TestResultDBAccess

v. 0.1.1

Tran Duy Ngoan

15.05.2024

Contents

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: DBAccessInterface

Imported by:

```
from TestResultDBAccess.DBAccess.db_accesss_interface import DBAccessInterface
```

Abstract base class defining the interface for database access.

This interface defines methods for connecting to and disconnecting from the database, as well as methods for retrieving, creating, updating, and calling stored procedures in the database.

3.1.1 Method: connect

Connects to the database using the provided connection parameters.

3.1.2 Method: disconnect

Disconnects from the database.

3.1.3 Method: commit

Commits a transaction (only applicable for DirectDBAccess with transaction support).

3.1.4 Method: arGetCategories

Retrieves categories from the database.

3.1.5 Method: bExistingResultID

Checks if the given result ID exists in the database.

3.1.6 Method: sGetLatestFileID

Retrieves the latest file ID from the database.

3.1.7 Method: sCreateNewTestResult

Creates a new test result record in the database.

3.1.8 Method: nCreateNewFile

Creates a new file record in the database.

3.1.9 Method: vCreateNewHeader

Creates a new file header record in the database.

3.1.10 Method: nCreateNewSingleTestCase

Creates a new single test case record in the database.

3.1.11 Method: nCreateNewTestCase

Creates new test case(s) record in the database.

3.1.12 Method: vCreateAbortReason

Creates a new abort reason record in the database.

3.1.13 Method: vCreateCCRdata

Creates new CCR data in the database.

3.1.14 Method: vCreateTags

Creates new tags in the database.

3.1.15 Method: vCreateReanimation

Updates an existing test result with reanimation data in the database.

3.1.16 Method: vSetCategory

Updates an existing test result with category information in the database.

3.1.17 Method: vUpdateFileEndTime

Updates an existing file record with end time in the database.

3.1.18 Method: vUpdateResultEndTime

Updates an existing test result record with end time in the database.

3.1.19 Method: vUpdateEvtbl

Calls a stored procedure to update event tables in the database.

3.1.20 Method: vUpdateEvtbls

Calls a stored procedure to update event tables (plural) in the database.

3.1.21 Method: vFinishTestResult

Calls a stored procedure to finish a test result in the database.

Chapter 4

direct_db_accesss.py

4.1 Class: DirectDBAccess

Imported by:

```
from TestResultDBAccess.DBAccess.direct_db_accesss 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. Deafult 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 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:
(*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: vUpdateStartEndTime

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 exsiting 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
```

RestApiDBAccess class provide methods to interact with TestResultWebApp's REST APIs.

This class implements the **DBAccessInterface** and extends it. It includes methods for connecting to the database, handling API requests, creating, updating, and calling stored procedures in the database via RESTful API calls.

5.1.1 Method: get_certs_file

Retrieves SSL certificates and saves them to a temporary file for request's verification.

Returns:

/ *Type*: str /
The path to the temporary file containing SSL certificates.

5.1.2 Method: encrypt_password

Encrypts a password using a public key.

Arguments:

- `password`
  / *Type*: str /
  The password to encrypt.
- `pubkey`
  / *Type*: str /
  The public key used for encryption.

Returns:

/ *Type*: str /
The encrypted password.

5.1.3 Method: connect

Connects to the database via REST API using the provided credentials.

Arguments:

- `host`
  / *Condition*: required / *Type*: str /
  URL which is hosted the TestResultWebApp's REST APIs.
- `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.

Returns:
(*no returns*)

5.1.4 Method: disconnect

Disconnect from TestResultWebApp's database.
Arguments:
(*no arguments*)
Returns:
(*no returns*)

5.1.5 Method: arGetCategories

Get existing categories.
Arguments:
(*no arguments*)
Returns:

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

5.1.6 Method: bExistingResultID

Verify the given test result UUID is existing or not.
Arguments:

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

Returns:

/ *Type*: bool /
True if given `result_id` is already existing.

5.1.7 Method: sGetLatestFileID

Get latest file ID of all result or given `result_id`.

Arguments:

- `result_id`
  / *Condition*: optional / *Type*: str /
  Test result ID. If used, the latest file id of given `result_id` is returned.

Returns:

/ *Type*: int /
Latest file ID.

5.1.8 Method: arGetProjectVersionSWByID

Get the project and version_sw information of given `result_id`

Arguments:

- `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.

5.1.9 Method: sCreateNewTestResult

Creates a new test result.

Arguments:

- `project`
  / *Condition*: required / *Type*: str /
  Project information.
- `variant`
  / *Condition*: required / *Type*: str /
  Variant information.
- `branch`
  / *Condition*: required / *Type*: str /
  Branch information.
- `result_id`
  / *Condition*: required / *Type*: str /
  UUID of test result.
- `result_interpretation`
  / *Condition*: required / *Type*: str /
  Result interpretation.

- `result_start_time`
  / *Condition*: required / *Type*: str /
  Test result start time as format `%Y-%m-%d %H:%M:%S`.
- `result_end_time`
  / *Condition*: required / *Type*: str /
  Test result end time as format `%Y-%m-%d %H:%M:%S`.
- `result_version_sw_target`
  / *Condition*: required / *Type*: str /
  Software version information.
- `result_version_sw_test`
  / *Condition*: required / *Type*: str /
  Test version information.
- `result_version_hw`
  / *Condition*: required / *Type*: str /
  Hardware version information.
- `result_build_url`
  / *Condition*: required / *Type*: str /
  Link to the execution of test result (Jenkins, Gitlab CI/CD, ...).
- `result_report_qualitygate`
  / *Condition*: required / *Type*: str /
  Qualitygate information for reporting.

Returns:

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

5.1.10 Method: nCreateNewFile

Create new result file.

Arguments:

- `file_name`
  / *Condition*: required / *Type*: str /
  File name information.
- `file_tester_account`
  / *Condition*: required / *Type*: str /
  Tester account information.
- `file_tester_machine`
  / *Condition*: required / *Type*: str /
  Test machine information.
- `file_time_start`
  / *Condition*: required / *Type*: str /
  Test file start time as format `%Y-%m-%d %H:%M:%S`.

- `file_time_end`
  / *Condition*: required / *Type*: str /
  Test file end time as format `%Y-%m-%d %H:%M:%S`.
- `result_id`
  / *Condition*: required / *Type*: str /
  UUID of test result to which this result file belongs.
- `file_origin`
  / *Condition*: required / *Type*: str /
  Origin (test framework) of test file."

Returns:

/ *Type*: int /
ID of new entry.

5.1.11 Method: vCreateNewHeader

Create a new result file header.

Arguments:

- `file_id`
  / *Condition*: required / *Type*: int /
  File ID information.
- `testtoolconfiguration_testtoolname`
  / *Condition*: required / *Type*: str /
  Test tool name.
- `testtoolconfiguration_testtoolversionstring`
  / *Condition*: required / *Type*: str /
  Test tool version.
- `testtoolconfiguration_projectname`
  / *Condition*: required / *Type*: str /
  Project name.
- `testtoolconfiguration_logfileencoding`
  / *Condition*: required / *Type*: str /
  Encoding of logfile.
- `testtoolconfiguration_pythonversion`
  / *Condition*: required / *Type*: str /
  Python version info.
- `testtoolconfiguration_testfile`
  / *Condition*: required / *Type*: str /
  Test file name.
- `testtoolconfiguration_logfilepath`
  / *Condition*: required / *Type*: str /
  Path to log file.
- `testtoolconfiguration_logfilemode`
  / *Condition*: required / *Type*: str /
  Mode of log file.

- `testtoolconfiguration_ctrlfilepath`
  / *Condition*: required / *Type*: str /
  Path to control file.
- `testtoolconfiguration_configfile`
  / *Condition*: required / *Type*: str /
  Path to configuration file.
- `testtoolconfiguration_confname`
  / *Condition*: required / *Type*: str /
  Configuration name.
- `testfileheader_author`
  / *Condition*: required / *Type*: str /
  File author.
- `testfileheader_project`
  / *Condition*: required / *Type*: str /
  Project information.
- `testfileheader_testfiledate`
  / *Condition*: required / *Type*: str /
  File creation date.
- `testfileheader_version_major`
  / *Condition*: required / *Type*: str /
  File major version.
- `testfileheader_version_minor`
  / *Condition*: required / *Type*: str /
  File minor version.
- `testfileheader_version_patch`
  / *Condition*: required / *Type*: str /
  File patch version.
- `testfileheader_keyword`
  / *Condition*: required / *Type*: str /
  File keyword.
- `testfileheader_shortdescription`
  / *Condition*: required / *Type*: str /
  File short description.
- `testexecution_useraccount`
  / *Condition*: required / *Type*: str /
  Tester account who run the execution.
- `testexecution_computername`
  / *Condition*: required / *Type*: str /
  Machine name which is executed on.
- `testrequirements_documentmanagement`
  / *Condition*: required / *Type*: str /
  Requirement management information.

- `testrequirements_testenvironment`
  / *Condition*: required / *Type*: str /
  Requirement environment information.
- `testbenchconfig_name`
  / *Condition*: required / *Type*: str /
  Testbench configuration name.
- `testbenchconfig_data`
  / *Condition*: required / *Type*: str /
  Testbench configuration data.
- `preprocessor_filter`
  / *Condition*: required / *Type*: str /
  Preprocessor filter information.
- `preprocessor_parameters`
  / *Condition*: required / *Type*: str /
  Preprocessor parameters definition.

Returns:
(*no returns*)

5.1.12 Method: nCreateNewSingleTestCase

Create single test case.
Arguments:

- `case_name`
  / *Condition*: required / *Type*: str /
  Test case name.
- `case_issue`
  / *Condition*: required / *Type*: str /
  Test case issue ID.
- `case_tcid`
  / *Condition*: required / *Type*: str /
  Test case ID (used for testmanagement tool).
- `case_fid`
  / *Condition*: required / *Type*: str /
  Test case requirement (function) ID.
- `case_testnumber`
  / *Condition*: required / *Type*: int /
  Order of test case in file.
- `case_repeatcount`
  / *Condition*: required / *Type*: int /
  Test case repeatition count.
- `case_component`
  / *Condition*: required / *Type*: str /
  Component which test case is belong to.

- `case_time_start`
  / *Condition*: required / *Type*: str /
  Test case start time as format `%Y-%m-%d %H:%M:%S`.
- `case_result_main`
  / *Condition*: required / *Type*: str /
  Test case main result.
- `case_result_state`
  / *Condition*: required / *Type*: str /
  Test case completion state.
- `case_result_return`
  / *Condition*: required / *Type*: int /
  Test case result code (as integer).
- `case_counter_resets`
  / *Condition*: required / *Type*: int /
  Counter of target reset within test case execution.
- `case_lastlog`
  / *Condition*: required / *Type*: str /
  Traceback information when test case is failed.
- `result_id`
  / *Condition*: required / *Type*: str /
  UUID of test result to which this test case belongs.
- `file_id`
  / *Condition*: required / *Type*: int /
  ID of result file to which this test case belongs.

Returns:

/ *Type*: int /
ID of new entry.

5.1.13 Method: nCreateNewTestCase

Alias for nCreateNewSingleTestCase, used in older import tools to import a bulk of test cases at once.

5.1.14 Method: vCreateAbortReason

Create abort reason entry.

Arguments:

- `result_id`
  / *Condition*: required / *Type*: str /
  UUID of test result.
- `abort_reason`
  / *Condition*: required / *Type*: str /
  Abort reason.
- `abort_message`
  / *Condition*: required / *Type*: str /
  Detail message of abort.

Returns:

(*no returns*)

5.1.15 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*)

5.1.16 Method: vCreateTags

Create tag entries.

Arguments:

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

Returns:

(*no returns*)

5.1.17 Method: vCreateReanimation

Create reanimation entry.

Arguments:

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

Returns:

(*no returns*)

5.1.18 Method: vSetCategory

Create category entry.

Arguments:

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

Returns:
(*no returns*)

5.1.19 Method: vUpdateFileEndTime

Update test file end time.
Arguments:

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

Returns:
(*no returns*)

5.1.20 Method: vUpdateResultEndTime

Update test result end time.
Arguments:

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

Returns:
(*no returns*)

5.1.21 Method: vFinishTestResult

Update state of given test result to "new report".
Arguments:

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

Returns:
(*no returns*)

5.1.22 Method: vUpdateEvtbl

Call `update_evtbl` stored procedure to update given `result_id`.

Arguments:

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

Returns:

(*no returns*)

5.1.23 Method: vUpdateEvtbls

Call `update_evtbls` stored procedure.

Arguments:

(*no arguments*)

Returns:

(*no returns*)

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	0.1.1
Date	15.05.2024
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
Initial version	
0.1.1	15.05.2024
Fixed issues of workflow to publish package to pypi	