Method: sCreateNewTestResult	4
3.1.8	Method: nCreateNewFile	4
3.1.9	Method: vCreateNewHeader	4
3.1.10	Method: nCreateNewSingleTestCase	4
3.1.11	Method: nCreateNewTestCase	4
3.1.12	Method: vCreateAbortReason	4
3.1.13	Method: vCreateCCRdata	4
3.1.14	Method: vCreateTags	4
3.1.15	Method: vCreateReanimation	4
3.1.16	Method: vSetCategory	4
3.1.17	Method: vUpdateFileEndTime	4
3.1.18	Method: vUpdateResultEndTime	4
3.1.19	Method: vUpdateEvtbl	4
3.1.20	Method: vUpdateEvtbls	4
3.1.21	Method: vFinishTestResult	4
4	direct_db_accesss.py	5
4.1	Class: DirectDBAccess	5
4.1.1	Method: connect	5
4.1.2	Method: commit	6
4.1.3	Method: disconnect	6
4.1.4	Method: cleanAllTables	6

4.1.5	Method: sCreateNewTestResult	6
4.1.6	Method: nCreateNewFile	7
4.1.7	Method: vCreateNewHeader	8
4.1.8	Method: nCreateNewSingleTestCase	10
4.1.9	Method: nCreateNewTestCase	11
4.1.10	Method: vCreateTags	13
4.1.11	Method: vSetCategory	13
4.1.12	Method: vUpdateStartEndTime	13
4.1.13	Method: arGetCategories	14
4.1.14	Method: vCreateAbortReason	14
4.1.15	Method: vCreateReanimation	14
4.1.16	Method: vCreateCCRdata	14
4.1.17	Method: vFinishTestResult	15
4.1.18	Method: vUpdateEvtbls	15
4.1.19	Method: vUpdateEvtbl	15
4.1.20	Method: vEnableForeignKeyCheck	15
4.1.21	Method: sGetLatestFileID	16
4.1.22	Method: vUpdateFileEndTime	16
4.1.23	Method: vUpdateResultEndTime	16
4.1.24	Method: bExistingResultID	16
4.1.25	Method: arGetProjectVersionSWByID	17
5	rest_api_db_access.py	18
5.1	Class: RestApiDBAccess	18
5.1.1	Method: get_certs_file	19
5.1.2	Method: encrypt_password	19
5.1.3	Method: connect	19
5.1.4	Method: disconnect	19
5.1.5	Method: arGetCategories	19
5.1.6	Method: bExistingResultID	19
5.1.7	Method: sGetLatestFileID	19
5.1.8	Method: arGetProjectVersionSWByID	19
5.1.9	Method: sCreateNewTestResult	19
5.1.10	Method: nCreateNewFile	19
5.1.11	Method: vCreateNewHeader	19
5.1.12	Method: nCreateNewSingleTestCase	19
5.1.13	Method: nCreateNewTestCase	19
5.1.14	Method: vCreateAbortReason	19
5.1.15	Method: vCreateCCRdata	19
5.1.16	Method: vCreateTags	19
5.1.17	Method: vCreateReanimation	19
5.1.18	Method: vSetCategory	19
5.1.19	Method: vUpdateFileEndTime	19
5.1.20	Method: vUpdateResultEndTime	19
5.1.21	Method: vFinishTestResult	19
5.1.22	Method: vUpdateEvtbl	19

5.1.23 Method: vUpdateEvtbls	19
6 DBAccessFactory.py	20
6.1 Class: DBAccessFactory	20
6.1.1 Method: create	20
7 Appendix	21
8 History	22

Chapter 1

Introduction

The **TestResultDBAccess** package is designed to streamline the process of importing test data into the database of a test result web application.

This package offers two distinct access methods: **Direct Access** and **REST API Access**.

Users can seamlessly choose between these methods based on their preferences and requirements.

- **Direct Access:** This method provides a direct connection to the database using the SQL protocol, allowing efficient data retrieval and import operations.
- **REST API Access:** This method interacts with the web application via HTTP/HTTPS protocol, providing flexibility and compatibility with modern web architectures.

The package's DB Access Factory facilitates the creation of instances for both **Direct Access** and **REST API Access**, ensuring a smooth and adaptable workflow for importing test data into the target database.

Whether users prefer direct database interaction or web-based API access, **TestResultDBAccess** offers a comprehensive solution to meet diverse testing needs.

Chapter 2

Description

The `TestResultDBAccess` package offers a flexible and efficient way to interact with the database of a test result web application.

Users can choose between direct database access and REST API access, depending on their specific needs and preferences.

2.0.1 DBAccessFactory

The **DBAccessFactory** class is responsible for creating instances of the **DBAccess** interface based on the specified access method.

It provides the following method:

- **create(access_method: str): DBAccess** - Creates an instance of **DBAccess** based on the specified access method (`rest` or `db`).

2.0.2 DBAccess Interface

The **DBAccess** interface defines a set of methods for interacting with the database. These methods include:

- **connect(host: str, user: str, passwd: str, database: str)** - Establishes a connection to the database.
- **disconnect()** - Destroys the connection to the database.
- Various methods for retrieving, creating, updating, and finishing test result data in the database, such as **arGetCategories()**, **sGetLatestFileID()**, **vCreateTags()**, **vUpdateResultEndTime()**, and **vFinishTestResult()**.

2.0.3 DirectDBAccess

The **DirectDBAccess** class implements the **DBAccess** interface for direct database access. It provides methods for establishing and destroying connections, as well as implementing the various data manipulation methods defined in **DBAccess**.

2.0.4 RestApiDBAccess

The **RestApiDBAccess** class implements the **DBAccess** interface for REST API access to the database. Similar to **DirectDBAccess**, it provides methods for connecting, disconnecting, and implementing data manipulation methods through the REST API.

Chapter 3

db_accesss_interface.py

3.1 Class: DBAccess

Imported by:

```
from TestResultDBAccess.DBAccess.db-accessss-interface import DBAccess
```

- 3.1.1 Method: connect
- 3.1.2 Method: disconnect
- 3.1.3 Method: commit
- 3.1.4 Method: arGetCategories
- 3.1.5 Method: bExistingResultID
- 3.1.6 Method: sGetLatestFileID
- 3.1.7 Method: sCreateNewTestResult
- 3.1.8 Method: nCreateNewFile
- 3.1.9 Method: vCreateNewHeader
- 3.1.10 Method: nCreateNewSingleTestCase
- 3.1.11 Method: nCreateNewTestCase
- 3.1.12 Method: vCreateAbortReason
- 3.1.13 Method: vCreateCCRdata
- 3.1.14 Method: vCreateTags
- 3.1.15 Method: vCreateReanimation
- 3.1.16 Method: vSetCategory
- 3.1.17 Method: vUpdateFileEndTime
- 3.1.18 Method: vUpdateResultEndTime
- 3.1.19 Method: vUpdateEvtbl
- 3.1.20 Method: vUpdateEvtbls
- 3.1.21 Method: vFinishTestResult

Chapter 4

direct_db_access.py

4.1 Class: DirectDBAccess

Imported by:

```
from TestResultDBAccess.DBAccess.direct_db_access import DirectDBAccess
```

DirectDBAccess class play a role as mysqlclient and provide methods to interact with TestResultWebApp's database.

4.1.1 Method: connect

Connect to the database with provided authentication and db info.

Arguments:

- `host`
/ *Condition*: required / *Type*: str /
URL which is hosted the TestResultWebApp's database.
- `user`
/ *Condition*: required / *Type*: str /
User name for database authentication.
- `passwd`
/ *Condition*: required / *Type*: str /
User's password for database authentication.
- `database`
/ *Condition*: required / *Type*: str /
Database name.
- `charset`
/ *Condition*: optional / *Type*: str / *Default*: 'utf8' /
The connection character set.
- `use_unicode`
/ *Condition*: optional / *Type*: bool / *Default*: True /
If True, CHAR and VARCHAR and TEXT columns are returned as Unicode strings, using the configured character set.

Returns:

(no returns)

4.1.2 Method: commit

Commit changes within transaction.

Arguments:

(no arguments)

Returns:

(no returns)

4.1.3 Method: disconnect

Disconnect from TestResultWebApp's database.

Arguments:

(no arguments)

Returns:

(no returns)

4.1.4 Method: cleanAllTables

Delete all table data. Please be careful before calling this method.

Arguments:

(no arguments)

Returns:

(no returns)

4.1.5 Method: sCreateNewTestResult

Creates a new test result in `tbl_result`. This is the main table which is linked to all other data by means of `test_result_id`.

Arguments:

- `tbl_prj_project`
/ Condition: required / Type: str /
Project information.
- `tbl_prj_variant`
/ Condition: required / Type: str /
Variant information.
- `tbl_prj_branch`
/ Condition: required / Type: str /
Branch information.
- `tbl_test_result_id`
/ Condition: required / Type: str /
UUID of test result.
- `tbl_result_interpretation`
/ Condition: required / Type: str /
Result interpretation.
- `tbl_result_time_start`
/ Condition: required / Type: str /
Test result start time as format `%Y-%m-%d %H:%M:%S`.

- `.tbl_result_time_end`
/ *Condition*: required / *Type*: str /
Test result end time as format %Y-%m-%d %H:%M:%S.
- `.tbl_result_version_sw_target`
/ *Condition*: required / *Type*: str /
Software version information.
- `.tbl_result_version_sw_test`
/ *Condition*: required / *Type*: str /
Test version information.
- `.tbl_result_version_target`
/ *Condition*: required / *Type*: str /
Hardware version information.
- `.tbl_result_jenkinsurl`
/ *Condition*: required / *Type*: str /
Jenkinsurl in case test result is executed by jenkins.
- `.tbl_result_reporting_qualitygate`
/ *Condition*: required / *Type*: str /
Qualitygate information for reporting.

Returns:

- `.tbl_test_result_id`
/ *Type*: str /
`test_result_id` of new test result.

4.1.6 Method: nCreateNewFile

Create new file entry in `tbl_file` table.

Arguments:

- `.tbl_file_name`
/ *Condition*: required / *Type*: str /
File name information.
- `.tbl_file_tester_account`
/ *Condition*: required / *Type*: str /
Tester account information.
- `.tbl_file_tester_machine`
/ *Condition*: required / *Type*: str /
Test machine information.
- `.tbl_file_time_start`
/ *Condition*: required / *Type*: str /
Test file start time as format %Y-%m-%d %H:%M:%S.
- `.tbl_file_time_end`
/ *Condition*: required / *Type*: str /
Test file end time as format %Y-%m-%d %H:%M:%S.

- `_tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result for linking to `tbl_result` table.
- `_tbl_file_origin`
/ *Condition*: required / *Type*: str /
Origin (test framework) of test file. Default is "ROBFW"

Returns:

- `iInsertedID`
/ *Type*: int /
ID of new entry.

4.1.7 Method: vCreateNewHeader

Create a new header entry in `tbl_file.header` table which is linked with the file.

Arguments:

- `_tbl_file_id`
/ *Condition*: required / *Type*: int /
File ID information.
- `_tbl_header_testtoolconfiguration_testtoolname`
/ *Condition*: required / *Type*: str /
Test tool name.
- `_tbl_header_testtoolconfiguration_testtoolversionstring`
/ *Condition*: required / *Type*: str /
Test tool version.
- `_tbl_header_testtoolconfiguration_projectname`
/ *Condition*: required / *Type*: str /
Project name.
- `_tbl_header_testtoolconfiguration_logfileencoding`
/ *Condition*: required / *Type*: str /
Encoding of logfile.
- `_tbl_header_testtoolconfiguration_pythonversion`
/ *Condition*: required / *Type*: str /
Python version info.
- `_tbl_header_testtoolconfiguration_testfile`
/ *Condition*: required / *Type*: str /
Test file name.
- `_tbl_header_testtoolconfiguration_logfilepath`
/ *Condition*: required / *Type*: str /
Path to log file.
- `_tbl_header_testtoolconfiguration_logfilemode`
/ *Condition*: required / *Type*: str /
Mode of log file.

- `.tbl_header.testtoolconfiguration.ctrlfilepath`
/ *Condition*: required / *Type*: str /
Path to control file.
- `.tbl_header.testtoolconfiguration.configfile`
/ *Condition*: required / *Type*: str /
Path to configuration file.
- `.tbl_header.testtoolconfiguration.confname`
/ *Condition*: required / *Type*: str /
Configuration name.
- `.tbl_header.testfileheader.author`
/ *Condition*: required / *Type*: str /
File author.
- `.tbl_header.testfileheader.project`
/ *Condition*: required / *Type*: str /
Project information.
- `.tbl_header.testfileheader.testfiledate`
/ *Condition*: required / *Type*: str /
File creation date.
- `.tbl_header.testfileheader.version.major`
/ *Condition*: required / *Type*: str /
File major version.
- `.tbl_header.testfileheader.version.minor`
/ *Condition*: required / *Type*: str /
File minor version.
- `.tbl_header.testfileheader.version.patch`
/ *Condition*: required / *Type*: str /
File patch version.
- `.tbl_header.testfileheader.keyword`
/ *Condition*: required / *Type*: str /
File keyword.
- `.tbl_header.testfileheader.shortdescription`
/ *Condition*: required / *Type*: str /
File short description.
- `.tbl_header.testexecution.useraccount`
/ *Condition*: required / *Type*: str /
Tester account who run the execution.
- `.tbl_header.testexecution.computername`
/ *Condition*: required / *Type*: str /
Machine name which is executed on.
- `.tbl_header.testrequirements.documentmanagement`
/ *Condition*: required / *Type*: str /
Requirement management information.

- `tbl_header.testrequirements.testenvironment`
/ *Condition*: required / *Type*: str /
Requirement environment information.
- `tbl_header.testbenchconfig.name`
/ *Condition*: required / *Type*: str /
Testbench configuration name.
- `tbl_header.testbenchconfig.data`
/ *Condition*: required / *Type*: str /
Testbench configuration data.
- `tbl_header.preprocessor.filter`
/ *Condition*: required / *Type*: str /
Preprocessor filter information.
- `tbl_header.preprocessor.parameters`
/ *Condition*: required / *Type*: str /
Preprocessor parameters definition.

Returns:*(no returns)***4.1.8 Method: nCreateNewSingleTestCase**Create single testcase entry in `tbl_case` table immediately.**Arguments:**

- `tbl_case.name`
/ *Condition*: required / *Type*: str /
Test case name.
- `tbl_case.issue`
/ *Condition*: required / *Type*: str /
Test case issue ID.
- `tbl_case.tcid`
/ *Condition*: required / *Type*: str /
Test case ID (used for testmanagement tool).
- `tbl_case.fid`
/ *Condition*: required / *Type*: str /
Test case requirement (function) ID.
- `tbl_case.testnumber`
/ *Condition*: required / *Type*: int /
Order of test case in file.
- `tbl_case.repeatcount`
/ *Condition*: required / *Type*: int /
Test case repeatition count.
- `tbl_case.component`
/ *Condition*: required / *Type*: str /
Component which test case is belong to.

- `tbl.case.time_start`
/ *Condition*: required / *Type*: str /
Test case start time as format %Y-%m-%d %H:%M:%S.
- `tbl.case.result_main`
/ *Condition*: required / *Type*: str /
Test case main result.
- `tbl.case.result_state`
/ *Condition*: required / *Type*: str /
Test case completion state.
- `tbl.case.result_return`
/ *Condition*: required / *Type*: int /
Test case result code (as integer).
- `tbl.case.counter_resets`
/ *Condition*: required / *Type*: int /
Counter of target reset within test case execution.
- `tbl.case.lastlog`
/ *Condition*: required / *Type*: str /
Traceback information when test case is failed.
- `tbl.test.result_id`
/ *Condition*: required / *Type*: str /
UUID of test result for linking to file in `tbl.result` table.
- `tbl.file.id`
/ *Condition*: required / *Type*: int /
Test file ID for linking to file in `tbl.file` table.

Returns:

- `iInsertedID`
/ *Type*: int /
ID of new entry.

4.1.9 Method: `nCreateNewTestCase`

Create bulk of test case entries: new test cases are buffered and inserted as bulk.

Once `_NUM_BUFFERD_ELEMENTS_FOR_EXECUTEMANY` is reached, the creation query is executed.

Arguments:

- `tbl.case.name`
/ *Condition*: required / *Type*: str /
Test case name.
- `tbl.case.issue`
/ *Condition*: required / *Type*: str /
Test case issue ID.
- `tbl.case.tcid`
/ *Condition*: required / *Type*: str /
Test case ID (used for testmanagement tool).

- `.tbl_case_fid`
/ *Condition*: required / *Type*: str /
Test case requirement (function) ID.
- `.tbl_case_testnumber`
/ *Condition*: required / *Type*: int /
Order of test case in file.
- `.tbl_case_repeatcount`
/ *Condition*: required / *Type*: int /
Test case repetition count.
- `.tbl_case_component`
/ *Condition*: required / *Type*: str /
Component which test case is belong to.
- `.tbl_case_time_start`
/ *Condition*: required / *Type*: str /
Test case start time as format %Y-%m-%d %H:%M:%S.
- `.tbl_case_result_main`
/ *Condition*: required / *Type*: str /
Test case main result.
- `.tbl_case_result_state`
/ *Condition*: required / *Type*: str /
Test case completion state.
- `.tbl_case_result_return`
/ *Condition*: required / *Type*: int /
Test case result code (as integer).
- `.tbl_case_counter_resets`
/ *Condition*: required / *Type*: int /
Counter of target reset within test case execution.
- `.tbl_case_lastlog`
/ *Condition*: required / *Type*: str /
Traceback information when test case is failed.
- `.tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result for linking to file in `tbl_result` table.
- `.tbl_file_id`
/ *Condition*: required / *Type*: int /
Test file ID for linking to file in `tbl_file` table.

Returns:*(no returns)*

4.1.10 Method: vCreateTags

Create tag entries.

Arguments:

- `.tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.
- `.tbl_usr_result_tags`
/ *Condition*: required / *Type*: str /
User tags information.

Returns:

(no returns)

4.1.11 Method: vSetCategory

Create category entry.

Arguments:

- `.tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.
- `.tbl_result_category_main`
/ *Condition*: required / *Type*: str /
Category information.

Returns:

(no returns)

4.1.12 Method: vUpdateStartTimeEndTime

Create start-end time entry.

Arguments:

- `.tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.
- `.tbl_result_time_start`
/ *Condition*: required / *Type*: str /
Result start time as format %Y-%m-%d %H:%M:%S.
- `.tbl_result_time_end`
/ *Condition*: required / *Type*: str /
Result end time as format %Y-%m-%d %H:%M:%S.

Returns:

(no returns)

4.1.13 Method: arGetCategories

Get existing categories.

Arguments:

(no arguments)

Returns:

- `arCategories`
/ *Type*: list /
List of existing categories.

4.1.14 Method: vCreateAbortReason

Create abort reason entry.

Arguments:

- `_tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.
- `_tbl_abort_reason`
/ *Condition*: required / *Type*: str /
Abort reason.
- `_tbl_abort_message`
/ *Condition*: required / *Type*: str /
Detail message of abort.

Returns:

(no returns)

4.1.15 Method: vCreateReanimation

Create reanimation entry.

Arguments:

- `_tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.
- `_tbl_num_of_reanimation`
/ *Condition*: required / *Type*: int /
Counter of target reanimation during execution.

Returns:

(no returns)

4.1.16 Method: vCreateCCRdata

Create CCR data per test case.

Arguments:

- `_tbl_test_case_id`
/ *Condition*: required / *Type*: int /
test case ID.

- `lCCRdata`
/ *Condition*: required / *Type*: list /
list of CCR data.

Returns:

(no returns)

4.1.17 Method: vFinishTestResult

Finish upload:

- First do bulk insert of rest of test cases if buffer is not empty.
- Then set state to "new report".

Arguments:

- `_tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.

Returns:

(no returns)

4.1.18 Method: vUpdateEvtbls

Call `update_evtbls` stored procedure.

Arguments:

(no arguments)

Returns:

(no returns)

4.1.19 Method: vUpdateEvtbl

Call `update_evtbl` stored procedure to update provided `test_result_id`.

Arguments:

- `_tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.

Returns:

(no returns)

4.1.20 Method: vEnableForeignKeyCheck

Switch `foreign_key_checks` flag.

Arguments:

- `enable`
/ *Condition*: optional / *Type*: bool / *Default*: True /
If True, enable foreign key constraint.

Returns:

(no returns)

4.1.21 Method: sGetLatestFileID

Get latest file ID from `tbl_file` table.

Arguments:

- `tbl_test_result_id`
/ *Condition*: required / *Type*: str /
UUID of test result.

Returns:

- `tbl_file_id`
/ *Type*: int /
File ID.

4.1.22 Method: vUpdateFileEndTime

Update test file end time.

Arguments:

- `tbl_file_id`
/ *Condition*: required / *Type*: int /
File ID to be updated.
- `tbl_file_time_end`
/ *Condition*: required / *Type*: str /
File end time as format `%Y-%m-%d %H:%M:%S`.

Returns:

(no returns)

4.1.23 Method: vUpdateResultEndTime

Update test result end time.

Arguments:

- `tbl_test_result_id`
/ *Condition*: required / *Type*: str /
Result UUID to be updated.
- `tbl_result_time_end`
/ *Condition*: required / *Type*: str /
Result end time as format `%Y-%m-%d %H:%M:%S`.

Returns:

(no returns)

4.1.24 Method: bExistingResultID

Verify the given test result UUID is existing in `tbl_result` table or not.

Arguments:

- `tbl_test_result_id`
/ *Condition*: required / *Type*: str /
Result UUID to be verified.

Returns:

- `bExisting`
/ *Type*: bool /
True if test result UUID is already existing.

4.1.25 Method: `arGetProjectVersionSWByID`

Get the project and version_sw information of given `test_result_id`

Arguments:

- `_tbl_test_result_id`
/ *Condition*: required / *Type*: str /
Result UUID to be get the information.

Returns:

- / *Type*: tuple /
None if test result UUID is not existing, else the tuple which contains project and version_sw: (project, variant) is returned.

Chapter 5

rest_api_db_access.py

5.1 Class: RestApiDBAccess

Imported by:

```
from TestResultDBAccess.DBAccess.rest_api_db_access import RestApiDBAccess
```

- 5.1.1 Method: get_certs_file
- 5.1.2 Method: encrypt_password
- 5.1.3 Method: connect
- 5.1.4 Method: disconnect
- 5.1.5 Method: arGetCategories
- 5.1.6 Method: bExistingResultID
- 5.1.7 Method: sGetLatestFileID
- 5.1.8 Method: arGetProjectVersionSWByID
- 5.1.9 Method: sCreateNewTestResult
- 5.1.10 Method: nCreateNewFile
- 5.1.11 Method: vCreateNewHeader
- 5.1.12 Method: nCreateNewSingleTestCase
- 5.1.13 Method: nCreateNewTestCase
- 5.1.14 Method: vCreateAbortReason
- 5.1.15 Method: vCreateCCRdata
- 5.1.16 Method: vCreateTags
- 5.1.17 Method: vCreateReanimation
- 5.1.18 Method: vSetCategory
- 5.1.19 Method: vUpdateFileEndTime
- 5.1.20 Method: vUpdateResultEndTime
- 5.1.21 Method: vFinishTestResult
- 5.1.22 Method: vUpdateEvtbl
- 5.1.23 Method: vUpdateEvtbls

Chapter 6

DBAccessFactory.py

6.1 Class: DBAccessFactory

Imported by:

```
from TestResultDBAccess.DBAccessFactory import DBAccessFactory
```

6.1.1 Method: create

Chapter 7

Appendix

About this package:

Table 7.1: Package setup

Setup parameter	Value
Name	TestResultDBAccess
Version	0.1.0
Date	25.04.2024
Description	Interfaces to access TestResultWebApp database
Package URL	python-testresultdbaccess
Author	Tran Duy Ngoan
Email	Ngoan.TranDuy@vn.bosch.com
Language	Programming Language :: Python :: 3
License	License :: OSI Approved :: Apache Software License
OS	Operating System :: OS Independent
Python required	>=3.0
Development status	Development Status :: 4 - Beta
Intended audience	Intended Audience :: Developers
Topic	Topic :: Software Development

Chapter 8

History

0.1.0	22.04.2024
<i>Initial version</i>	