BRIAR Client Documentation

Generated by Doxygen 1.9.1

Chapter 1

BRIAR Command Line Interface (CLI) and Client

Command Line Interface (CLI)

The command line interface provides a terminal based method of interacting with algorithms developed using the BRIAR API and unifies the commands given to the services into a set of universal commands shared across all projects developed using the API. It is the method of interfacing with any service built using the BRIAR framework as it shares the method calls assigned to the service and provides callable methods which invoke service functions. The client and the associated command line tools will be shared across all algorithms created with BRIAR and should not be modified, ensuring the Evaluation Harness and other sets of tests can be run across algorithms created by different sets of developers and generate comparable sets of results.

Like most command line toos, the command line interface can be scripted to act as part of a larger task. The client in briar_client.py (which the command line interface is an interface for) can also be imported into a python project

Usage

After running `setup.py`, the briar command line can be run by entering `python -m briar` anywhere. This will print a help statement showing the different functions made available by the command line tool.

Before running any of these functions, however, you will need to first start the provided example service, so it can reply to the commands. This is done by either directly calling the service python file `python service.py` or calling it as a module `python -m briar.service`. This will start the service, and it will run until you forcefully exit it.

You can get the status and version of the example service with `python -m briar status` and you should see the results printed by the client. Attempting to run any of the other functions with the example service in service.py will raise a "NotImplementedError".

More Details

The stubs and protobuf files which the client uses are detailed more thouroughly in the briar protobuf and stubs documentation

Chapter 2

Namespace Index

2.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 briar Namespace Reference

Namespaces

- __main__
- briar_cli

  Created on 2021 at Oak Ridge National Laboratory.
- briar_client

  Copyright 2021 Oak Ridge National Laboratory.
- briar_media

  Defines a media class which acts as a wrapper for image and video files.
- cli

  The modules contained in the CLI package each contain a function (detect, extract, enroll, etc.) which is a command within the broader CLI toolkit along with the assorted helper functions.
- evaluation
- grpc_json

  I couldn't find a good library to write gRPC/protobuf objects to disk so this is my own implementation of a generalized json converter for said objects.
- media
- media_converters

  Contained in this are functions for converting numpy arrays into various protobuf objects and back again since numpy arrays cannot be sent directly over gRPC.
- sigset
- timing

Classes

- class Rect

Functions

- def _wait_forever (server)
- def CLIServe (serviceClass, add_custom_options=None)
- def dyn_import (name)
- def get_process_number ()
- def get_thread_number ()
- def multiproc_serve (serviceClass, options=None, serve_port=None)
- def parse_ports (options)
- def serve (serviceClass, options=None, serve_port=None)

Variables

- string __version__ = '2.3.6'
- _ONE_DAY = datetime.timedelta(days=1)
- int DEFAULT_MAX_MESSAGE_SIZE = 64 * 1024 * 1024 *8
- string DEFAULT_PORT = "0.0.0.0:50051"
- string DEFAULT_SERVE_PORT = '[::]:50051'
- string PLATFORM = "UNKNOWN"

6.1.1 Function Documentation

6.1.1.1 _wait_forever()

```
def briar._wait_forever (
              server )  [private]
```

```
The _wait_forever function is a simple function that waits forever.
It's used to keep the server running until it receives a KeyboardInterrupt (Ctrl+C).


:param server: Stop the server
:return: Nothing
:doc-author: Joel Brogan
```

6.1.1.2 CLIServe()

```
def briar.CLIServe (
              serviceClass,
              add_custom_options = None )
```

```
:param serviceClass: The class that contains the implementation of the service to be served.
:param add_custom_options: A function that can be used to add custom options to the command line parser.
:return: None

This method sets up a command line interface (CLI) for serving a gRPC service. It takes in a serviceClass para
* to be served. It also takes an optional add_custom_options parameter, which is a function that can be used t

The method uses the `optparse` module to define and parse command line options. It creates an instance of `Opt
* command line options such as verbosity, maximum message size, port number(s), port range, number of services

If the add_custom_options parameter is not None, it calls the add_custom_options function to add custom optior

The method then parses the command line arguments using the `parse_args` method of the `OptionParser` instance
*` variables, respectively.

Depending on the number of ports specified or the number of services per port, the method either invokes the `
*. In case of an IndexError exception, an AssertionError is raised.

Note: This method assumes the existence of other functions and variables such as DEFAULT_MAX_MESSAGE_SIZE, DEF
* snippet.
```

6.1.1.3 dyn_import()

```
def briar.dyn_import (
              name )
```

```
The dyn_import function is a helper function that allows you to import modules
dynamically.  This means that you can pass in the name of a module as a string, and
the dyn_import function will return the actual module object.  For example:

:param name: Specify the name of the module to be imported
:return: A module object
:doc-author: Joel Brogan
```

6.1.1.4 get_process_number()

```
def briar.get_process_number ( )
```

```
The get_process_number function returns the process number of the current process.
The main process is denoted by a 0 while all other processes are 1-indexed.
:return: The process number of the current process
:doc-author: Joel Brogan
```

6.1.1.5 get_thread_number()

```
def briar.get_thread_number ( )
```

```
The get_thread_number function returns the thread number of the current thread.

:return: The thread number of the current thread
:doc-author: Joel Brogan
```

6.1.1.6 multiproc_serve()

```
def briar.multiproc_serve (
              serviceClass,
              options = None,
              serve_port = None )
```

```
The multiproc_serve function is a wrapper around the serve function that allows
multiple instances of the same service to be run on different ports. This is useful
for running multiple instances of a service in parallel, which can improve performance.
The multiproc_serve function takes three arguments:

:param serviceClass: Specify the class of service that is being served
:param options: Pass in the options for the service
:param serve_port: Specify the port that the server will listen on
:return: The return value of the serve function
:doc-author: Joel Brogan
```

6.1.1.7 parse_ports()

```
def briar.parse_ports (
              options )
```

```
The parse_ports function takes in a string of ports separated by commas, and returns a list of strings.
If the port range is greater than 1, then it will return a list with all the ports in that range.
For example: parse_ports(&quot;localhost:8080&quot;) -&gt; [&quot;localhost:8080&quot;]
         parse_ports(&quot;localhost:8000-8002&quot;) -&gt; [&quot;localhost:8000&quot;, &quot;localhost:80001

:param options: Parse the command line arguments
:return: A list of ports
:doc-author: Joel Brogan
```

6.1.1.8 serve()

```
def briar.serve (
              serviceClass,
              options = None,
              serve_port = None )
```

```
The serve function is the main entry point for a BRIARService. It initializes and runs the service until kille
Initialize and run the BRIARService. Runs until killed
:param serviceClass: Specify the service class to be used
:param options: Pass in the configuration options for the service
:param serve_port: Specify the port to serve on
:return: A server object
:doc-author: Joel Brogan
```

6.1.2 Variable Documentation

6.1.2.1 __version__

```
string __version__ = '2.3.6'  [private]
```

6.1.2.2 _ONE_DAY

```
_ONE_DAY = datetime.timedelta(days=1)  [private]
```

6.1.2.3 DEFAULT_MAX_MESSAGE_SIZE

```
int DEFAULT_MAX_MESSAGE_SIZE = 64 * 1024 * 1024 *8
```

6.1.2.4 DEFAULT_PORT

```
string DEFAULT_PORT = "0.0.0.0:50051"
```

6.1.2.5 DEFAULT_SERVE_PORT

```
string DEFAULT_SERVE_PORT = '[::]:50051'
```

6.1.2.6 PLATFORM

```
string PLATFORM = "UNKNOWN"
```

6.2 briar.__main__ Namespace Reference

6.3 briar.briar_cli Namespace Reference

Created on 2021 at Oak Ridge National Laboratory.

Functions

- def briar_command_line ()

  Entry point for the CLI - switches on the first command line argument (such as 'status', 'detect', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.
- def briar_database_command_line ()

  Entry point for the Database CLI - switches on the second command line argument (such as 'delete', 'merge', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.
- def briar_test_command_line ()

  Entry point for the Test CLI - switches on the second command line argument (such as 'delete', 'merge', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.
- def incomplete ()

Variables

- dictionary COMMANDS
- dictionary DATABASE_COMMANDS
- string DETECTION_FILE_EXT = ".detection"
- int FACE_COUNT = 0
- string MATCHES_FILE_EXT = '.matches'
- string TEMPLATE_FILE_EXT = '.template'

6.3.1 Detailed Description

Created on 2021 at Oak Ridge National Laboratory.

The Briar Command Line Interface (Briar CLI) provides a universal method to interface with different gRPC created with the compiled protobuf stubs. It provides a series of common functions to run detection and identification on faces, whole bodies, and walking gaits, as well as various database enrollment and search functions. Briar does not implement these detect, extract, enroll, etc functions itself, but rather acts as a means for connecting with servers (outlined with service.py)

Author

: Joel Brogan

6.3.2 Function Documentation

6.3.2.1 briar_command_line()

```
def briar.briar_cli.briar_command_line ( )
```

Entry point for the CLI - switches on the first command line argument (such as 'status', 'detect', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.

Each 'command' should be treated as a 'switch' which defines additional command line arguments.

Returns

:

6.3.2.2 briar_database_command_line()

```
def briar.briar_cli.briar_database_command_line ( )
```

Entry point for the Database CLI - switches on the second command line argument (such as 'delete', 'merge', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.

Each 'command' should be treated as a 'switch' which defines additional command line arguments.

Returns

:

6.3.2.3 briar_test_command_line()

```
def briar.briar_cli.briar_test_command_line ( )
```

Entry point for the Test CLI - switches on the second command line argument (such as 'delete', 'merge', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.

Each 'command' should be treated as a 'switch' which defines additional command line arguments.

Returns

:

6.3.2.4 incomplete()

```
def briar.briar_cli.incomplete ( )
```

```
The incomplete function is a placeholder for functions that have not yet been implemented.
It raises a NotImplementedError exception to indicate that the function has not yet been implemented.

:return: A notimplementederror
:doc-author: Joel Brogan
```

6.3.3 Variable Documentation

6.3.3.1 COMMANDS

```
dictionary COMMANDS
```

Initial value:

```
=  {
    ’detect’: [’Run detection on media files.’, detect],
    ’track’: [’Run tracking on media files (videos only).’, track],
    ’enhance’: [’Run tracking on media files (videos only).’, enhance],
    ’extract’: [’Run feature extraction to generate templates or embeddings.’, extract],
    ’enroll’: [’Scan media files and enroll templates into a database.’, enroll],
    ’verify’: [’Verify a given peice of media against a reference media.’, verify],
    ’finalize’: [’Finalize a database.’, database_finalize],
    ’search’: [’Search a database for an example identity.’, search],
    ’sigset-stats’: [’Convert a sigset to a csv file and compute statistics.’, sigset_stats],
    ’sigset-enroll’: [’Process a sigset and enroll in a database.’, sigset_enroll],
    ’status’: [’Connects to the server and displays version and status information.’, status],
    ’service-configuration’: [’Returns configuration settings from the service’,
    print_service_configuration],
    ’database’: [’Base call for database-related functions’, briar_database_command_line],
    ’vis’: [’Visualize saved results output from API Calls’, viz],
    ’test’: [’Base call for API test functions’, briar_test_command_line]
}
```

6.3.3.2 DATABASE_COMMANDS

```
dictionary DATABASE_COMMANDS
```

Initial value:

```
=  {
    ’create’: [’Create and initialize a new database.’, database_create],
    ’delete’: [’Delete a database from the service.’, database_delete],
    ’rename’: [’Rename a database from the service.’, database_rename],
    ’!insert’: [’Insert templates directly into a database.’, incomplete],
    ’load’: [’Load database onto the server.’, database_load],
    ’list’: [’List the names of the databases on this service.’, database_list],
    ’ls’: [’List the names of the databases on this service.’, database_list],
    ’info’: [’List information about a given database.’, database_info],
    ’list-entries’: [’List the entries contained within a database stored on this service.’,
    database_list_entries],
    ’finalize’: [’Finalize a database.’, database_finalize],
    ’checkpoint’: [’Checkpoint a database to save progress, without finalizing’, database_checkpoint],
    ’checkpoint-subject’: [’Checkpoint a database subject to save progress, without finalizing’,
    database_checkpoint_subject],
    ’compute-search’: [’Searches a probe database against a gallery database’, database_compute_search],
    ’compute-verify’: [’Performs batch verification ’, database_compute_verify],
    ’refresh’:  [’Performs a refresh of the list of databases to keep them coherent between
    services’,database_refresh],
    ’!remove-entries’: [’Remove entries from the database’, incomplete],
    ’merge’: [’Merge a list of existing databases together’, database_merge],
}
```

6.3.3.3 DETECTION_FILE_EXT

```
string DETECTION_FILE_EXT = ".detection"
```

6.3.3.4 FACE_COUNT

```
int FACE_COUNT = 0
```

6.3.3.5 MATCHES_FILE_EXT

```
string MATCHES_FILE_EXT = '.matches'
```

6.3.3.6 TEMPLATE_FILE_EXT

```
string TEMPLATE_FILE_EXT = '.template'
```

6.4 briar.briar_client Namespace Reference

Copyright 2021 Oak Ridge National Laboratory.

Classes

- class BriarClient

  Provide a client to a BRIAR service.

Functions

- def _initialize_worker (server_address, proc_number, thread_number, count_q)
- def _shutdown_worker ()

Variables

- _client_identifier_singleton = None
- _worker_channel_singleton = None
- _worker_port_singleton = None
- _worker_proccess_position_singleton = None
- _worker_stub_singleton = None
- _worker_thread_position_singleton = None

6.4.1 Detailed Description

Copyright 2021 Oak Ridge National Laboratory.

The BRIAR API is divided into two primary parts, the client and the service. This, the client, is the part which interfaces with grpc servers based off briar.service.BRIARService using the BRIARServiceStub. The service stub contains the same methods contained in the service which, when invoked with the appropriate request, sends said request to the service which the client is connected to, and accepts the reply containing processed detections, extracts, templates, etc...

The BRIAR client is designed to serve as a unified interface with gRPC services which are designed after BRIARService and implement various performer algorithms for face and body detection/extraction. From a performer standpoint, The BRIAR client can be used as either part of the command line tools, or invoked alone as a module

6.4.2 Function Documentation

6.4.2.1 _initialize_worker()

```
def briar.briar_client._initialize_worker (
              server_address,
              proc_number,
              thread_number,
              count_q )  [private]
```

6.4.2.2 _shutdown_worker()

```
def briar.briar_client._shutdown_worker ( )  [private]
```

6.4.3 Variable Documentation

6.4.3.1 _client_identifier_singleton

```
_client_identifier_singleton = None  [private]
```

6.4.3.2 _worker_channel_singleton

```
_worker_channel_singleton = None  [private]
```

6.4.3.3 _worker_port_singleton

```
_worker_port_singleton = None  [private]
```

6.4.3.4 _worker_proccess_position_singleton

```
_worker_proccess_position_singleton = None  [private]
```

6.4.3.5 _worker_stub_singleton

```
_worker_stub_singleton = None  [private]
```

6.4.3.6 _worker_thread_position_singleton

```
_worker_thread_position_singleton = None  [private]
```

6.5 briar.briar_media Namespace Reference

Defines a media class which acts as a wrapper for image and video files.

Classes

- class BriarMedia
- class MediaStream

Functions

- def briar_media_from_pb2 (pb2_object)
- def briar_media_to_pb2 (media)
- def load_media_from_folder (folder_path, recursive=False)
- def load_media_from_image (image_path)
- def load_media_from_numpy (numpy_array)

6.5.1 Detailed Description

Defines a media class which acts as a wrapper for image and video files.

6.5.2 Function Documentation

6.5.2.1 briar_media_from_pb2()

```
def briar.briar_media.briar_media_from_pb2 (
              pb2_object )
```

6.5.2.2 briar_media_to_pb2()

```
def briar.briar_media.briar_media_to_pb2 (
              media )
```

6.5.2.3 load_media_from_folder()

```
def briar.briar_media.load_media_from_folder (
              folder_path,
              recursive = False )
```

6.5.2.4 load_media_from_image()

```
def briar.briar_media.load_media_from_image (
              image_path )
```

6.5.2.5 load_media_from_numpy()

```
def briar.briar_media.load_media_from_numpy (
              numpy_array )
```

6.6 briar.cli Namespace Reference

The modules contained in the CLI package each contain a function (detect, extract, enroll, etc.) which is a command within the broader CLI toolkit along with the assorted helper functions.

Namespaces

- connection
- database
- detect
- enhance
- enroll
- extract
- finalize
- media
- search
- sigset
- status
- test
- track
- verify
- viz

6.6.1 Detailed Description

The modules contained in the CLI package each contain a function (detect, extract, enroll, etc.) which is a command within the broader CLI toolkit along with the assorted helper functions.

The briar_cli file has each of these important functions mapped in a dictionary which accesses them based off of user commands. The module functions then add additional command line options which can be parsed into options or displayed as part of a help message. From there, the module functions will connect to the specified service and send it messages and receive replies based off of the arguments passed in through the command line.

6.7 briar.cli.connection Namespace Reference

Functions

- def addConnectionOptions (parser)

  Accumulatively add options for connecting to the Briar API service.

Variables

- int DEFAULT_MAX_ASYNC = 8
- int DEFAULT_MAX_MESSAGE_SIZE = 64 ∗ 1024 ∗ 1024 ∗ 8

6.7.1 Function Documentation

6.7.1.1 addConnectionOptions()

```
def briar.cli.connection.addConnectionOptions (
              parser )
```

Accumulatively add options for connecting to the Briar API service.

Modifiers the parser in plase

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding connection options

6.7.2 Variable Documentation

6.7.2.1 DEFAULT_MAX_ASYNC

```
int DEFAULT_MAX_ASYNC = 8
```

6.7.2.2 DEFAULT_MAX_MESSAGE_SIZE

```
int DEFAULT_MAX_MESSAGE_SIZE = 64 ∗ 1024 ∗ 1024 ∗ 8
```

6.8 briar.cli.database Namespace Reference

Namespaces

- checkpoint
- checkpoint_subject
- common
- compute_scores
- compute_search
- create
- delete
- finalize
- info
- list
- list_entries
- load
- merge
- move_entry
- refresh
- rename
- retrieve

6.9 briar.cli.database.checkpoint Namespace Reference

Functions

- def database_checkpoint (options=None, args=None, input_command=None, ret=False)

6.9.1 Function Documentation

6.9.1.1 database_checkpoint()

```
def briar.cli.database.checkpoint.database_checkpoint (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 Checkpoints a database without finalizing it
```

6.10 briar.cli.database.checkpoint_subject Namespace Reference

Functions

- def database_checkpoint_subject (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseCheckpointSubjectOptions (inputCommand=None)

  Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

6.10.1 Function Documentation

6.10.1.1 database_checkpoint_subject()

```
def briar.cli.database.checkpoint_subject.database_checkpoint_subject (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

6.10.1.2 parseDatabaseCheckpointSubjectOptions()

```
def briar.cli.database.checkpoint_subject.parseDatabaseCheckpointSubjectOptions (
              inputCommand = None )
```

Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.11 briar.cli.database.common Namespace Reference

Functions

- def db_no_exist (name)

6.11.1 Function Documentation

6.11.1.1 db_no_exist()

```
def briar.cli.database.common.db_no_exist (
              name )
```

```
The db_no_exist function is called when the user attempts to access a database that does not exist.
It prints an error message and returns None.

:param name: Print the name of the database that doesn't exist
:return: A string
:doc-author: Joel Brogan
```

6.12 briar.cli.database.compute_scores Namespace Reference

Functions

- def addDatabaseComputeScore_options2proto (options)
- def addDatabaseComputeScoreOptions (parser)

  Add options for search of a database using a database.
- def database_compute_verify (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a batch verification betweeen the specified databases using specified probe database.
- def parseDatabaseComputeScoreOptions (inputCommand=None)

  Generate options for matching databases against other databases, and parse command line arguments into the API call.

6.12.1 Function Documentation

6.12.1.1 addDatabaseComputeScore_options2proto()

```
def briar.cli.database.compute_scores.addDatabaseComputeScore_options2proto (
              options )
```

```
Parse command line options and populate a proto object for grpc
```

6.12.1.2 addDatabaseComputeScoreOptions()

```
def briar.cli.database.compute_scores.addDatabaseComputeScoreOptions (
              parser )
```

Add options for search of a database using a database.

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.12.1.3 database_compute_verify()

```
def briar.cli.database.compute_scores.database_compute_verify (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs a batch verification betweeen the specified databases using specified probe database.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.12.1.4 parseDatabaseComputeScoreOptions()

```
def briar.cli.database.compute_scores.parseDatabaseComputeScoreOptions (
              inputCommand = None )
```

Generate options for matching databases against other databases, and parse command line arguments into the API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.13 briar.cli.database.compute_search Namespace Reference

Functions

- def database_compute_search (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a search within the specified database using specified probe database.

6.13.1 Function Documentation

6.13.1.1 database_compute_search()

```
def briar.cli.database.compute_search.database_compute_search (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs a search within the specified database using specified probe database.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.14 briar.cli.database.create Namespace Reference

Functions

- def database_create (options=None, args=None, input_command=None, ret=False)

6.14.1 Function Documentation

6.14.1.1 database_create()

```
def briar.cli.database.create.database_create (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 Checkpoints a database without finalizing it
```

6.15 briar.cli.database.delete Namespace Reference

Functions

- def database_delete (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseDeleteOptions (inputCommand=None)

  Generate options for Deleting a pre-existing database and parse command line arguments into API call.

6.15.1 Function Documentation

6.15.1.1 database_delete()

```
def briar.cli.database.delete.database_delete (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 delete a database
```

6.15.1.2 parseDatabaseDeleteOptions()

```
def briar.cli.database.delete.parseDatabaseDeleteOptions (
              inputCommand = None )
```

Generate options for Deleting a pre-existing database and parse command line arguments into API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.16 briar.cli.database.finalize Namespace Reference

Functions

- def database_finalize (options=None, args=None, input_command=None, ret=False)

  Parses the command line options and saves the database to disk.
- def finalizeParseOptions (inputCommand=None)

  Generate options for running 'finalize' (saving the loaded databases) and parse command line arguments into them.

6.16.1 Function Documentation

6.16.1.1 database_finalize()

```
def briar.cli.database.finalize.database_finalize (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Parses the command line options and saves the database to disk.

Returns

: None - results are written to disk to a location specified by options

6.16.1.2 finalizeParseOptions()

```
def briar.cli.database.finalize.finalizeParseOptions (
              inputCommand = None )
```

Generate options for running 'finalize' (saving the loaded databases) and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.17 briar.cli.database.info Namespace Reference

Functions

- def database_info (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseInfoOptions (inputCommand=None)

    Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

6.17.1 Function Documentation

6.17.1.1 database_info()

```
def briar.cli.database.info.database_info (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 list the information pertaining to a database
```

6.17.1.2 parseDatabaseInfoOptions()

```
def briar.cli.database.info.parseDatabaseInfoOptions (
              inputCommand = None )
```

Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.18 briar.cli.database.list Namespace Reference

Functions

- def database_list (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseListOptions (inputCommand=None)

    Generate options for listing all pre-existing databases and parse command line arguments into them.

6.18.1 Function Documentation

6.18.1.1 database_list()

```
def briar.cli.database.list.database_list (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
list the names of the databases
```

6.18.1.2 parseDatabaseListOptions()

```
def briar.cli.database.list.parseDatabaseListOptions (
              inputCommand = None )
```

Generate options for listing all pre-existing databases and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.19 briar.cli.database.list_entries Namespace Reference

Functions

- def database_list_entries (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseListEntriesOptions (inputCommand=None)

  Generate options for Listing entries within a pre-existing database and parse command line arguments into an API call.

6.19.1 Function Documentation

6.19.1.1 database_list_entries()

```
def briar.cli.database.list_entries.database_list_entries (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
list the entries in a database
```

6.19.1.2 parseDatabaseListEntriesOptions()

```
def briar.cli.database.list_entries.parseDatabaseListEntriesOptions (
              inputCommand = None )
```

Generate options for Listing entries within a pre-existing database and parse command line arguments into an API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.20 briar.cli.database.load Namespace Reference

Functions

- def database_load ()

6.20.1 Function Documentation

6.20.1.1 database_load()

```
def briar.cli.database.load.database_load ( )
```

```
 Loads a database from storage (in case the database needs a procedure for loading from disk into memory)
```

6.21 briar.cli.database.merge Namespace Reference

Functions

- def database_merge (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseMergeOptions (inputCommand=None)

  Generate options for merging databases and parse command line arguments into the API call.

6.21.1 Function Documentation

6.21.1.1 database_merge()

```
def briar.cli.database.merge.database_merge (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 Merge a set of databases
```

6.21.1.2 parseDatabaseMergeOptions()

```
def briar.cli.database.merge.parseDatabaseMergeOptions (
              inputCommand = None )
```

Generate options for merging databases and parse command line arguments into the API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.22 briar.cli.database.move_entry Namespace Reference

Functions

- def database_move_entry (options=None, args=None, input_command=None, ret=False, client=None)
- def parseDatabaseMoveEntryOptions (inputCommand=None)

  Generate options for moving database entries and parse command line arguments into the API call.

6.22.1 Function Documentation

6.22.1.1 database_move_entry()

```
def briar.cli.database.move_entry.database_move_entry (
              options = None,
              args = None,
              input_command = None,
              ret = False,
              client = None )
```

```
 Merge a set of databases
```

6.22.1.2 parseDatabaseMoveEntryOptions()

```
def briar.cli.database.move_entry.parseDatabaseMoveEntryOptions (
              inputCommand = None )
```

Generate options for moving database entries and parse command line arguments into the API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.23 briar.cli.database.refresh Namespace Reference

Functions

- def database_checkpoint (options=None, args=None, input_command=None, ret=False)
- def database_refresh (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseRefreshOptions (inputCommand=None)

  Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

6.23.1 Function Documentation

6.23.1.1 database_checkpoint()

```
def briar.cli.database.refresh.database_checkpoint (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 Checkpoints a database without finalizing it
```

6.23.1.2 database_refresh()

```
def briar.cli.database.refresh.database_refresh (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 Refresh the list of databases
```

6.23.1.3 parseDatabaseRefreshOptions()

```
def briar.cli.database.refresh.parseDatabaseRefreshOptions (
              inputCommand = None )
```

Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.24 briar.cli.database.rename Namespace Reference

Functions

- def database_load ()
- def database_rename (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseRenameOptions (inputCommand=None)

  Generate options for Renaming a pre-existing database to a new name and parse command line arguments into API call.

6.24.1 Function Documentation

6.24.1.1 database_load()

```
def briar.cli.database.rename.database_load ( )
```

```
 Loads a database from storage
```

6.24.1.2 database_rename()

```
def briar.cli.database.rename.database_rename (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 rename a database
```

6.24.1.3 parseDatabaseRenameOptions()

```
def briar.cli.database.rename.parseDatabaseRenameOptions (
              inputCommand = None )
```

Generate options for Renaming a pre-existing database to a new name and parse command line arguments into API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.25 briar.cli.database.retrieve Namespace Reference

Functions

- def database_retrieve (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseRetrieveOptions (inputCommand=None)

  Generate options for retrieving a pre-existing database and parse command line arguments API call.

6.25.1 Function Documentation

6.25.1.1 database_retrieve()

```
def briar.cli.database.retrieve.database_retrieve (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

```
 list the entries in a database
```

6.25.1.2 parseDatabaseRetrieveOptions()

```
def briar.cli.database.retrieve.parseDatabaseRetrieveOptions (
              inputCommand = None )
```

Generate options for retrieving a pre-existing database and parse command line arguments API call.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.26 briar.cli.detect Namespace Reference

Functions

- def addDetectorOptions (parser)

  Add options for running detections to the parser.
- def addTrackingOptions (parser)

  Add options for running detections to the parser.
- def detect (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a detection on the specified files.
- def detect_options2proto (options)
- def detectParseOptions (inputCommand=None)

  Generate options for running detections and parse command line arguments into them.
- def detectRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)
- def get_detection_path (media_file, options, i, modality=None, media_id=None)
- def isFinalReply (srvc_pb2.DetectReply reply)
- def save_detections (media_file, reply, options, i, modality=None, media_id=None)
- def tracking_options2proto (options)

Variables

- string DETECTION_FILE_EXT = ".detection"

6.26.1 Function Documentation

6.26.1.1 addDetectorOptions()

```
def briar.cli.detect.addDetectorOptions (
              parser )
```

Add options for running detections to the parser.

Modifiers the parser in plase

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.26.1.2 addTrackingOptions()

```
def briar.cli.detect.addTrackingOptions (
              parser )
```

Add options for running detections to the parser.

Modifiers the parser in place

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.26.1.3 detect()

```
def briar.cli.detect.detect (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs a detection on the specified files.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.26.1.4 detect_options2proto()

```
def briar.cli.detect.detect_options2proto (
              options )
```

```
Parse command line options and populate a proto object for grpc
```

6.26.1.5 detectParseOptions()

```
def briar.cli.detect.detectParseOptions (
              inputCommand = None )
```

Generate options for running detections and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.26.1.6 detectRequestConstructor()

```
def briar.cli.detect.detectRequestConstructor (
             briar_pb2.BriarMedia media,
             briar_pb2.BriarDurations durations,
              options_dict = {},
              det_list_list = None,
             str  database_name = None )
```

6.26.1.7 get_detection_path()

```
def briar.cli.detect.get_detection_path (
              media_file,
              options,
              i,
              modality = None,
              media_id = None )
```

6.26.1.8 isFinalReply()

```
def briar.cli.detect.isFinalReply (
             srvc_pb2.DetectReply reply )
```

6.26.1.9 save_detections()

```
def briar.cli.detect.save_detections (
              media_file,
              reply,
              options,
              i,
              modality = None,
              media_id = None )
```

6.26.1.10 tracking_options2proto()

```
def briar.cli.detect.tracking_options2proto (
              options )
```

```
Parse command line options and populate a proto object for grpc
```

6.26.2 Variable Documentation

6.26.2.1 DETECTION_FILE_EXT

```
string DETECTION_FILE_EXT = ".detection"
```

6.27 briar.cli.enhance Namespace Reference

Functions

- def addEnhanceOptions (parser)

  Add options for running detections to the parser.
- def enhance (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a detection on the specified files.
- def enhance_options2proto (options)
- def enhanceParseOptions (inputCommand=None)

  Generate options for running enhancement and parse command line arguments into them.
- def enhanceRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options←↩ _dict={}, det_list_list=None, str database_name=None)
- def save_Enhancement (media_file, reply, options, i, modality=None)

Variables

- string ENHANCE_FILE_EXT = '.enh'

6.27.1 Function Documentation

6.27.1.1 addEnhanceOptions()

```
def briar.cli.enhance.addEnhanceOptions (
              parser )
```

Add options for running detections to the parser.

Modifiers the parser in plase

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.27.1.2 enhance()

```
def briar.cli.enhance.enhance (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs a detection on the specified files.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.27.1.3 enhance_options2proto()

```
def briar.cli.enhance.enhance_options2proto (
              options )
```

```
Parse command line options and populate a proto object for grpc
```

6.27.1.4 enhanceParseOptions()

```
def briar.cli.enhance.enhanceParseOptions (
              inputCommand = None )
```

Generate options for running enhancement and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.27.1.5 enhanceRequestConstructor()

```
def briar.cli.enhance.enhanceRequestConstructor (
             briar_pb2.BriarMedia media,
             briar_pb2.BriarDurations durations,
              options_dict = {},
              det_list_list = None,
             str  database_name = None )
```

6.27.1.6 save_Enhancement()

```
def briar.cli.enhance.save_Enhancement (
              media_file,
              reply,
              options,
              i,
              modality = None )
```

6.27.2 Variable Documentation

6.27.2.1 ENHANCE_FILE_EXT

```
string ENHANCE_FILE_EXT = '.enh'
```

6.28 briar.cli.enroll Namespace Reference

Functions

- def addEnrollOptions (parser)

  Add options for enrollment into a database.
- def enroll (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs an enroll on the specified files.
- def enroll_options2proto (options)
- def enrollParseOptions (inputCommand=None)

  Generate options for running enrollments and parse command line arguments into them.
- def enrollRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)

6.28.1 Function Documentation

6.28.1.1 addEnrollOptions()

```
def briar.cli.enroll.addEnrollOptions (
              parser )
```

Add options for enrollment into a database.

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.28.1.2 enroll()

```
def briar.cli.enroll.enroll (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs an enroll on the specified files.

Can enroll media files (runs auto detection), detections (auto extracts ROI defined by detects) or templates (skips detect/extract)

Writes results to disk to a location specified by the cmd arguments.

Returns

: No return - Function writes results to disk

6.28.1.3 enroll_options2proto()

```
def briar.cli.enroll.enroll_options2proto (
              options )
```

```
The enroll_options2proto function takes an EnrollOptions object and converts it to a protobuf message.

:param options: Pass in the options for enrolling a template
:return: A proto object that can be sent to the server
:doc-author: Joel Brogan
```

6.28.1.4 enrollParseOptions()

```
def briar.cli.enroll.enrollParseOptions (
              inputCommand = None )
```

Generate options for running enrollments and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.28.1.5 enrollRequestConstructor()

```
def briar.cli.enroll.enrollRequestConstructor (
            briar_pb2.BriarMedia media,
            briar_pb2.BriarDurations durations,
             options_dict = {},
             det_list_list = None,
            str  database_name = None )
```

```
The enrollRequestConstructor function is used to construct an EnrollRequest object.
The EnrollRequest object contains the following fields:
database - The name of the database that will be used for enrollment. If no name is provided, then a default
           database will be created and used for enrollment.

:param media: briar_pb2.BriarMedia: Specify the media type
:param durations: briar_pb2.BriarDurations: Record the time it takes to perform each step of the
:param options_dict: Pass the options for each of the three stages of processing
:param det_list_list: Pass the detections from the previous frame to the next frame
:param database_name: str: Specify the name of the database to enroll into
:return: A enrollrequest object
:doc-author: Joel Brogan
```

6.29 briar.cli.extract Namespace Reference

Functions

- def addExtractOptions (parser)

  Add options for extractions to the parser.
- def extract (options=None, args=None, inputCommand=None, ret=False)

  Using the options specified in the command line, runs an extract on the specified files.
- def extract_options2proto (options)
- def extractParseOptions (inputCommand=None)

  Generate options for running extracts and parse command line arguments into them.
- def extractRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_dict={}, det_list_list=None, str database_name=None)
- def save_extractions (media_file, templates, options, i, modality=None, media_id=None)

Variables

- string TEMPLATE_FILE_EXT = '.template'

6.29.1 Function Documentation

6.29.1.1 addExtractOptions()

```
def briar.cli.extract.addExtractOptions (
             parser )
```

Add options for extractions to the parser.

@type parser: optparse.OptionParser

Parameters

parser	A parser to modify in place by adding options

6.29.1.2 extract()

```
def briar.cli.extract.extract (
              options = None,
              args = None,
              inputCommand = None,
              ret = False )
```

Using the options specified in the command line, runs an extract on the specified files.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.29.1.3 extract_options2proto()

```
def briar.cli.extract.extract_options2proto (
              options )
```

6.29.1.4 extractParseOptions()

```
def briar.cli.extract.extractParseOptions (
              inputCommand = None )
```

Generate options for running extracts and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.29.1.5 extractRequestConstructor()

```
def briar.cli.extract.extractRequestConstructor (
            briar_pb2.BriarMedia media,
            briar_pb2.BriarDurations durations,
             options_dict = {},
             det_list_list = None,
            str  database_name = None )
```

6.29.1.6 save_extractions()

```
def briar.cli.extract.save_extractions (
             media_file,
             templates,
             options,
             i,
             modality = None,
             media_id = None )
```

6.29.2 Variable Documentation

6.29.2.1 TEMPLATE_FILE_EXT

```
string TEMPLATE_FILE_EXT = '.template'
```

6.30 briar.cli.finalize Namespace Reference

Functions

- def database_finalize (options=None, args=None, input_command=None, ret=False)

  Parses the command line options and saves the database to disk.
- def finalizeParseOptions (inputCommand=None)

  Generate options for running 'finalize' (saving the loaded databases) and parse command line arguments into them.

6.30.1 Function Documentation

6.30.1.1 database_finalize()

```
def briar.cli.finalize.database_finalize (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Parses the command line options and saves the database to disk.

Returns

: None - results are written to disk to a location specified by options

6.30.1.2 finalizeParseOptions()

```
def briar.cli.finalize.finalizeParseOptions (
              inputCommand = None )
```

Generate options for running 'finalize' (saving the loaded databases) and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.31 briar.cli.media Namespace Reference

Functions

- def addMediaOptions (parser)
  Add options for running detections to the parser.
- def collect_files (args, options, extension=None)
  Take the paths specified by 'args' and find all the media files that they define: folders will be searched for all media files contained inside.
- def hasExtension (f, extension)

Variables

- int DEFAULT_MAX_SIZE = 1920

6.31.1 Function Documentation

6.31.1.1 addMediaOptions()

```
def briar.cli.media.addMediaOptions (
              parser )
```

Add options for running detections to the parser.

Modifiers the parser in plase

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.31.1.2 collect_files()

```
def briar.cli.media.collect_files (
              args,
              options,
              extension = None )
```

Take the paths specified by 'args' and find all the media files that they define: folders will be searched for all media files contained inside.

Parameters

args	list(str): List of paths to add as/search for media files
options	optparse.Values: Command line options which dictate collect behavior
extension	str: A specific extension which defines the csv files associated with media.

Returns

> : Return value depends on 'extension' If 'extension' is None, Tuple will be two elements (list of str, list of str) representing lists of image paths and video paths respectively

If 'extension' is not None, returns a single list of csv files with extensions matching 'extension'

6.31.1.3 hasExtension()

```
def briar.cli.media.hasExtension (
              f,
              extension )
```

6.31.2 Variable Documentation

6.31.2.1 DEFAULT_MAX_SIZE

```
int DEFAULT_MAX_SIZE = 1920
```

6.32 briar.cli.search Namespace Reference

Functions

- def addSearchOptions (parser)

  Add options for search of a database.
- def search (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a search within the specified database using specified probe template(s).
- def search_options2proto (options)
- def searchParseOptions (inputCommand=None)

  Generate options for running searches and parse command line arguments into them.
- def searchRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)

Variables

- string MATCHES_FILE_EXT = '.matches'

6.32.1 Function Documentation

6.32.1.1 addSearchOptions()

```
def briar.cli.search.addSearchOptions (
              parser )
```

Add options for search of a database.

Parameters

parser	optparse.OptionParser: A parser to modify in place by adding options

6.32.1.2 search()

```
def briar.cli.search.search (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs a search within the specified database using specified probe template(s).

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.32.1.3 search_options2proto()

```
def briar.cli.search.search_options2proto (
              options )
```

```
Parse command line options and populate a proto object for grpc
```

6.32.1.4 searchParseOptions()

```
def briar.cli.search.searchParseOptions (
              inputCommand = None )
```

Generate options for running searches and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.32.1.5 searchRequestConstructor()

```
def briar.cli.search.searchRequestConstructor (
             briar_pb2.BriarMedia media,
             briar_pb2.BriarDurations durations,
              options_dict = {},
              det_list_list = None,
             str  database_name = None )
```

6.32.2 Variable Documentation

6.32.2.1 MATCHES_FILE_EXT

```
string MATCHES_FILE_EXT = '.matches'
```

6.33 briar.cli.sigset Namespace Reference

Functions

- def checkpoint_subject_threaded (obj)
- def df_batch_consumer (batch_queue, identifier, port, server_configuration, id_queue, progress_queue)
- def df_batch_generator (list_of_dfs, start, dataset_dir, detect_options, extract_options, enroll_options, options)
- def df_batch_producer (batch_queue, identifier, progress_position, list_of_dataframes, start, dataset_dir, detect_options, extract_options, enroll_options, options)
- def df_row_generator (rowiter, start, dataset_dir, detect_options, extract_options, enroll_options, options)
- def enroll_call_threaded (input)
- def inner_pool_mapper (batch_obj, local_pool, progress_queue, port)
- def parseSigsetEnrollOptions (inputCommand=None)
- def parseSigsetStatsOptions (inputCommand=None)
- def progress_consumer (progress_queue, progress_position, total_len, options)
- def sigset_enroll (input_command=None)
- def sigset_stats (options=None, args=None)

Variables

- my_pool = None
- proc_number = None
- service_address_number = None

6.33.1 Function Documentation

6.33.1.1 checkpoint_subject_threaded()

```
def briar.cli.sigset.checkpoint_subject_threaded (
              obj )
```

6.33.1.2 df_batch_consumer()

```
def briar.cli.sigset.df_batch_consumer (
              batch_queue,
              identifier,
              port,
              server_configuration,
              id_queue,
              progress_queue )
```

```
The df_batch_consumer function is a function that takes in a batch of dataframe objects, and processes them us
Briar API. The function uses the multiprocessing library to create multiple worker processes, each with their
connection to the Briar server. Each process then creates multiple threads which are used for processing image

:param batch_queue: Pass in the queue of batches to be processed
:param identifier: Identify the process
:param port: Determine which port to connect to
:param server_configuration: Specify the number of processes and threads per process
:param id_queue: Pass the connection id to the worker process
:param progress_queue: Send progress information to the main process
:return: A list of results
:doc-author: Joel Brogan
```

6.33.1.3 df_batch_generator()

```
def briar.cli.sigset.df_batch_generator (
              list_of_dfs,
              start,
              dataset_dir,
              detect_options,
              extract_options,
              enroll_options,
              options )
```

```
The df_batch_generator function is a generator that yields batches of dataframes to be processed by the
multiprocessing pool. The function takes in a list of dataframes, and returns each one as an element in the
generator. This allows for parallel processing using multiple cores on your machine.

:param list_of_dfs: Pass in the list of dataframes that are to be processed
:param start: Keep track of the current index in the list_of_dfs
:param dataset_dir: Specify the directory where the images are stored
:param detect_options: Specify the type of detection algorithm to use
:param extract_options: Specify the extraction algorithm to be used
:param enroll_options: Specify the enrollment options for each batch
:param options: Pass in the number of processes to use
:return: A generator that yields a tuple of the following form:
:doc-author: Joel Brogan
```

6.33.1.4 df_batch_producer()

```
def briar.cli.sigset.df_batch_producer (
              batch_queue,
              identifier,
              progress_position,
              list_of_dataframes,
              start,
              dataset_dir,
              detect_options,
              extract_options,
              enroll_options,
              options )
```

```
The df_batch_producer function is a generator that takes in a list of dataframes and yields batches of images.
The function will yield batches until the end of the list_of_dataframes is reached.

:param batch_queue: Pass the batches to the main process
:param identifier: Identify the thread
:param progress_position: Keep track of the progress of the batch_producer function
:param list_of_dataframes: Store the list of dataframes that are to be processed
:param start: Determine the starting index of the dataframe
:param dataset_dir: Specify the directory where the dataset is located
:param detect_options: Specify the detection options
:param extract_options: Specify the extraction options
:param enroll_options: Specify the enrollment options
:param options: Pass the options for each of the functions
:return: A batch of dataframes
:doc-author: Joel Brogan
```

6.33.1.5 df_row_generator()

```
def briar.cli.sigset.df_row_generator (
              rowiter,
              start,
              dataset_dir,
              detect_options,
              extract_options,
              enroll_options,
              options )
```

```
The df_row_generator function is a generator that yields the row of the dataframe,
the index of the row, and all other arguments passed to it. This allows us to use
multiprocessing.Pool's map function with multiple arguments.

:param rowiter: Iterate over the rows of a dataframe
:param start: Keep track of the row number
:param dataset_dir: Specify the directory where the dataset is located
:param detect_options: Set the detection options
:param extract_options: Pass the extract_options to the worker function
:param enroll_options: Pass the enroll_options to the function
:param options: Pass the options to the enroll_options function
:return: A generator that yields a list of arguments
:doc-author: Joel Brogan
```

6.33.1.6 enroll_call_threaded()

```
def briar.cli.sigset.enroll_call_threaded (
              input )
```

```
Enroll the given input data into the Briar database.

Parameters:
    input (tuple): A tuple containing the following elements:
        - row (dict): A dictionary containing the information about the input data.
        - i (int): Index of the input data.
        - dataset_dir (str): Directory path of the dataset.
        - detect_options (briar_pb2.FrameDetectOptions): Detection options for the enrollment process.
        - extract_options (briar_pb2.TemplateExtractOptions): Template extraction options for the enrollment p
        - enroll_options (briar_pb2.TemplateEnrollOptions): Template enrollment options for the enrollment pro
        - options (argparse.Namespace): Command-line options for the enrollment process.

Returns:
    None
```

6.33.1.7 inner_pool_mapper()

```
def briar.cli.sigset.inner_pool_mapper (
              batch_obj,
              local_pool,
              progress_queue,
              port )
```

```
The inner_pool_mapper function is a function that takes in a batch of data, and then maps the enroll_call_thre
The inner pool mapper is used to map the enroll call threaded function over batches of data. The inner pool ma

:param batch_obj: Pass the following parameters to the inner_pool_mapper function:
:param local_pool: Specify the number of threads to use for processing
:param progress_queue: Communicate progress back to the main process
:return: A list of results
:doc-author: Joel Brogan
```

6.33.1.8 parseSigsetEnrollOptions()

```
def briar.cli.sigset.parseSigsetEnrollOptions (
              inputCommand = None )
```

```
The parseSigsetEnrollOptions function parses command line arguments for the sigset-enroll program.

:param inputCommand: Pass in the command line arguments
:return: The options and args
:doc-author: Joel Brogan, David Bolme, Trelent
```

6.33.1.9 parseSigsetStatsOptions()

```
def briar.cli.sigset.parseSigsetStatsOptions (
              inputCommand = None )
```

```
Parse command line arguments.
```

6.33.1.10 progress_consumer()

```
def briar.cli.sigset.progress_consumer (
              progress_queue,
              progress_position,
              total_len,
              options )
```

```
The progress_consumer function is a consumer for the progress_queue.
It takes in a queue, and an integer representing the position of the progress bar on screen.
The function then creates a BriarProgress object with that position, and starts consuming from
the queue until it receives None as an item in the queue (which indicates that all items have been consumed).
For each item received from the queue, it increments its counter by 1.

:param progress_queue: Communicate with the progress_consumer function
:param progress_position: Determine where the progress bar should be placed on the screen
:param total_len: Set the maximum value of the progress bar
:param options: Pass the options to the progress bar
:return: A function that takes a progress queue,
:doc-author: Joel Brogan
```

6.33.1.11 sigset_enroll()

```
def briar.cli.sigset.sigset_enroll (
              input_command = None )
```

```
The sigset_enroll function is used to enroll a signature set into the Briar system.

:return: A list of enrollments
:doc-author: Joel Brogan, David Bolme, Trelent
```

6.33.1.12 sigset_stats()

```
def briar.cli.sigset.sigset_stats (
              options = None,
              args = None )
```

```
The sigset_stats function parses a sigset file and prints out some statistics about the contents.

:param options: Pass in the command line arguments
:param args: Pass in the command line arguments
:return: The number of unique names, the total media count,
:doc-author: Joel Brogan, David Bolme, Trelent
```

6.33.2 Variable Documentation

6.33.2.1 my_pool

```
my_pool = None
```

6.33.2.2 proc_number

```
proc_number = None
```

6.33.2.3 service_address_number

```
service_address_number = None
```

6.34 briar.cli.status Namespace Reference

Functions

- def get_service_configuration (options=None, args=None, input_command=None, ret=False)
- def print_service_configuration (options=None, args=None)
- def status (options=None, args=None, input_command=None, ret=False)

  Conects to the server and gets status information.
- def statusParseOptions (inputCommand=None)

  Generate options for getting status and parse command line arguments into them.

6.34.1 Function Documentation

6.34.1.1 get_service_configuration()

```
def briar.cli.status.get_service_configuration (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

6.34.1.2 print_service_configuration()

```
def briar.cli.status.print_service_configuration (
              options = None,
              args = None )
```

6.34.1.3 status()

```
def briar.cli.status.status (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Conects to the server and gets status information.

Print results.

Returns

: None - results are printed

6.34.1.4 statusParseOptions()

```
def briar.cli.status.statusParseOptions (
              inputCommand = None )
```

Generate options for getting status and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.35 briar.cli.test Namespace Reference

Classes

- class BriarTest
- class BriarTestResult
- class DatabaseTest
- class DetectTest
- class EnrollTest
- class ExtractTest
- class Fore

Functions

- def detection_output_tests (detection_obj_loaded, testimage, return_media)
- def extraction_output_tests (template_obj_loaded, testimage, return_media)

Variables

- bool use_colorama = False

6.35.1 Function Documentation

6.35.1.1 detection_output_tests()

```
def briar.cli.test.detection_output_tests (
              detection_obj_loaded,
              testimage,
              return_media )
```

6.35.1.2 extraction_output_tests()

```
def briar.cli.test.extraction_output_tests (
              template_obj_loaded,
              testimage,
              return_media )
```

6.35.2 Variable Documentation

6.35.2.1 use_colorama

```
bool use_colorama = False
```

6.36 briar.cli.track Namespace Reference

Functions

- def get_tracklet_path (media_file, options, i, modality=None, media_id=None)
- def save_tracklets (media_file, tracklets, options, i, verbose=False, modality=None, media_id=None)
- def track (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a detection on the specified files.
- def trackRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)

Variables

- string TRACKLET_FILE_EXT = ".tracklet"

6.36.1 Function Documentation

6.36.1.1 get_tracklet_path()

```
def briar.cli.track.get_tracklet_path (
              media_file,
              options,
              i,
              modality = None,
              media_id = None )
```

6.36.1.2 save_tracklets()

```
def briar.cli.track.save_tracklets (
              media_file,
              tracklets,
              options,
              i,
              verbose = False,
              modality = None,
              media_id = None )
```

6.36.1.3 track()

```
def briar.cli.track.track (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs a detection on the specified files.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.36.1.4 trackRequestConstructor()

```
def briar.cli.track.trackRequestConstructor (
             briar_pb2.BriarMedia media,
             briar_pb2.BriarDurations durations,
              options_dict = {},
              det_list_list = None,
             str  database_name = None )
```

6.36.2 Variable Documentation

6.36.2.1 TRACKLET_FILE_EXT

```
string TRACKLET_FILE_EXT = ".tracklet"
```

6.37 briar.cli.verify Namespace Reference

Functions

- def addVerifyOptions (parser)

  Add options for verification to the parser.
- def save_verifications (media_file, reply, options, i, modality=None, media_id=None)
- def verify (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs an extract on the specified files.
- def verify_options2proto (options)
- def verifyParseOptions (inputCommand=None)

  Generate options for running verifications and parse command line arguments into them.

Variables

- string VERIFICATION_FILE_EXT = ".verification"

6.37.1 Function Documentation

6.37.1.1 addVerifyOptions()

```
def briar.cli.verify.addVerifyOptions (
              parser )
```

Add options for verification to the parser.

@type parser: optparse.OptionParser

Parameters

parser	A parser to modify in place by adding options

6.37.1.2 save_verifications()

```
def briar.cli.verify.save_verifications (
              media_file,
              reply,
              options,
              i,
              modality = None,
              media_id = None )
```

6.37.1.3 verify()

```
def briar.cli.verify.verify (
              options = None,
              args = None,
              input_command = None,
              ret = False )
```

Using the options specified in the command line, runs an extract on the specified files.

Writes results to disk to a location specified by the cmd arguments

Returns

: No return - Function writes results to disk

6.37.1.4 verify_options2proto()

```
def briar.cli.verify.verify_options2proto (
              options )
```

6.37.1.5 verifyParseOptions()

```
def briar.cli.verify.verifyParseOptions (
              inputCommand = None )
```

Generate options for running verifications and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.37.2 Variable Documentation

6.37.2.1 VERIFICATION_FILE_EXT

```
string VERIFICATION_FILE_EXT = ".verification"
```

6.38 briar.cli.viz Namespace Reference

Functions

- def viz ()

  Using the options specified in the command line, runs visualization on the specified files.
- def vizParseOptions ()

  Generate options for running detections and parse command line arguments into them.

6.38.1 Function Documentation

6.38.1.1 viz()

```
def briar.cli.viz.viz ( )
```

Using the options specified in the command line, runs visualization on the specified files.

Returns

: No return - Function writes results to disk

6.38.1.2 vizParseOptions()

```
def briar.cli.viz.vizParseOptions ( )
```

Generate options for running detections and parse command line arguments into them.

Returns

: 2 element Tuple of (optparse.Values, list) containing the parsed options and parameters respectively

6.39 briar.evaluation Namespace Reference

Namespaces

- full_evaluation
- stage1_probe_enroll
- stage2
- stage3_result_scoring

Functions

- def runStages (stages, report=False)

6.39.1 Function Documentation

6.39.1.1 runStages()

```
def briar.evaluation.runStages (
              stages,
              report = False )
```

6.40 briar.evaluation.full_evaluation Namespace Reference

Classes

- class Test000InitialConfig
- class Test001SigsetEnrollProbe
- class Test002ProbeDatabaseMerge
- class Test003SigsetEnrollMultiProbe
- class Test004MultiProbeDatabaseMerge
- class Test005SigsetEnrollGalleries
- class Test006GalleryDatabaseMerge
- class Test007SigsetScoreG1
- class Test008SigsetScoreGaitG1
- class Test009SigsetScoreFaceG1
- class Test010SigsetScoreWholeBodyG1
- class Test011SigsetScoreG2
- class Test012SigsetScoreGaitG2
- class Test013SigsetScoreFaceG2
- class Test014SigsetScoreWholeBodyG2
- class Test015SigsetBlendedScoreG1
- class Test016SigsetScoreBlendedGaitG1
- class Test017SigsetScoreBlendedFaceG1
- class Test018SigsetScoreBlendedWholeBodyG1
- class Test019SigsetBlendedScoreG2
- class Test020SigsetScoreBlendedGaitG2
- class Test021SigsetScoreBlendedFaceG2
- class Test022SigsetScoreBlendedWholeBodyG2
- class Test023MultiSigsetScoreG1
- class Test023SigsetSearchOutputFormatting
- class Test024MultiSigsetScoreGaitG1
- class Test024SigsetVerifyOutputFormatting
- class Test025MultiSigsetScoreFaceG1
- class Test026MultiSigsetScoreWholeBodyG1
- class Test027MultiSigsetScoreG2
- class Test028MultiSigsetScoreGaitG2
- class Test029MultiSigsetScoreFaceG2
- class Test030MultiSigsetScoreWholeBodyG2
- class Test031MultiSigsetBlendedScoreG1
- class Test032MultiSigsetScoreBlendedGaitG1
- class Test033MultiSigsetScoreBlendedFaceG1
- class Test034MultiSigsetScoreBlendedWholeBodyG1
- class Test035MultiSigsetBlendedScoreG2
- class Test036MultiSigsetScoreBlendedGaitG2
- class Test037MultiSigsetScoreBlendedFaceG2
- class Test038MultiSigsetScoreBlendedWholeBodyG2

Functions

- def compute_search (probe_db_name, gal_db_name, probe_sigset_path, output_path, modality=None, blended=False)
- def compute_verify (probe_db_name, gal_db_name, probe_sigset_path, gal_sigset_path, output_path, csv↩ _path, modality=None, blended=False)
- def get_info (self, db_name)
- def get_multi_info (self, base_db_name)
- def merge_dbs (self, db_name)
- def run_on_multi (self, base_db_name, mapped_function)
- def runall ()
- None setUpClass_main (cls)
- def setUpModule ()

Variables

- string args_string = " --progress "
- string blended_gallery_1_filename = "sigsets_gallery/Blended_Gallery_1.xml"
- string blended_gallery_2_filename = "sigsets_gallery/Blended_Gallery_2.xml"
- category
- string database_blended_gallery_1_name = 'db_eval_phase2_blended_gallery_1'
- string database_blended_gallery_2_name = 'db_eval_phase2_blended_gallery_2'
- string database_gallery_1_name = 'db_eval_phase2_gallery_1'
- string database_gallery_2_name = 'db_eval_phase2_gallery_2'
- string database_multi_probe_name = 'db_eval_phase2_multisubject_probe'
- string database_probe_name = 'db_eval_phase2_probe'
- DATABASE_SUFFIX_FLAG = os.getenv('BRIAR_DATABASE_SUFFIX_FLAG')
- DATASET_DIR = os.getenv('BRIAR_DATASET_DIR')
- string enroll_args = " --auto-create-database "
- EVAL_PHASE = os.getenv('BRIAR_EVAL_PHASE')
- EVALUATION_DIR = os.getenv('BRIAR_EVALUATION_DIR')
- EVALUATION_MULTISUBJECT_DIR = os.getenv('BRIAR_MULTISUBJECT_EVALUATION_DIR')
- gallery1_blended_sigset
- gallery1_sigset
- gallery2_blended_sigset
- gallery2_sigset
- gallery_1_blended_sigset_path
- string gallery_1_filename = "sigsets_gallery/Gallery_1.xml"
- gallery_1_sigset_path
- gallery_2_blended_sigset_path
- string gallery_2_filename = "sigsets_gallery/Gallery_2.xml"
- gallery_2_sigset_path
- generate_report = os.environ.get('REPORT', False)
- string media_args = " --no-save "
- module
- string multisubject_probe_filename = "sigsets_multiperson/Probe_BTS_briar-rd_multi.xml"
- int number_of_partitions = 1
- OUTPUT_DIR = os.getenv('BRIAR_EVALUATION_OUTPUT_DIR')
- list port_list = [ ]
- string probe_filename = "sigsets_main/Probe_BTS_briar-rd_ALL.xml"
- probe_multisubject_sigset
- probe_multisubject_sigset_path
- probe_sigset
- probe_sigset_path

- bool requires_database_merge = False
- run_multisubject_evaluation
- RUN_STAGES = os.getenv('RUN_STAGES',None)
- sortTestMethodsUsing
- list stages_temp = [ ]
- USE_SINGLE_SUBJECT = os.getenv('BRIAR_USE_SINGLE_SUBJECT')
- USES_FRONTEND_MERGING = os.getenv('BRIAR_USE_FRONTEND_MERGING')

6.40.1 Function Documentation

6.40.1.1 compute_search()

```
def briar.evaluation.full_evaluation.compute_search (
              probe_db_name,
              gal_db_name,
              probe_sigset_path,
              output_path,
              modality = None,
              blended = False )
```

6.40.1.2 compute_verify()

```
def briar.evaluation.full_evaluation.compute_verify (
              probe_db_name,
              gal_db_name,
              probe_sigset_path,
              gal_sigset_path,
              output_path,
              csv_path,
              modality = None,
              blended = False )
```

6.40.1.3 get_info()

```
def briar.evaluation.full_evaluation.get_info (
              self,
              db_name )
```

6.40.1.4 get_multi_info()

```
def briar.evaluation.full_evaluation.get_multi_info (
              self,
              base_db_name )
```

6.40.1.5 merge_dbs()

```
def briar.evaluation.full_evaluation.merge_dbs (
              self,
              db_name )
```

6.40.1.6 run_on_multi()

```
def briar.evaluation.full_evaluation.run_on_multi (
              self,
              base_db_name,
              mapped_function )
```

6.40.1.7 runall()

```
def briar.evaluation.full_evaluation.runall ( )
```

6.40.1.8 setUpClass_main()

```
None briar.evaluation.full_evaluation.setUpClass_main (
              cls )
```

6.40.1.9 setUpModule()

```
def briar.evaluation.full_evaluation.setUpModule ( )
```

6.40.2 Variable Documentation

6.40.2.1 args_string

```
string args_string = " --progress "
```

6.40.2.2 blended_gallery_1_filename

```
string blended_gallery_1_filename = "sigsets_gallery/Blended_Gallery_1.xml"
```

6.40.2.3 blended_gallery_2_filename

```
string blended_gallery_2_filename = "sigsets_gallery/Blended_Gallery_2.xml"
```

6.40.2.4 category

```
category
```

6.40.2.5 database_blended_gallery_1_name

```
string database_blended_gallery_1_name = 'db_eval_phase2_blended_gallery_1'
```

6.40.2.6 database_blended_gallery_2_name

```
string database_blended_gallery_2_name = 'db_eval_phase2_blended_gallery_2'
```

6.40.2.7 database_gallery_1_name

```
string database_gallery_1_name = 'db_eval_phase2_gallery_1'
```

6.40.2.8 database_gallery_2_name

```
string database_gallery_2_name = 'db_eval_phase2_gallery_2'
```

6.40.2.9 database_multi_probe_name

```
string database_multi_probe_name = 'db_eval_phase2_multisubject_probe'
```

6.40.2.10 database_probe_name

```
string database_probe_name = 'db_eval_phase2_probe'
```

6.40.2.11 DATABASE_SUFFIX_FLAG

```
string DATABASE_SUFFIX_FLAG = os.getenv('BRIAR_DATABASE_SUFFIX_FLAG')
```

6.40.2.12 DATASET_DIR

```
DATASET_DIR = os.getenv('BRIAR_DATASET_DIR')
```

6.40.2.13 enroll_args

```
string enroll_args = " --auto-create-database "
```

6.40.2.14 EVAL_PHASE

```
EVAL_PHASE = os.getenv('BRIAR_EVAL_PHASE')
```

6.40.2.15 EVALUATION_DIR

```
EVALUATION_DIR = os.getenv('BRIAR_EVALUATION_DIR')
```

6.40.2.16 EVALUATION_MULTISUBJECT_DIR

```
EVALUATION_MULTISUBJECT_DIR = os.getenv('BRIAR_MULTISUBJECT_EVALUATION_DIR')
```

6.40.2.17 gallery1_blended_sigset

```
gallery1_blended_sigset
```

6.40.2.18 gallery1_sigset

```
gallery1_sigset
```

6.40.2.19 gallery2_blended_sigset

```
gallery2_blended_sigset
```

6.40.2.20 gallery2_sigset

```
gallery2_sigset
```

6.40.2.21 gallery_1_blended_sigset_path

```
gallery_1_blended_sigset_path
```

6.40.2.22 gallery_1_filename

```
string gallery_1_filename = "sigsets_gallery/Gallery_1.xml"
```

6.40.2.23 gallery_1_sigset_path

```
gallery_1_sigset_path
```

6.40.2.24 gallery_2_blended_sigset_path

```
gallery_2_blended_sigset_path
```

6.40.2.25 gallery_2_filename

```
string gallery_2_filename = "sigsets_gallery/Gallery_2.xml"
```

6.40.2.26 gallery_2_sigset_path

```
gallery_2_sigset_path
```

6.40.2.27 generate_report

```
generate_report = os.environ.get('REPORT', False)
```

6.40.2.28 media_args

```
string media_args = " --no-save "
```

6.40.2.29 module

```
module
```

6.40.2.30 multisubject_probe_filename

```
string multisubject_probe_filename = "sigsets_multiperson/Probe_BTS_briar-rd_multi.xml"
```

6.40.2.31 number_of_partitions

```
int number_of_partitions = 1
```

6.40.2.32 OUTPUT_DIR

```
OUTPUT_DIR = os.getenv('BRIAR_EVALUATION_OUTPUT_DIR')
```

6.40.2.33 port_list

```
list port_list = []
```

6.40.2.34 probe_filename

```
string probe_filename = "sigsets_main/Probe_BTS_briar-rd_ALL.xml"
```

6.40.2.35 probe_multisubject_sigset

```
probe_multisubject_sigset
```

6.40.2.36 probe_multisubject_sigset_path

```
probe_multisubject_sigset_path
```

6.40.2.37 probe_sigset

```
probe_sigset
```

6.40.2.38 probe_sigset_path

```
probe_sigset_path
```

6.40.2.39 requires_database_merge

```
bool requires_database_merge = False
```

6.40.2.40 run_multisubject_evaluation

```
run_multisubject_evaluation
```

6.40.2.41 RUN_STAGES

```
list RUN_STAGES = os.getenv('RUN_STAGES',None)
```

6.40.2.42 sortTestMethodsUsing

```
sortTestMethodsUsing
```

6.40.2.43 stages_temp

```
list stages_temp = [ ]
```

6.40.2.44 USE_SINGLE_SUBJECT

```
string USE_SINGLE_SUBJECT = os.getenv('BRIAR_USE_SINGLE_SUBJECT')
```

6.40.2.45 USES_FRONTEND_MERGING

```
bool USES_FRONTEND_MERGING = os.getenv('BRIAR_USE_FRONTEND_MERGING')
```

6.41 briar.evaluation.stage1_probe_enroll Namespace Reference

Variables

- generate_report = os.environ.get('REPORT',False)
- main = unittest.TestProgram

6.41.1 Variable Documentation

6.41.1.1 generate_report

```
generate_report = os.environ.get('REPORT',False)
```

6.41.1.2 main

```
main = unittest.TestProgram
```

6.42 briar.evaluation.stage2 Namespace Reference

Variables

- generate_report = os.environ.get('REPORT', False)
- main = unittest.TestProgram

6.42.1 Variable Documentation

6.42.1.1 generate_report

```
generate_report = os.environ.get('REPORT', False)
```

6.42.1.2 main

```
main = unittest.TestProgram
```

6.43 briar.evaluation.stage3_result_scoring Namespace Reference

Variables

- generate_report = os.environ.get('REPORT', False)
- main = unittest.TestProgram

6.43.1 Variable Documentation

6.43.1.1 generate_report

```
generate_report = os.environ.get('REPORT', False)
```

6.43.1.2 main

```
main = unittest.TestProgram
```

6.44 briar.grpc_json Namespace Reference

I couldn't find a good library to write gRPC/protobuf objects to disk so this is my own implementation of a generalized json converter for said objects.

Classes

- class GrpcDecoder
  Object which extends the JSONDecoded to allow it to read saved gRPC files.
- class GrpcEncoder
  Encoder class which extends the normal JSON encoder to allow the encoding of gRPC objects.

Functions

- def dict_to_proto_obj (obj_dict, options=None)
  *Take the object dictionary, read the dict which is saved in in the '**class**' key, and initialize it with values stored in the dictionary's key/value pairs.*
- def load (load_path, options=None)
  Load the json file at the given directory, reloading dictionaries with "__class__" fields into the specified objects and initializing them with values defined by key/value pairs within the dictionary.
- def proto_obj_to_dict (obj, options=None)
  Takes a general gRPC/protobuf object, eliminates the unnecessary fields, and stores the data in a dict.
- def save (json_obj, save_path, options=None)
  Save a list or dictionary containing protobuf classes to a json file.

Variables

- list ATTRIB_IGNORE

6.44.1 Detailed Description

I couldn't find a good library to write gRPC/protobuf objects to disk so this is my own implementation of a generalized json converter for said objects.

Good for writing the objects themselves but be aware that it will write any vectors/matrixes/images stored within the objects to disk so it may be a good idea to remove said data before converting to json if you care about performance in your implementations.

6.44.2 Function Documentation

6.44.2.1 dict_to_proto_obj()

```
def briar.grpc_json.dict_to_proto_obj (
              obj_dict,
              options = None )
```

Take the object dictionary, read the dict which is saved in in the '**class**' key, and initialize it with values stored in the dictionary's key/value pairs.

Parameters

obj_dict	dict: A dictionary specifically containing a '**class**' key/value pair storing the full module path to the object.

Returns

: A gRPC object defined by '**class**'

6.44.2.2 load()

```
def briar.grpc_json.load (
              load_path,
              options = None )
```

Load the json file at the given directory, reloading dictionaries with "__class__" fields into the specified objects and initializing them with values defined by key/value pairs within the dictionary.

Parameters

load_path	str: Path to the json file to load

Returns

: The contents of the json file deserialized into the appropriate objects

6.44.2.3 proto_obj_to_dict()

```
def briar.grpc_json.proto_obj_to_dict (
              obj,
              options = None )
```

Takes a general gRPC/protobuf object, eliminates the unnecessary fields, and stores the data in a dict.

Classes will be saved as dictionaries with a "__class__" attribute. This should be the full import path to the class within its module.

Parameters

obj	Any gRPC object generated by protobuf files

Returns

: A dictionary representing the object

6.44.2.4 save()

```
def briar.grpc_json.save (
              json_obj,
              save_path,
              options = None )
```

Save a list or dictionary containing protobuf classes to a json file.

Parameters

json_obj	list||dict: List or dict containing data to save
save_path	str: Path to the file to save

Returns: None

6.44.3 Variable Documentation

6.44.3.1 ATTRIB_IGNORE

```
list ATTRIB_IGNORE
```

Initial value:

```
=  ['ByteSize', 'Clear', 'ClearExtension', 'ClearField', 'CopyFrom',
                'DESCRIPTOR', 'DiscardUnknownFields', 'Extensions',
                'FindInitializationErrors', 'FromString', 'HasExtension',
                'HasField', 'IsInitialized', 'ListFields', 'MergeFrom',
                'MergeFromString', 'ParseFromString', 'RegisterExtension',
                'SerializePartialToString', 'SerializeToString',
                'SetInParent', 'UnknownFields', 'WhichOneof',
                '_CheckCalledFromGeneratedFile', '_SetListener',
                '__deepcopy__', '__delattr__', '__dir__', '__doc_',
                 '_extensions_by_name', '_extensions_by_number', 'EnumTypeWrapper']
```

6.45 briar.media Namespace Reference

Namespaces

- VideoStream
- visualize

Classes

- class BriarProgress
- class BriarVideoIterator
- class ImageIterator
- class MediaSetIterator
- class ThreadedVideoIterator
- class VideoIterator

Functions

- def aenumerate (asequence, start=0)
- def decodeMedia (media_pb, newsource=None)

  Convert protobuf media into a numpy array.
- def enroll_frames_iter (database_name, video, detect_options=None, extract_options=None, enroll_↩options=None, det_list_list=None, whole_image=False, request_start=-1)

  Iterates the paths in the media file list, loading them one by one and yielding grpc enroll requests.
- def enroll_frames_iter_async (database_name, video, detect_options=None, extract_options=None, enroll↩_options=None, det_list_list=None, whole_image=False, request_start=-1)

  Asyncronously Iterates the paths in the media file list, loading them one by one and yielding grpc enroll requests.
- def file_iter (list[str] media_files, briar_pb2.DetectionOptions clientoptions=None, dict options_dict=None, str database_name=None, bool verbose=False, int request_start=-1, callable requestConstructor=None)
- def frame_iter (media_iterator, clientoptions=None, dict options_dict={}, database_name=None, det_list_↩list=None, whole_image=False, request_start=-1, requestConstructor=None)
- def ImageGenerator (filepath, start=None, stop=None, unit=None, options=None)
- def isFinalFrame (request)
- def single_frame_generate (frame, frame_num, ittype, filepath, start_frame, stop_frame, fps, video_length, clientoptions=None, dict options_dict={}, database_name=None, det_list_list=None, whole_image=False, request_start=-1, frame_load_time_start=-1, file_level_client_time_end=-1, requestConstructor=None)

6.45.1 Function Documentation

6.45.1.1 aenumerate()

```
def briar.media.aenumerate (
              asequence,
              start = 0 )
```

```
Asynchronously enumerate an async iterator from a given start value
```

6.45.1.2 decodeMedia()

```
def briar.media.decodeMedia (
              media_pb,
              newsource = None )
```

Convert protobuf media into a numpy array.

Parameters

media_pb	briar_pb2.BriarMedia

return: numpy.array

6.45.1.3 enroll_frames_iter()

```
def briar.media.enroll_frames_iter (
              database_name,
              video,
              detect_options = None,
              extract_options = None,
              enroll_options = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1 )
```

Iterates the paths in the media file list, loading them one by one and yielding grpc enroll requests.

@type database_name: str

Parameters

database_name	Name of the database to enroll templates in

@type video: an iterator that generates cv2 frames

Parameters

media_files	Paths to the media files to enroll from

@type detect_options: briar_pb2.DetectionOptions

Parameters

options	Command line options in protobuf format which control detection functionality

@type optons: briar_pb2.ExtractOptions

Parameters

options	Command line options in protobuf format which control extraction functionality

@type optons: briar_pb2.EnrollOptions

Parameters

options	Command line options in protobuf format which control enrollment functionality

@type det_list_list: List of list of briar_pb2.Detection

Parameters

det_list_list	If not None, it will contains 1 list of detections per media file

@type whole_image: boolean

Parameters

	Ignore detections and run an extract on the whole image

@type request_start: int

Parameters

	timestamp of when the request started

@yield: briar_service_pb2.EnrollRequest

6.45.1.4 enroll_frames_iter_async()

```
def briar.media.enroll_frames_iter_async (
              database_name,
              video,
              detect_options = None,
              extract_options = None,
              enroll_options = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1 )
```

Asyncronously Iterates the paths in the media file list, loading them one by one and yielding grpc enroll requests.

This method should provide better video load performance and better request generation performance.

@type database_name: str

Parameters

database_name	Name of the database to enroll templates in

@type video: an iterator that generates cv2 frames

Parameters

media_files	Paths to the media files to enroll from

@type detect_options: briar_pb2.DetectionOptions

Parameters

options	Command line options in protobuf format which control detection functionality

@type optons: briar_pb2.ExtractOptions

Parameters

options	Command line options in protobuf format which control extraction functionality

@type optons: briar_pb2.EnrollOptions

Parameters

options	Command line options in protobuf format which control enrollment functionality

@type det_list_list: List of list of briar_pb2.Detection

Parameters

det_list_list	If not None, it will contains 1 list of detections per media file

@type whole_image: boolean

Parameters

	Ignore detections and run an extract on the whole image

@type request_start: int

Parameters

	timestamp of when the request started

@yield: briar_service_pb2.EnrollRequest

6.45.1.5 file_iter()

```
def briar.media.file_iter (
             list[str] media_files,
             briar_pb2.DetectionOptions  clientoptions = None,
             dict  options_dict = None,
             str  database_name = None,
             bool  verbose = False,
             int  request_start = -1,
             callable  requestConstructor = None )
```

```
Iterates the paths in the media file list, loading them one by one and yielding grpc requests generated by <re

Args:
    media_files (list of str): Paths to the media files to enroll from.
    clientoptions (briar_pb2.DetectionOptions, optional): Command line options in protobuf format which contro
    options_dict (dict, optional): Additional options in dictionary format.
    database_name (str, optional): Name of the database.
    verbose (bool, optional): If True, enables verbose output.
    request_start (int, optional): Starting index for the request.
    requestConstructor (callable, optional): Function to construct the request.

Yields:
    briar_service_pb2.DetectRequest: gRPC request generated by requestConstructor.
```

6.45.1.6 frame_iter()

```
def briar.media.frame_iter (
              media_iterator,
              clientoptions = None,
             dict  options_dict = {},
              database_name = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1,
              requestConstructor = None )
```

```
The frame_iter function is a generator that takes in an iterator of frames, and returns a request object for e
The request object contains the following information:
    - The frame number (int)
    - The type of media_iterator (str)
    - The filepath to the video/image sequence (str)
    - A list containing all detections from previous iterations, if applicable. If not applicable, this will b

:param media_iterator: Iterate through the frames of a video
:param clientoptions: Pass the client options to the request
:param options_dict : dict: Pass in the options dictionary from the client
:param database_name: Specify the database name
:param det_list_list: Pass in a list of detection lists
:param whole_image: Determine whether or not to send the whole image
:param request_start: Keep track of when the request was begun
:param requestConstructor: Create a request object
:return: A generator that yields a single frame request
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.45.1.7 ImageGenerator()

```
def briar.media.ImageGenerator (
              filepath,
              start = None,
              stop = None,
              unit = None,
              options = None )
```

```
The ImageGenerator function is a generator that yields frames from an image file.

:param filepath: Specify the location of the image
:param start: Set the starting frame of the video
:param stop: Stop the generator at a certain frame
:param unit: Determine the unit of time that is used for the start and stop parameters
:return: A single frame, so we need to wrap it in a loop
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.45.1.8 isFinalFrame()

```
def briar.media.isFinalFrame (
              request )
```

6.45.1.9 single_frame_generate()

```
def briar.media.single_frame_generate (
              frame,
              frame_num,
              ittype,
              filepath,
              start_frame,
              stop_frame,
              fps,
              video_length,
              clientoptions = None,
             dict  options_dict = {},
              database_name = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1,
              frame_load_time_start = -1,
              file_level_client_time_end = -1,
              requestConstructor = None )
```

```
The single_frame_generate function is used to generate a single frame of BriarMedia.
It takes in the following parameters:
- frame: The image or video frame that will be converted into BriarMedia.
- ittype: The type of iterator being used (ImageIterator, VideoIterator, MediaSetIterator).  This is needed be
- filepath: A string containing the path and filename for the media being processed by BRIAR.

:param frame: Pass in the frame to be processed
:param frame_num: Keep track of the frame number
:param ittype: Determine the type of iterator used to generate frames
:param filepath: Indicate the path to the file that is being processed
:param start_frame: Indicate the starting frame of a video, and stop_frame is used to indicate the ending fram
:param stop_frame: Determine the last frame to be processed
:param video_length: Set the frame_count in the briarmedia
:param clientoptions: Pass the options to the client
:param options_dict : dict: Pass the options dictionary to the server
:param database_name: Specify the database to which the request should be sent
:param det_list_list: Pass the detection list from one frame to another
:param whole_image: Determine if the whole image is sent to briar or just a cropped version
:param request_start: Record the time at which the client api was called
:param file_level_client_time_end: Record the time at which the file-level operations of the briar client api
:param requestConstructor: Create a request object
:return: A request object, which is a briarrequest message
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.46 briar.media.VideoStream Namespace Reference

Classes

- class FileVideoStream_cv2
- class FileVideoStream_imageio

6.47 briar.media.visualize Namespace Reference

Classes

- class match_matrix_visualizer

Functions

- def decode_track (tracklet, framenum=None, newsource=None)
- def get_frame (vidfile)
- def playVideo (vidfiles, titles=None, attributes=None, isvideo=True)
- def update_annot (ind, visualizer, pltloc, playvid=False)
- def update_annot_filename_only (ind, visualizer, pltloc)
- def visualize_detection (detection_path)
- def visualize_matches (matches_path)
- def visualize_track (track_path, options)
- def windowclick (event, visualizer)
- def windowhover (event, visualizer, playvid=False)
- def windowhover_filename_only (event, visualizer)

Variables

- string fdir = "/Users/2r6/Projects/briar/briar-api/media/test_probe/clinton3.matches"
- list files = [os.path.join(fdir, f) for f in os.listdir(fdir)]

6.47.1 Function Documentation

6.47.1.1 decode_track()

```
def briar.media.visualize.decode_track (
              tracklet,
              framenum = None,
              newsource = None )
```

6.47.1.2 get_frame()

```
def briar.media.visualize.get_frame (
              vidfile )
```

6.47.1.3 playVideo()

```
def briar.media.visualize.playVideo (
              vidfiles,
              titles = None,
              attributes = None,
              isvideo = True )
```

6.47.1.4 update_annot()

```
def briar.media.visualize.update_annot (
              ind,
              visualizer,
              pltloc,
              playvid = False )
```

6.47.1.5 update_annot_filename_only()

```
def briar.media.visualize.update_annot_filename_only (
              ind,
              visualizer,
              pltloc )
```

6.47.1.6 visualize_detection()

```
def briar.media.visualize.visualize_detection (
              detection_path )
```

6.47.1.7 visualize_matches()

```
def briar.media.visualize.visualize_matches (
              matches_path )
```

6.47.1.8 visualize_track()

```
def briar.media.visualize.visualize_track (
              track_path,
              options )
```

6.47.1.9 windowclick()

```
def briar.media.visualize.windowclick (
              event,
              visualizer )
```

6.47.1.10 windowhover()

```
def briar.media.visualize.windowhover (
              event,
              visualizer,
              playvid = False )
```

6.47.1.11 windowhover_filename_only()

```
def briar.media.visualize.windowhover_filename_only (
              event,
              visualizer )
```

6.47.2 Variable Documentation

6.47.2.1 fdir

```
string fdir = "/Users/2r6/Projects/briar/briar-api/media/test_probe/clinton3.matches"
```

6.47.2.2 files

```
list files = [os.path.join(fdir, f) for f in os.listdir(fdir)]
```

6.48 briar.media_converters Namespace Reference

Contained in this are functions for converting numpy arrays into various protobuf objects and back again since numpy arrays cannot be sent directly over gRPC.

Functions

- def attribute_find (key, List[briar_pb2.Attribute] attributes)
- def attribute_proto2val (briar_pb2.Attribute attribute)
- def attribute_retrieve (briar_pb2.Attribute attribute, return_type=False)
- def attribute_val2proto (key, val, briar_pb2.BriarDataType override_type=None)
- def check_if_delete_request (srvc_pb2.DatabaseInsertRequest request)
- def check_if_delete_request_due_to_error (srvc_pb2.DatabaseInsertRequest request)
- def get_entry_id_list (srvc_pb2.DatabaseInsertRequest request)
- def image_cv2proto (im, compression='uint8', quality=99, flip_channels=True)

  Convert a cv2 numpy array to a protobuf format.
- def image_file2proto (imfile, path_map={})
- def image_np2proto (im, compression='uint8', quality=99, flip_channels=True)

  Convert a numpy array to a protobuf format.
- def image_proto2cv (pb_data, flip_channels=False)

  Convert a protobuf BriarMedia image to a cv2 numpy array.
- def image_proto2np (pb_data, flip_channels=True)

  Convert a protobuf image to a numpy array.
- def matrix_np2proto (mat)

  Convert a numpy matrix into a BriarMatrix.
- def matrix_proto2np (protomat)

  Convert a protobuf matrix into a numpy matrix.
- def modality_proto2string (modality)
- def modality_string2proto (modality)
- def pathmap_path2remotepath (path, path_map, exclude_cases_containing_folder=['mugshots'])
- def pathmap_str2dict (path_map)
- def subjectID_int2str (subjectid)
- def subjectID_str2int (subjectid)
- def subjectList_list2string (subject_list_str, chomp=True)
- def subjectList_string2list (subject_list)
- def tracklet_list2proto (track_list)
- def vector_np2proto (vec)

  Convert a 1 dimensional np array into a BriarVector.
- def vector_proto2np (protovec)

  Convert a protobuf vector into a numpy array.
- def video_file2proto (vidfile, start, end, path_map={})

Variables

- dictionary attribute_type_name_map = {'int': 'ivalue', 'float': 'fvalue','string':'text'}
- dictionary modalityDict
- dictionary reverseModalityDict = {modalityDict[k]: k for k in modalityDict}

6.48.1 Detailed Description

Contained in this are functions for converting numpy arrays into various protobuf objects and back again since numpy arrays cannot be sent directly over gRPC.

6.48.2 Function Documentation

6.48.2.1 attribute_find()

```
def briar.media_converters.attribute_find (
             key,
            List[briar_pb2.Attribute] attributes )
```

```
Find an attribute by its key in a list of attributes.

Parameters:
    key (Any): The key of the attribute to find.
    attributes (list[briar_pb2.Attribute]): The list of attributes to search through.

Returns:
    Any: The value of the attribute if found, otherwise None.
```

6.48.2.2 attribute_proto2val()

```
def briar.media_converters.attribute_proto2val (
            briar_pb2.Attribute attribute )
```

```
The attribute_proto2val function takes a briar_pb2.Attribute object and returns the value of that attribute ir

:param attribute: briar_pb2.Attribute: Store the attribute
:return: The value of the attribute
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.3 attribute_retrieve()

```
def briar.media_converters.attribute_retrieve (
            briar_pb2.Attribute attribute,
             return_type = False )
```

```
The attribute_retrieve function takes in a briar_pb2.Attribute object and returns the value of that attribute
as a python object. The function also has an optional parameter, return_type, which if set to True will returr
both the value of the attribute and its type as a tuple.

:param attribute: briar_pb2.Attribute: Specify the attribute to retrieve
:param return_type: Determine if the type of the attribute should be returned as well
:return: A tuple of the attribute value and type
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.4 attribute_val2proto()

```
def briar.media_converters.attribute_val2proto (
              key,
              val,
             briar_pb2.BriarDataType  override_type = None )
```

```
The attribute_val2proto function takes a key, value pair and converts it to a BriarAttribute protobuf.
The function will attempt to determine the type of the value automatically, but you can override this by passi
If you pass in an override_type parameter, then the val must be of that type or else it will raise an exceptio

:param key: Identify the attribute
:param val: Set the value of the attribute
:param override_type :briar_pb2.BriarDataType: Force the type of the attribute
:return: A briar_pb2
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.5 check_if_delete_request()

```
def briar.media_converters.check_if_delete_request (
             srvc_pb2.DatabaseInsertRequest request )
```

```
Checks if the delete request is valid.

Parameters:
request (srvc_pb2.DatabaseInsertRequest): The request containing the ids.

Returns:
bool: True if the delete request is valid, False otherwise.
```

6.48.2.6 check_if_delete_request_due_to_error()

```
def briar.media_converters.check_if_delete_request_due_to_error (
             srvc_pb2.DatabaseInsertRequest request )
```

```
Check if a delete request is due to an error.

Args:
    request (srvc_pb2.DatabaseInsertRequest): The delete request to be checked.

Returns:
    bool: True if the delete request is due to an error, False otherwise.
```

6.48.2.7 get_entry_id_list()

```
def briar.media_converters.get_entry_id_list (
             srvc_pb2.DatabaseInsertRequest request )
```

```
    Get Entry ID List

    This method accepts a request object of type srvc_pb2.DatabaseInsertRequest and
    returns a list of entry IDs.

    Parameters:
        request (srvc_pb2.DatabaseInsertRequest): The request object containing entry IDs

    Returns:
        list: A list of entry IDs
```

6.48.2.8 image_cv2proto()

```
def briar.media_converters.image_cv2proto (
              im,
              compression = 'uint8',
              quality = 99,
              flip_channels = True )
```

Convert a cv2 numpy array to a protobuf format.

Parameters

img	numpy.array: array containing the image to convert to BriarMedia
compression	str: What compression to use. Can be 'uint8', 'png', 'jpg'
quality	int: 0-100 How much do you want to mutilate the image in the name of saving memory?
flip_channels	boolean : Flips the channels dimension of a cv2-type numpy image. This could translate an image from RGB->BGR or vice-versa. Default is True.

return: briar_pb2.BriarMedia

6.48.2.9 image_file2proto()

```
def briar.media_converters.image_file2proto (
              imfile,
              path_map = {} )
```

6.48.2.10 image_np2proto()

```
def briar.media_converters.image_np2proto (
              im,
              compression = 'uint8',
              quality = 99,
              flip_channels = True )
```

Convert a numpy array to a protobuf format.

Parameters

img	numpy.array: array containing the image to convert to BriarMedia
compression	str: What compression to use. Can be 'uint8', 'png', 'jpg'
quality	int: 0-100 How much do you want to mutilate the image in the name of saving memory?
flip_channels	boolean : Flips the channels dimension of a cv2-type numpy image. This could translate an image from RGB->BGR or vice-versa. Default is True.

return: briar_pb2.BriarMedia

6.48.2.11 image_proto2cv()

```
def briar.media_converters.image_proto2cv (
              pb_data,
              flip_channels = False )
```

Convert a protobuf BriarMedia image to a cv2 numpy array.

Parameters

pb_data	briar_pb2.BriarMedia: Protobuf object containing image data
flip_channels	boolean : Flips the channels dimension of a cv2-type numpy image. This could translate an image from RGB->BGR or vice-versa. Default is False.

Returns

: numpy.array cv2 formatted np array containing image

6.48.2.12 image_proto2np()

```
def briar.media_converters.image_proto2np (
              pb_data,
              flip_channels = True )
```

Convert a protobuf image to a numpy array.

Parameters

pb_data	briar_pb2.BriarMedia: Protobuf object containing image data

Returns

: np.array

6.48.2.13 matrix_np2proto()

```
def briar.media_converters.matrix_np2proto (
              mat )
```

Convert a numpy matrix into a BriarMatrix.

Parameters

mat	numpy.array: Matrix to convert

Returns

: briar_pb2.BriarMatrix

6.48.2.14 matrix_proto2np()

```
def briar.media_converters.matrix_proto2np (
              protomat )
```

Convert a protobuf matrix into a numpy matrix.

Parameters

protomat	briar_pb2.BriarMatrix: Protobuf matrix to convert

Returns

: numpy.array

6.48.2.15 modality_proto2string()

```
def briar.media_converters.modality_proto2string (
              modality )
```

```
The modality_proto2string function takes a modality and returns the string representation of that modality.

:param modality: Determine which modality the data is from
:return: The string representation of the modality
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.16 modality_string2proto()

```
def briar.media_converters.modality_string2proto (
              modality )
```

```
The modality_string2proto function takes a string and returns the corresponding modality enum value.

:param modality: Determine the type of data that is being used
:return: The modality of the image
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.17 pathmap_path2remotepath()

```
def briar.media_converters.pathmap_path2remotepath (
              path,
              path_map,
              exclude_cases_containing_folder = ['mugshots'] )
```

```
Maps a local file path to a remote file path based on a given path map and excludes specific folders.

Args:
    path (str): The local file path to be mapped.
    path_map (dict): A dictionary containing mappings from local folders to remote folders.
    exclude_cases_containing_folder (list, optional): A list of folder names. If any of these folder names are
*shots’].

Returns:
    str: The mapped remote file path.

Raises:
    None

Example:
    path = ’/Users/johndoe/Documents/pictures/mugshots/2020/abc.jpg’
    path_map = {
        ’pictures’: ’photo’,
        ’mugshots’: ’headshots’
    }
    exclude_cases_containing_folder = [’mugshots’]

    remote_path = pathmap_path2remotepath(path, path_map, exclude_cases_containing_folder)
    print(remote_path)
    # Output: ’/Users/johndoe/Documents/photo/headshots/2020/abc.jpg’
```

6.48.2.18 pathmap_str2dict()

```
def briar.media_converters.pathmap_str2dict (
              path_map )
```

```
The pathmap_str2dict function takes a string of the form
’key:value,key2:value2’
and returns a dictionary with keys and values as follows:
{ key : value, key2 : value2 }.

:param path_map: Map the path of a file to another path
:return: A dictionary mapping the paths in the
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.19 subjectID_int2str()

```
def briar.media_converters.subjectID_int2str (
              subjectid )
```

```
The subjectID_int2str function takes an integer subject ID and converts it to a string.
The function is used in the process of creating a new subject ID for each participant.
It ensures that all IDs are 5 digits long, with leading zeros if necessary.

:param subjectid: Convert the subjectid from an integer to a string
:return: A string of the subject id with a g in front and zeros to fill out the rest
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.20 subjectID_str2int()

```
def briar.media_converters.subjectID_str2int (
              subjectid )
```

```
The subjectID_str2int function is for legacy systems that utilize only numbers as gallery entries.
This function takes a string of the form &quot;G####&quot; and returns an integer of the form ####.

:param subjectid: Create a subjectid that is only numbers
:return: An integer value of the subjectid
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.21 subjectList_list2string()

```
def briar.media_converters.subjectList_list2string (
              subject_list_str,
              chomp = True )
```

```
Convert a list of subjects to a string representation.

Parameters:
subject_list_str (list): A list containing the subjects as strings.

Returns:
str: A string representation of the list of subjects, where each subject is separated by a comma.

Example:
>>> subjects = ['Math', 'Science', 'English']
>>> subjectList_list2string(subjects)
'Math,Science,English'
```

6.48.2.22 subjectList_string2list()

```
def briar.media_converters.subjectList_string2list (
              subject_list )
```

```
Convert a comma-separated string of subjects to a list of strings.

Parameters:
    subject_list (str): A comma-separated string of subjects.

Returns:
    List[str]: A list of subjects.

Example:
    >>> subject_string2list("Math,Science,English")
    ['Math', 'Science', 'English']
```

6.48.2.23 tracklet_list2proto()

```
def briar.media_converters.tracklet_list2proto (
              track_list )
```

```
The tracklet_list2proto function takes a list of dictionaries and converts them into a tracklet proto.
The dictionary must have the following keys:
x, y, width, height (all floats)
frame (int)
confidence (float)

:param track_list: Create a tracklet proto
:return: A protobuf object
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.2.24 vector_np2proto()

```
def briar.media_converters.vector_np2proto (
              vec )
```

Convert a 1 dimensional np array into a BriarVector.

Parameters

vec	numpy.array: Numpy array contaiing vector data

Returns

: briar_pb2.BriarVector

6.48.2.25 vector_proto2np()

```
def briar.media_converters.vector_proto2np (
              protovec )
```

Convert a protobuf vector into a numpy array.

Parameters

protovec	briar_pb2.BriarVector: Protobuf object containing vector info

Returns

: numpy.array

6.48.2.26 video_file2proto()

```
def briar.media_converters.video_file2proto (
              vidfile,
              start,
              end,
              path_map = {} )
```

```
The video_file2proto function takes in a video file, start frame, end frame and path_map.
The path_map is used to map the local paths of the video files to their server side paths.
This function returns a BriarMedia proto object with all of its fields filled out.

:param vidfile: Get the video file name
:param start: Specify the frame number of the first frame in a video
:param end: Determine the last frame of a video
:param path_map: Map the local path to the server path
:return: A briar_pb2
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.48.3 Variable Documentation

6.48.3.1 attribute_type_name_map

```
dictionary attribute_type_name_map = {'int':  'ivalue', 'float':  'fvalue','string':'text'}
```

6.48.3.2 modalityDict

```
dictionary modalityDict
```

Initial value:

```
=  {'whole_body': briar_pb2.WHOLE_BODY,
                'wholeBody': briar_pb2.WHOLE_BODY,
                'wholebody': briar_pb2.WHOLE_BODY,
                'face': briar_pb2.FACE,
                'gait': briar_pb2.GAIT,
                'unspecified': briar_pb2.UNSPECIFIED}
```

6.48.3.3 reverseModalityDict

```
dictionary reverseModalityDict = {modalityDict[k]:  k for k in modalityDict}
```

6.49 briar.sigset Namespace Reference

Namespaces

- parse

6.50 briar.sigset.parse Namespace Reference

Functions

- def create_test_sigset (str sigset_probe_file, str sigset_gallery_file, str base_dir, output_dir)
- def expandTree (root, level=0, spaces=3)
- def parseBriarSigset (filename)

Variables

- args = sys.argv

6.50.1 Function Documentation

6.50.1.1 create_test_sigset()

```
def briar.sigset.parse.create_test_sigset (
             str sigset_probe_file,
             str sigset_gallery_file,
             str base_dir,
              output_dir )
```

6.50.1.2 expandTree()

```
def briar.sigset.parse.expandTree (
              root,
              level = 0,
              spaces = 3 )
```

```
The expandTree function takes a root element and prints out the tag, length of padding, and level for each ele
The function is recursive, so it will print out all elements in the tree.

:param root: Pass the root of the tree to be expanded
:param level: Keep track of the depth of the tree
:param spaces: Control the number of spaces used for indentation
:return: The name of each tag and the number of spaces between tags
:doc-author: Joel Brogan, BRIAR team, Trelent
```

6.50.1.3 parseBriarSigset()

```
def briar.sigset.parse.parseBriarSigset (
              filename )
```

```
The parseBriarSigset function parses a Briar XML sigset file and returns a pandas dataframe with the following
name - The name of the signature.
subjectId - The ID of the subject who signed this signature.
filepath - The path to where this sigmember is stored on disk. This may be relative or absolute, depending on
modality - What type of media is used for this sigmember (e.g., 'online', 'offline').  See https://briarsigset

:param filename: Specify the path to the xml file
:return: A dataframe with the following columns:
:doc-author: Joel Brogan, BRIAR team, Joel Brogan, BRIAR team, Trelent
```

6.50.2 Variable Documentation

6.50.2.1 args

```
args = sys.argv
```

6.51 briar.timing Namespace Reference

Functions

- def end_duration (reply)
- def generate_progress (frame_id, media)
- def loadDurationsFolder (durations_directory)
- def parseDurations (durationsperfile_dictionary)
- def print_duration (name, duration)
- def print_durations (durations)
- def save_durations (media_file, durations_list, options, operation, modality=None)
- def start_duration (request, reply)
- def timeElapsed (duration)
- def timestamp ()

Variables

- string DURATION_FILE_EXT = ".durations"

6.51.1 Function Documentation

6.51.1.1 end_duration()

```
def briar.timing.end_duration (
              reply )
```

6.51.1.2 generate_progress()

```
def briar.timing.generate_progress (
              frame_id,
              media )
```

6.51.1.3 loadDurationsFolder()

```
def briar.timing.loadDurationsFolder (
              durations_directory )
```

6.51.1.4 parseDurations()

```
def briar.timing.parseDurations (
              durationsperfile_dictionary )
```

6.51.1.5 print_duration()

```
def briar.timing.print_duration (
              name,
              duration )
```

6.51.1.6 print_durations()

```
def briar.timing.print_durations (
              durations )
```

6.51.1.7 save_durations()

```
def briar.timing.save_durations (
              media_file,
              durations_list,
              options,
              operation,
              modality = None )
```

6.51.1.8 start_duration()

```
def briar.timing.start_duration (
              request,
              reply )
```

6.51.1.9 timeElapsed()

```
def briar.timing.timeElapsed (
              duration )
```

6.51.1.10 timestamp()

```
def briar.timing.timestamp ( )
```

6.51.2 Variable Documentation

6.51.2.1 DURATION_FILE_EXT

```
string DURATION_FILE_EXT = ".durations"
```

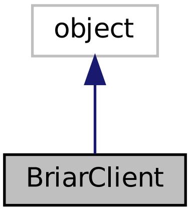
Chapter 7

Class Documentation

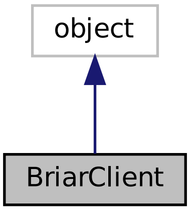
7.1 BriarClient Class Reference

Provide a client to a BRIAR service.

Inheritance diagram for BriarClient:



Collaboration diagram for BriarClient:



Public Member Functions

- def __init__ (self, options=None, reused_channel=None, reused_stub=None)

  Initialize the client and connect it to the specified server.
- def database_create (self, database_name)

  Creates an empty database of the given name.
- def database_insert (self, database_name, template_list, template_ids)

  Database functions: Insert.
- def database_list_templates (self, database_name)

  Lists the templates stored inside the given database name.
- def database_refresh (self)
- def database_remove_templates (self, database_name, template_ids)

  Remove templates matching the ids from the database.
- def database_retrieve (self, database_name, template_ids)

  Iteratively grab and return templates matching template_ids from the database.
- def detect (self, detect_requests, options=None)

  Detection functions.
- def enhance (self, enhance_requests, options=None)

  Enhancement Functions.
- def enroll (self, enroll_iter)

  Enroll images contained in the enroll iterator.
- def enroll_frames_iter (self, database_name, video, clientoptions=None, detect_options=None, extract_options=None, enroll_options=None, det_list_list=None, whole_image=False, request_start=-1, as_async=True, constructor=None)
- def enroll_frames_iter_async (self, database_name, video, detect_options=None, extract_options=None, enroll_options=None, det_list_list=None, whole_image=False, request_start=-1, yieldextra=False)
- def extract (self, extract_iter)

  Extract images contained in the extract iterator.
- def finalize (self, database_name)

  Write the given database to disk on the server on which it is running.
- def get_database_names (self)

  Database functions.
- def get_service_configuration (self)

  Utility Functions:
- def get_status (self, options=None)

  Service Functions.
- def iter_over_async (self, ait, loop)

  Enroll Functions.
- def load_database (self, database_name)

  Database functions: Load/Create.
- def print_verbose (self, ∗args)

  Simple helper function to print only when the verbose client is given the verbose flag.
- def retrieve_req_iter (self, database_name, template_ids)

  Database functions: Retrieve.
- def search (self, search_iter)

  Search Functions.
- def sync_enroll_frames_iter (self, database_name, video, detect_options=None, extract_options=None, enroll_options=None, det_list_list=None, whole_image=False, request_start=-1)
- def track (self, track_iter)

  Tracking Functions.
- def verify (self, flag, reference_media=None, verification_media=None, reference_dets=None, verification_dets=None, reference_tmpls=None, verification_tmpls=None)

Verify Functions.

- def verify_file_iter (self, reference_media_file, verify_media_file, client_options, detect_options=None, extract_options=None, verify_options=None, verify_templates=None, det_list_list=None, whole_↩ image=False, request_start=-1)

  Iterates the paths in the media file list, loading them one by one and yielding grpc verification requests.
- def verify_files (self, reference_file, verify_file, client_options, detect_options=None, extract_options=None, verify_options=None, verify_templates=None, det_list_list=None, whole_image=False, request_start=-1)

  Iterator which verifies images or videos from the given file paths, automatically creating and yielding enroll requests initialized from the read images.

Public Attributes

- channel
- options
- port
- stub

Static Public Attributes

- string DEFAULT_PORT = "0.0.0.0:50051"

7.1.1 Detailed Description

Provide a client to a BRIAR service.

It defines and sends the messages which are sent to the connected server

7.1.2 Constructor & Destructor Documentation

7.1.2.1 __init__()

```
def __init__ (
              self,
              options = None,
              reused_channel = None,
              reused_stub = None )
```

Initialize the client and connect it to the specified server.

Attempts a connection to localhost by default

Parameters

options	optparse.Values: Options which define the connection being established

7.1.3 Member Function Documentation

7.1.3.1 database_create()

```
def database_create (
              self,
              database_name )
```

Creates an empty database of the given name.

Parameters

database_name	str: Name of database to create

Returns

: briar_pb2.BriarDurations

7.1.3.2 database_insert()

```
def database_insert (
              self,
              database_name,
              template_list,
              template_ids )
```

Database functions: Insert.

Insert the given templates and ids into the database

TODO remove template_ids: is superfluous - template_list is all that is needed

TODO Insert should automatically generate template ids for templates with no ids and return new ids

Parameters

database_name	str: Name of the database to insert into
template_list	List of briar_pb2.Template
template_list	Templates to insert into database
template_ids	List of str
template_ids	IDs of templates being inserted

Returns

: 2 element Tuple (list of str, briar_pb2.BriarDurations

7.1.3.3 database_list_templates()

```
def database_list_templates (
              self,
              database_name )
```

Lists the templates stored inside the given database name.

Parameters

database_name	str: Name of the database to get the templates from

Returns

: 2 element tuple (list of str, briar_pb2.Durations)

7.1.3.4 database_refresh()

```
def database_refresh (
              self )
```

7.1.3.5 database_remove_templates()

```
def database_remove_templates (
              self,
              database_name,
              template_ids )
```

Remove templates matching the ids from the database.

Parameters

database_name	str: Name of the database to remove from
template_ids	list(str): Ids of the templates to remove

Returns

: briar_pb2.BriarDurations

7.1.3.6 database_retrieve()

```
def database_retrieve (
              self,
              database_name,
              template_ids )
```

Iteratively grab and return templates matching template_ids from the database.

Parameters

database_name	str: Name of the database to retrieve from
template_ids	list(str): List of ids to retrieve from the database

Returns

: 2 element Tuple (briar_pb2.Template, briar_pb2.BriarDurations

7.1.3.7 detect()

```
def detect (
              self,
              detect_requests,
              options = None )
```

Detection functions.

Run detection on media contained in detect_requests.

Parameters

detect_requests	Iterator yielding briar_service_pb2.DetectRequest: gRPC communication packet containing the data to run the detections on along with any additional options
options	optparse.Values
options	Additional options to feed to control the detect functions

yield: briar_service_pb2.DetectReply containing results

7.1.3.8 enhance()

```
def enhance (
              self,
              enhance_requests,
              options = None )
```

Enhancement Functions.

Run enhancement on media contained in enhance_requests.

Parameters

enhance_requests	Iterator yielding briar_service_pb2.EnhanceRequest: gRPC communication packet containing the data to run the detections on along with any additional options
options	optparse.Values
options	Additional options to feed to control the enhance functions

yield: briar_service_pb2.EnhanceReply containing results

7.1.3.9 enroll()

```
def enroll (
              self,
              enroll_iter )
```

Enroll images contained in the enroll iterator.

Parameters

enroll_iter	Generator: Generator object which yields enroll requests

Returns

: briar_service_pb2.ExtractReply

7.1.3.10 enroll_frames_iter()

```
def enroll_frames_iter (
              self,
              database_name,
              video,
              clientoptions = None,
              detect_options = None,
              extract_options = None,
              enroll_options = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1,
              as_async = True,
              constructor = None )
```

7.1.3.11 enroll_frames_iter_async()

```
def enroll_frames_iter_async (
              self,
              database_name,
              video,
              detect_options = None,
              extract_options = None,
              enroll_options = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1,
              yieldextra = False )
```

7.1.3.12 extract()

```
def extract (
              self,
              extract_iter )
```

Extract images contained in the extract iterator.

Parameters

extract_iter	Generator: Generator object which yields extract requests

Returns

: briar_service_pb2.ExtractReply

7.1.3.13 finalize()

```
def finalize (
              self,
              database_name )
```

Write the given database to disk on the server on which it is running.

Parameters

database_name	str: Name of the database to write to disk

Returns

: briar_pb2.durations

7.1.3.14 get_database_names()

```
def get_database_names (
              self )
```

Database functions.

Gets a list of names from the service representing the databases human readable names

Returns

: List of str

7.1.3.15 get_service_configuration()

```
def get_service_configuration (
              self )
```

Utility Functions:

7.1.3.16 get_status()

```
def get_status (
              self,
              options = None )
```

Service Functions.

Initialize the client and connect it to the specified server. Attempts a connection to localhost by default

Parameters

options	optparse.↩ Values:

Returns

: 5 element Tuple of str

7.1.3.17 iter_over_async()

```
def iter_over_async (
              self,
              ait,
              loop )
```

Enroll Functions.

7.1.3.18 load_database()

```
def load_database (
              self,
              database_name )
```

Database functions: Load/Create.

Load the database from disk into memory

Parameters

database_name	str: Name of database to load

Returns

: 3 element Tuple(list of str, list of records, briar_pb2.BriarDurations)

7.1.3.19 print_verbose()

```
def print_verbose (
              self,
             * args )
```

Simple helper function to print only when the verbose client is given the verbose flag.

Parameters

args	tuple(object): Arguments to get passed to print

Returns

: None - outputs to screen

7.1.3.20 retrieve_req_iter()

```
def retrieve_req_iter (
              self,
              database_name,
              template_ids )
```

Database functions: Retrieve.

Generator which yields retrieve requests

Parameters

database_name	str: Name of the database to retrieve from
template_ids	list(str): The templates to retrieve

Returns

: briar_service_pb2.DatabaseRetrieveRequest

7.1.3.21 search()

```
def search (
              self,
              search_iter )
```

Search Functions.

Given a probe, search the database and return matches

Parameters

database_name	str: Name of the database to search
media	briar_pb2.BriarMedia: Media to pull probes from
search_templates	briar_pb2.Template: Probe Template
detections	briar_pb2.Detection: Detections to extract probe templates from
flag	int, briar_pb2.SearchFlags: Tells search whether to use auto-detect, extract detections, or provided templates,

Returns

: briar_service_pb2.SearchReply

7.1.3.22 sync_enroll_frames_iter()

```
def sync_enroll_frames_iter (
              self,
              database_name,
              video,
              detect_options = None,
              extract_options = None,
              enroll_options = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1 )
```

7.1.3.23 track()

```
def track (
              self,
              track_iter )
```

Tracking Functions.

Track person instances contained in the tracking iterator

Parameters

extract_iter	Generator: Generator object which yields extract requests

Returns

: briar_service_pb2.ExtractReply

7.1.3.24 verify()

```
def verify (
              self,
              flag,
              reference_media = None,
              verification_media = None,
              reference_dets = None,
              verification_dets = None,
              reference_tmpls = None,
              verification_tmpls = None )
```

Verify Functions.

Either takes probe templates or generates them from provided media and compares them against the 'verification' variables of a matching type. I.e. templates<->templates, media<->media, etc.

Parameters

flag	int briar_pb2.VerifyFlags: Tells the service details about the media it is going to verify perform detections on the images then extract and verify, should it use existing dets, or should it use the provided templates to verify
reference_media	briar_pb2.BriarMedia: This media is where the the probe templates will be pulled from

Parameters

verification_media	briar_pb2.BriarMedia: This media is where the comparison templates will be pulled from
reference_dets	List of briar_pb2.Detection: These are the detections to extract probe templates from
verification_dets	List of briar_pb2.Detection: These are the detections to extract comparison templates from
reference_tmpls	list(briar_pb2.Template): Probe templates
verification_tmpls	list(briar_pb2.Template): Comparison Templates

Returns

: 2 element Tuple (briar_pb2.MatchSimilarities, briar_pb2.BriarDurations)

```
TODO Currently, this can only compare templates<->templates, detections<->detections, media<->media
TODO Ideally any combination should be able to be compared, i.e. template<->media, detection<->template, etc.
```

7.1.3.25 verify_file_iter()

```
def verify_file_iter (
              self,
              reference_media_file,
              verify_media_file,
              client_options,
              detect_options = None,
              extract_options = None,
              verify_options = None,
              verify_templates = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1 )
```

Iterates the paths in the media file list, loading them one by one and yielding grpc verification requests.

@type reference_media_file: File path as string

Parameters

reference_media_file	File path as string to the media file that acts as the reference media to be verify against

@type verify_media_file: File path as string

Parameters

verify_media_file	File path as string to the media file that requires verification, should be of same length as reference_media_files

@type detect_options: briar_pb2.DetectionOptions

Parameters

options	Command line options in protobuf format which control detection functionality

@type extract_optons: briar_pb2.ExtractOptions

Parameters

options	Command line options in protobuf format which control extraction functionality

@type earch_optons: briar_pb2.SearchOptions

Parameters

options	Command line options in protobuf format which control search functionality

@type det_list_list: List of list of briar_pb2.Detection

Parameters

det_list_list	If not None, it will contains 1 list of detections per media file

@type whole_image: boolean

Parameters

	Ignore detections and run an extract on the whole image

@yield: briar_service_pb2.EnrollRequest

7.1.3.26 verify_files()

```
def verify_files (
              self,
              reference_file,
              verify_file,
              client_options,
              detect_options = None,
              extract_options = None,
              verify_options = None,
              verify_templates = None,
              det_list_list = None,
              whole_image = False,
              request_start = -1 )
```

Iterator which verifies images or videos from the given file paths, automatically creating and yielding enroll requests initialized from the read images.

@type reference_media_files: File path string of reference media

Parameters

reference_media_files	File path string to the media that act as the reference media to be verify against

@type verify_media_files: File path string

Parameters

verify_media_files	File path string of the media file that requires verification

@type options: optparse.Values

Parameters

options	Command line options which control enrollment functionality

@type det_list_list: List of list of briar_pb2.Detection

Parameters

det_list_list	If not None, it will contains 1 list of detections per media file

@type whole_image: boolean

Parameters

	Ignore detections and run an extract on the whole image

yield: briar_service_pb2.VerifyReply containing results

7.1.4 Member Data Documentation

7.1.4.1 channel

```
channel
```

7.1.4.2 DEFAULT_PORT

```
string DEFAULT_PORT = "0.0.0.0:50051"  [static]
```

7.1.4.3 options

```
options
```

7.1.4.4 port

```
port
```

7.1.4.5 stub

```
stub
```

The documentation for this class was generated from the following file:

- briar_client.py

7.2 BriarMedia Class Reference

Public Member Functions

- def __init__ (self, media_input="", description="", datetime=None, metadata=None)

Public Attributes

- channels
- datetime
- description
- fps
- height
- len
- metadata
- source
- width

Static Public Attributes

- DATA_TYPES
- list IMAGE_FORMATS
- list VIDEO_FORMATS = ['.avi', '.mp4', '.mov', '.m4v', '.ts']

7.2.1 Constructor & Destructor Documentation

7.2.1.1 __init__()

```
def __init__ (
              self,
              media_input = "",
              description = "",
              datetime = None,
              metadata = None )
```

7.2.2 Member Data Documentation

7.2.2.1 channels

```
channels
```

7.2.2.2 DATA_TYPES

```
DATA_TYPES  [static]
```

Initial value:

```
=  dict(UINT8=0, UINT16=1, FLOAT32=2, URL=3, PNG=4, JPG=5, MJPG=6,
                        H264=7, H265=8)
```

7.2.2.3 datetime

```
datetime
```

7.2.2.4 description

```
description
```

7.2.2.5 fps

```
fps
```

7.2.2.6 height

```
height
```

7.2.2.7 IMAGE_FORMATS

```
list IMAGE_FORMATS  [static]
```

Initial value:

```
=  ['.bmp', 'dib', '.jpeg', '.jpg', '.jpe', '.jp2', '.png',
                      '.webp', '.pbm', '.pgm', '.ppm', '.pxm', '.pnm', '.sr',
                      '.ras', '.tiff', '.tif', '.exr', '.hdr', '.pic']
```

7.2.2.8 len

```
len
```

7.2.2.9 metadata

```
metadata
```

7.2.2.10 source

```
source
```

7.2.2.11 VIDEO_FORMATS

```
list VIDEO_FORMATS = ['.avi', '.mp4', '.mov', '.m4v', '.ts']  [static]
```

7.2.2.12 width

```
width
```

The documentation for this class was generated from the following file:

- briar_media.py

7.3 BriarProgress Class Reference

Public Member Functions

- def __init__ (self, options, desc=None, name=None, leave=True, position=None)
- def close (self)
- def refresh (self)
- def update (self, current=1, total=-1)

Public Attributes

- desc
- enabled
- leave
- name
- pbar
- position
- prevstep
- tqdm

7.3.1 Constructor & Destructor Documentation

7.3.1.1 __init__()

```
def __init__ (
              self,
              options,
              desc = None,
              name = None,
              leave = True,
              position = None )
```

```
The __init__ function is called when the class is instantiated.
It sets up the progress bar and initializes some variables.


:param self: Represent the instance of the class
:param options: Determine if progress bars are enabled
:param desc: Set the description of the progress bar
:param name: Name the progress bar
:param leave: Determine whether the progress bar should be left on screen after completion
:param position: Set the position of the progress bar
:return: Nothing
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.3.2 Member Function Documentation

7.3.2.1 close()

```
def close (
              self )
```

7.3.2.2 refresh()

```
def refresh (
              self )
```

7.3.2.3 update()

```
def update (
              self,
              current = 1,
              total = -1 )
```

7.3.3 Member Data Documentation

7.3.3.1 desc

```
desc
```

7.3.3.2 enabled

```
enabled
```

7.3.3.3 leave

```
leave
```

7.3.3.4 name

```
name
```

7.3.3.5 pbar

```
pbar
```

7.3.3.6 position

```
position
```

7.3.3.7 prevstep

```
prevstep
```

7.3.3.8 tqdm

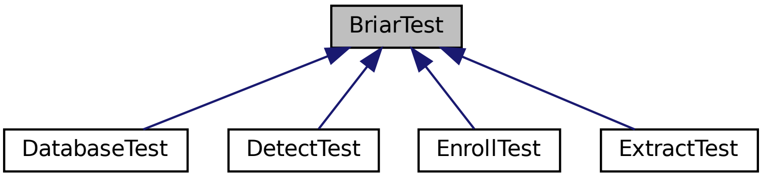
```
tqdm
```

The documentation for this class was generated from the following file:

- media/__init__.py

7.4 BriarTest Class Reference

Inheritance diagram for BriarTest:



Public Member Functions

- def __init__ (self)
- def description (self)
- def run (self)
- def test (self)

7.4.1 Constructor & Destructor Documentation

7.4.1.1 __init__()

```
def __init__ (
              self )
```

7.4.2 Member Function Documentation

7.4.2.1 description()

```
def description (
              self )
```

Reimplemented in ExtractTest, and DetectTest.

7.4.2.2 run()

```
def run (
              self )
```

7.4.2.3 test()

```
def test (
              self )
```

Reimplemented in DatabaseTest, and EnrollTest.

The documentation for this class was generated from the following file:

- cli/test.py

7.5 BriarTestResult Class Reference

Public Member Functions

- def __init__ (self, name, passed, reason=None, level=0)

Public Attributes

- level
- name
- passed
- reason

7.5.1 Constructor & Destructor Documentation

7.5.1.1 __init__()

```
def __init__ (
              self,
              name,
              passed,
              reason = None,
              level = 0 )
```

7.5.2 Member Data Documentation

7.5.2.1 level

```
level
```

7.5.2.2 name

```
name
```

7.5.2.3 passed

```
passed
```

7.5.2.4 reason

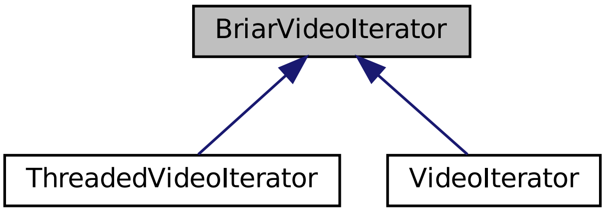
```
reason
```

The documentation for this class was generated from the following file:

- cli/test.py

7.6 BriarVideoIterator Class Reference

Inheritance diagram for BriarVideoIterator:



Public Member Functions

- def __aiter__ (self)
- def __anext__ (self)
- def __init__ (self, filepath, start=None, stop=None, unit=None, debug_empty=False, options=None)
- def __iter__ (self)
- def __len__ (self)
- def __next__ (self)

7.6.1 Constructor & Destructor Documentation

7.6.1.1 __init__()

```
def __init__ (
              self,
              filepath,
              start = None,
              stop = None,
              unit = None,
              debug_empty = False,
              options = None )
```

```
The __init__ function is called when the class is instantiated.
It sets up the instance of the class, and defines all its attributes.
The __init__ function takes in arguments that are passed to it by whoever creates an instance of this class,
and assigns these arguments to self variables so they can be used throughout this object.

:param self: Represent the instance of the class
:param filepath: Specify the path to the video file
:param start: Specify the start frame of the video
:param stop: Set the last frame to be read from the video
:param unit: Specify the unit of start and stop, choices: frame, time in seconds, NA (defaults to full video)
:param debug_empty: specified for creating a debug video iterator object that passes empty frames for testing
:return: Nothing
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented in ThreadedVideoIterator, and VideoIterator.

7.6.2 Member Function Documentation

7.6.2.1 __aiter__()

```
def __aiter__ (
              self )
```

```
The __aiter__ function is used to define an asynchronous iterator.

:param self: Refer to the current instance of a class
:return: The __iter__ function
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented in ThreadedVideoIterator, and VideoIterator.

7.6.2.2 __anext__()

```
def __anext__ (
              self )
```

```
The __anext__ function is the asynchronous iterator protocol.
It allows you to use async for loops, which are a lot more efficient than regular for loops.
The __anext__ function should return an awaitable object that resolves to the next item in your sequence.

:param self: Represent the instance of the class
:return: A frame, which is a numpy array
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented in ThreadedVideoIterator, and VideoIterator.

7.6.2.3 __iter__()

```
def __iter__ (
              self )
```

```
The __iter__ function is called when an iterator is required for a container.
This function should return a new iterator object that can iterate over all the objects in the container.
For mappings, it should iterate over the keys of the container, and should also be made available as the metho

:param self: Represent the instance of the class
:return: Self
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented in ThreadedVideoIterator, and VideoIterator.

7.6.2.4 __len__()

```
def __len__ (
              self )
```

```
The __len__ function is used to determine the length of an object.
For example, if you have a list with 5 items in it, calling len(my_list) will return 5.
The __len__ function is called when using the built-in len() function.

:param self: Allow an object to refer to itself
:return: The length of the iterator
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented in ThreadedVideoIterator, and VideoIterator.

7.6.2.5 __next__()

```
def __next__ (
              self )
```

```
The __next__ function is called by the for loop to get each item from the iterator.
The __next__ function should raise a StopIteration exception when there are no more items in the container.


:param self: Represent the instance of the class
:return: A frame from the video
:doc-author: Joel Brogan, BRIAR team, Trelent
```

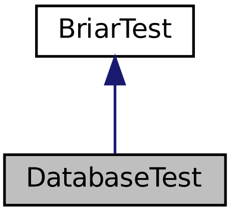
Reimplemented in ThreadedVideoIterator, and VideoIterator.

The documentation for this class was generated from the following file:

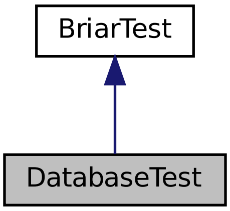
- media/__init__.py

7.7 DatabaseTest Class Reference

Inheritance diagram for DatabaseTest:



Collaboration diagram for DatabaseTest:



Public Member Functions

- def test (self)

7.7.1 Member Function Documentation

7.7.1.1 test()

```
def test (
              self )
```

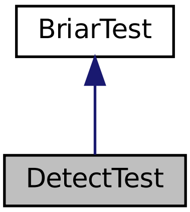
Reimplemented from BriarTest.

The documentation for this class was generated from the following file:

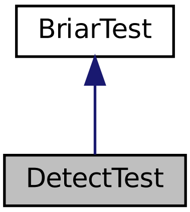
- cli/test.py

7.8 DetectTest Class Reference

Inheritance diagram for DetectTest:



Collaboration diagram for DetectTest:



Public Member Functions

- def description (self)
- def test_1_detection_image (self, testim_path=None, output_path=None, return_media=False)
- def test_2_detection_image_output (self, testim_path=None, output_path=None, return_media=False)
- def test_3_detection_image_withreturn (self)
- def test_4_detection_image_output_withreturn (self)

Public Attributes

- detection_file_path

Static Public Attributes

- string output_path = "./briar-integration-test-results"
- string testim_path = "testdata/BTS1/distractors/G00038/controlled/images_jpg/face/G00038_set2_face0_↩ 03_45_662fb70a.jpg"

7.8.1 Member Function Documentation

7.8.1.1 description()

```
def description (
              self )
```

Reimplemented from BriarTest.

7.8.1.2 test_1_detection_image()

```
def test_1_detection_image (
              self,
              testim_path = None,
              output_path = None,
              return_media = False )
```

7.8.1.3 test_2_detection_image_output()

```
def test_2_detection_image_output (
              self,
              testim_path = None,
              output_path = None,
              return_media = False )
```

7.8.1.4 test_3_detection_image_withreturn()

```
def test_3_detection_image_withreturn (
              self )
```

7.8.1.5 test_4_detection_image_output_withreturn()

```
def test_4_detection_image_output_withreturn (
              self )
```

7.8.2 Member Data Documentation

7.8.2.1 detection_file_path

```
detection_file_path
```

7.8.2.2 output_path

```
string output_path = "./briar-integration-test-results"  [static]
```

7.8.2.3 testim_path

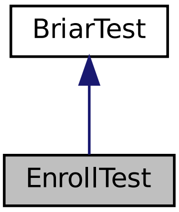
```
string testim_path = "testdata/BTS1/distractors/G00038/controlled/images_jpg/face/G00038_↩
set2_face0_03_45_662fb70a.jpg"  [static]
```

The documentation for this class was generated from the following file:

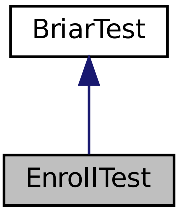
- cli/test.py

7.9 EnrollTest Class Reference

Inheritance diagram for EnrollTest:



Collaboration diagram for EnrollTest:



Public Member Functions

- def test (self)

7.9.1 Member Function Documentation

7.9.1.1 test()

```
def test (
              self )
```

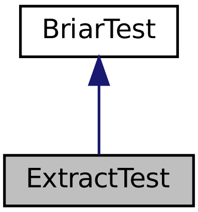
Reimplemented from BriarTest.

The documentation for this class was generated from the following file:

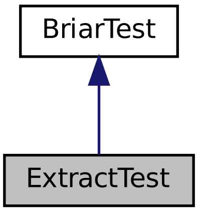
- cli/test.py

7.10 ExtractTest Class Reference

Inheritance diagram for ExtractTest:



Collaboration diagram for ExtractTest:



Public Member Functions

- def description (self)
- def test_1_extraction_image (self, testim_path=None, output_path=None, return_media=False)
- def test_2_extraction_image_output (self, testim_path=None, output_path=None, return_media=False)

Public Attributes

- detection_file_path
- template_file_path

Static Public Attributes

- string output_path = "./briar-integration-test-results"
- string testim_path = "testdata/BTS1/distractors/G00038/controlled/images_jpg/face/G00038_set2_face0_↩ 03_45_662fb70a.jpg"

7.10.1 Member Function Documentation

7.10.1.1 description()

```
def description (
              self )
```

Reimplemented from BriarTest.

7.10.1.2 test_1_extraction_image()

```
def test_1_extraction_image (
              self,
              testim_path = None,
              output_path = None,
              return_media = False )
```

7.10.1.3 test_2_extraction_image_output()

```
def test_2_extraction_image_output (
              self,
              testim_path = None,
              output_path = None,
              return_media = False )
```

7.10.2 Member Data Documentation

7.10.2.1 detection_file_path

```
detection_file_path
```

7.10.2.2 output_path

```
string output_path = "./briar-integration-test-results"  [static]
```

7.10.2.3 template_file_path

```
template_file_path
```

7.10.2.4 testim_path

```
string testim_path = "testdata/BTS1/distractors/G00038/controlled/images_jpg/face/G00038_↩
set2_face0_03_45_662fb70a.jpg"  [static]
```

The documentation for this class was generated from the following file:

- cli/test.py

7.11 FileVideoStream_cv2 Class Reference

Public Member Functions

- def __init__ (self, path, transform=None, queue_size=60 ∗3, options=None)
- def get_fps (self)
- def get_height (self)
- def get_length (self)
- def get_position (self)
- def get_width (self)
- def is_open (self)
- def more (self)
- def read (self)
- def running (self)
- def scrub_to (self, index)
- def start (self)
- def stop (self)
- def update (self)

Public Attributes

- options
- Q
- stopped
- stream
- thread
- transform

7.11.1 Constructor & Destructor Documentation

7.11.1.1 __init__()

```
def __init__ (
              self,
              path,
              transform = None,
              queue_size = 60*3,
              options = None )
```

7.11.2 Member Function Documentation

7.11.2.1 get_fps()

```
def get_fps (
              self )
```

7.11.2.2 get_height()

```
def get_height (
              self )
```

7.11.2.3 get_length()

```
def get_length (
              self )
```

7.11.2.4 get_position()

```
def get_position (
              self )
```

7.11.2.5 get_width()

```
def get_width (
              self )
```

7.11.2.6 is_open()

```
def is_open (
              self )
```

7.11.2.7 more()

```
def more (
              self )
```

7.11.2.8 read()

```
def read (
              self )
```

7.11.2.9 running()

```
def running (
              self )
```

7.11.2.10 scrub_to()

```
def scrub_to (
              self,
              index )
```

7.11.2.11 start()

```
def start (
              self )
```

7.11.2.12 stop()

```
def stop (
              self )
```

7.11.2.13 update()

```
def update (
              self )
```

7.11.3 Member Data Documentation

7.11.3.1 options

```
options
```

7.11.3.2 Q

```
Q
```

7.11.3.3 stopped

```
stopped
```

7.11.3.4 stream

```
stream
```

7.11.3.5 thread

```
thread
```

7.11.3.6 transform

```
transform
```

The documentation for this class was generated from the following file:

- media/VideoStream.py

7.12 FileVideoStream_imageio Class Reference

Public Member Functions

- def __init__ (self, path, transform=None, queue_size=60 *3, options=None)
- def get_fps (self)
- def get_height (self)
- def get_length (self)
- def get_position (self)
- def get_width (self)
- def is_open (self)
- def more (self)
- def read (self)
- def running (self)
- def scrub_to (self, index)
- def start (self)
- def stop (self)
- def update (self)

Public Attributes

- backend
- fps
- options
- Q
- stopped
- stream
- thread
- transform

7.12.1 Constructor & Destructor Documentation

7.12.1.1 __init__()

```
def __init__ (
              self,
              path,
              transform = None,
              queue_size = 60 * 3,
              options = None )
```

7.12.2 Member Function Documentation

7.12.2.1 get_fps()

```
def get_fps (
              self )
```

7.12.2.2 get_height()

```
def get_height (
              self )
```

7.12.2.3 get_length()

```
def get_length (
              self )
```

7.12.2.4 get_position()

```
def get_position (
              self )
```

7.12.2.5 get_width()

```
def get_width (
              self )
```

7.12.2.6 is_open()

```
def is_open (
              self )
```

7.12.2.7 more()

```
def more (
              self )
```

7.12.2.8 read()

```
def read (
              self )
```

7.12.2.9 running()

```
def running (
              self )
```

7.12.2.10 scrub_to()

```
def scrub_to (
              self,
              index )
```

7.12.2.11 start()

```
def start (
              self )
```

7.12.2.12 stop()

```
def stop (
              self )
```

7.12.2.13 update()

```
def update (
              self )
```

7.12.3 Member Data Documentation

7.12.3.1 backend

```
backend
```

7.12.3.2 fps

```
fps
```

7.12.3.3 options

```
options
```

7.12.3.4 Q

```
Q
```

7.12.3.5 stopped

```
stopped
```

7.12.3.6 stream

```
stream
```

7.12.3.7 thread

```
thread
```

7.12.3.8 transform

```
transform
```

The documentation for this class was generated from the following file:

- media/VideoStream.py

7.13 Fore Class Reference

Static Public Attributes

- string BLUE = ''
- string GREEN = ''
- string RED = ''
- string RESET = ''
- string YELLOW = ''

7.13.1 Member Data Documentation

7.13.1.1 BLUE

```
string BLUE = ''  [static]
```

7.13.1.2 GREEN

```
string GREEN = ''  [static]
```

7.13.1.3 RED

```
string RED = ''  [static]
```

7.13.1.4 RESET

```
string RESET = ''  [static]
```

7.13.1.5 YELLOW

```
string YELLOW = ''  [static]
```

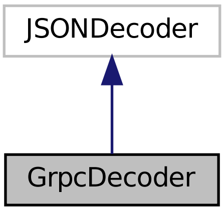
The documentation for this class was generated from the following file:

- cli/test.py

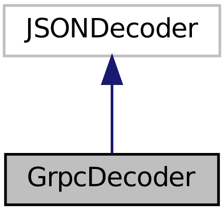
7.14 GrpcDecoder Class Reference

Object which extends the JSONDecoded to allow it to read saved gRPC files.

Inheritance diagram for GrpcDecoder:



Collaboration diagram for GrpcDecoder:



Public Member Functions

- def __init__ (self, options)
- def default (self, obj)

  Takes the given object and convert it into a gRPC object if its a dictionary.

Public Attributes

- options

7.14.1 Detailed Description

Object which extends the JSONDecoded to allow it to read saved gRPC files.

Applied as a hook in the json load function. Inherits from json.JSONDecoder

7.14.2 Constructor & Destructor Documentation

7.14.2.1 __init__()

```
def __init__ (
              self,
              options )
```

7.14.3 Member Function Documentation

7.14.3.1 default()

```
def default (
              self,
              obj )
```

Takes the given object and convert it into a gRPC object if its a dictionary.

Parameters

obj	object||dict: Dictionary which represents an object.

Returns

: object

7.14.4 Member Data Documentation

7.14.4.1 options

```
options
```

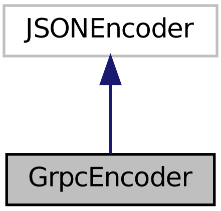
The documentation for this class was generated from the following file:

- grpc_json.py

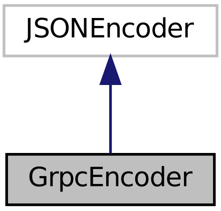
7.15 GrpcEncoder Class Reference

Encoder class which extends the normal JSON encoder to allow the encoding of gRPC objects.

Inheritance diagram for GrpcEncoder:



Collaboration diagram for GrpcEncoder:



Public Member Functions

- def __init__ (self, options=None)
- def default (self, obj)

  Json hook function to convert gRPC objects into a json-serializable object.

Public Attributes

- options

7.15.1 Detailed Description

Encoder class which extends the normal JSON encoder to allow the encoding of gRPC objects.

Inherits from json.JSONEncoder

7.15.2 Constructor & Destructor Documentation

7.15.2.1 __init__()

```
def __init__ (
              self,
              options = None )
```

7.15.3 Member Function Documentation

7.15.3.1 default()

```
def default (
              self,
              obj )
```

Json hook function to convert gRPC objects into a json-serializable object.

Parameters

obj	Any: General object to convert to a json-serializable object

Returns

: Json-Serializable object

7.15.4 Member Data Documentation

7.15.4.1 options

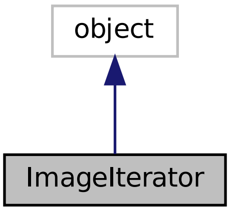
```
options
```

The documentation for this class was generated from the following file:

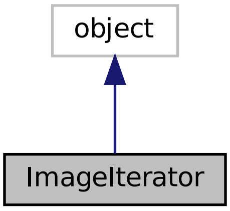
- grpc_json.py

7.16 ImageIterator Class Reference

Inheritance diagram for ImageIterator:



Collaboration diagram for ImageIterator:



Public Member Functions

- def __init__ (self, filepath, start=None, stop=None, unit=None, debug_empty=False, options=None)
- def __iter__ (self)
- def __len__ (self)
- def __next__ (self)

Public Attributes

- debug_empty
- filepath
- fps
- frame
- frame_count
- frame_height
- frame_width
- i
- isOpened
- length
- msec
- pos
- processed
- start_frame
- stop_frame

7.16.1 Constructor & Destructor Documentation

7.16.1.1 __init__()

```
def __init__ (
              self,
              filepath,
              start = None,
              stop = None,
              unit = None,
              debug_empty = False,
              options = None )
```

```
The __init__ function is called when the class is instantiated.
It sets up the object with all of its initial values.


:param self: Represent the instance of the class
:param filepath: Specify the path to the image file
:param start: Set the start frame of the video
:param stop: Specify the last frame to be read
:param unit: Specify the unit of time for the start and stop parameters
:return: The following:
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.16.2 Member Function Documentation

7.16.2.1 __iter__()

```
def __iter__ (
              self )
```

```
The __iter__ function is called when an iterator object is created for the class.
This function should return a new iterator object that can iterate over all the objects in the class.
For example, list, tuple or string classes have this method defined that allows them to be iterated over with

:param self: Access the attributes and methods of the class
:return: The object itself
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.16.2.2 __len__()

```
def __len__ (
              self )
```

```
The __len__ function is used to determine the length of an object.
For example, len(s) returns the number of items in s.
The built-in function len() calls s.__len__().


:param self: Refer to the instance of the class
:return: The length of the list
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.16.2.3 __next__()

```
def __next__ (
              self )
```

```
The __next__ function is called by the Python interpreter to fetch the next value from an iterator.
It should raise StopIteration when there are no more values to fetch.

:param self: Represent the instance of the class
:return: The next frame in the video
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.16.3 Member Data Documentation

7.16.3.1 debug_empty

```
debug_empty
```

7.16.3.2 filepath

```
filepath
```

7.16.3.3 fps

```
fps
```

7.16.3.4 frame

```
frame
```

7.16.3.5 frame_count

```
frame_count
```

7.16.3.6 frame_height

```
frame_height
```

7.16.3.7 frame_width

```
frame_width
```

7.16.3.8 i

```
i
```

7.16.3.9 isOpened

```
isOpened
```

7.16.3.10 length

```
length
```

7.16.3.11 msec

```
msec
```

7.16.3.12 pos

```
pos
```

7.16.3.13 processed

```
processed
```

7.16.3.14 start_frame

```
start_frame
```

7.16.3.15 stop_frame

```
stop_frame
```

The documentation for this class was generated from the following file:

- media/__init__.py

7.17 match_matrix_visualizer Class Reference

Public Member Functions

- def __init__ (self, searchReply, probedbname, gallerydbname)
- def showmat_interactive (self)

Public Attributes

- annotations
- ax
- fig
- figures
- gallerydb_name
- gt
- mat
- prevx
- prevy
- probedb_name
- searchReply
- xlabs
- xsources
- ylabs
- ysources

7.17.1 Constructor & Destructor Documentation

7.17.1.1 __init__()

```
def __init__ (
              self,
              searchReply,
              probedbname,
              gallerydbname )
```

7.17.2 Member Function Documentation

7.17.2.1 showmat_interactive()

```
def showmat_interactive (
              self )
```

7.17.3 Member Data Documentation

7.17.3.1 annotations

```
annotations
```

7.17.3.2 ax

```
ax
```

7.17.3.3 fig

```
fig
```

7.17.3.4 figures

```
figures
```

7.17.3.5 gallerydb_name

```
gallerydb_name
```

7.17.3.6 gt

```
gt
```

7.17.3.7 mat

```
mat
```

7.17.3.8 prevx

```
prevx
```

7.17.3.9 prevy

```
prevy
```

7.17.3.10 probedb_name

```
probedb_name
```

7.17.3.11 searchReply

```
searchReply
```

7.17.3.12 xlabs

```
xlabs
```

7.17.3.13 xsources

```
xsources
```

7.17.3.14 ylabs

```
ylabs
```

7.17.3.15 ysources

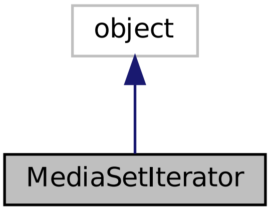
```
ysources
```

The documentation for this class was generated from the following file:

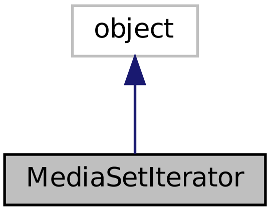
- media/visualize.py

7.18 MediaSetIterator Class Reference

Inheritance diagram for MediaSetIterator:



Collaboration diagram for MediaSetIterator:



Public Member Functions

- def __init__ (self, filepaths, start_frames, stop_frames, unit=None, path_map={}, options=None)
- def __iter__ (self)
- def __len__ (self)
- def __next__ (self)

Public Attributes

- filepaths
- i
- isOpened
- media_set
- processed
- start_frame
- start_frames
- stop_frame
- stop_frames

7.18.1 Constructor & Destructor Documentation

7.18.1.1 __init__()

```
def __init__ (
              self,
              filepaths,
              start_frames,
              stop_frames,
              unit = None,
              path_map = {},
              options = None )
```

```
The __init__ function is called when the class is instantiated.
It sets up the object with all of its initial values and does any other setup that needs to be done.


:param self: Refer to the object itself
:param filepaths: Store the filepaths of all the media files that are to be processed
:param start_frames: Specify the starting frame of each video
:param stop_frames: Set the last frame to be read from a video file
:param unit: Specify the unit of the start and stop frames
:param path_map: Map the filepaths to a new location
:return: Nothing
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.18.2 Member Function Documentation

7.18.2.1 __iter__()

```
def __iter__ (
              self )
```

```
The __iter__ function is called when an iterator object is created for the class.
This function should return an object (usually just self) that has a next() method defined.
The next() method should return the next value for the iterable, or raise StopIteration if there are no more v

:param self: Represent the instance of the class
:return: The object itself
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.18.2.2 __len__()

```
def __len__ (
              self )
```

```
The __len__ function is a special function that returns the length of an object.
In this case, it’s returning the number of media objects in the set.

:param self: Refer to the class itself
:return: The length of the media_set list
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.18.2.3 __next__()

```
def __next__ (
              self )
```

```
The __next__ function is called by the for loop to get the next item in
the iterable. It should raise StopIteration when there are no more items
to return.

:param self: Represent the instance of the class
:return: The next item in the sequence
:doc-author: Joel Brogan, BRIAR team, Trelent
```

7.18.3 Member Data Documentation

7.18.3.1 filepaths

```
filepaths
```

7.18.3.2 i

```
i
```

7.18.3.3 isOpened

```
isOpened
```

7.18.3.4 media_set

```
media_set
```

7.18.3.5 processed

```
processed
```

7.18.3.6 start_frame

```
start_frame
```

7.18.3.7 start_frames

```
start_frames
```

7.18.3.8 stop_frame

```
stop_frame
```

7.18.3.9 stop_frames

```
stop_frames
```

The documentation for this class was generated from the following file:

- media/__init__.py

7.19 MediaStream Class Reference

Public Member Functions

- def __init__ (self, briar_media)
- def __iter__ (self, request_type)

Private Attributes

- _media_list

7.19.1 Constructor & Destructor Documentation

7.19.1.1 __init__()

```
def __init__ (
              self,
              briar_media )
```

7.19.2 Member Function Documentation

7.19.2.1 __iter__()

```
def __iter__ (
              self,
              request_type )
```

7.19.3 Member Data Documentation

7.19.3.1 _media_list

```
_media_list  [private]
```

The documentation for this class was generated from the following file:

- briar_media.py

7.20 Rect Class Reference

Public Member Functions

- def __init__ (self, x, y, width, height)

Public Attributes

- height
- width
- x
- y

7.20.1 Detailed Description

```
Basic rectangle for storing ROIs without needing to mess with the gRPC BriarRect
```

7.20.2 Constructor & Destructor Documentation

7.20.2.1 __init__()

```
def __init__ (
              self,
              x,
              y,
              width,
              height )
```

```
The __init__ function is called when the class is instantiated.
It sets up the object with its initial state.


:param self: Represent the instance of the class
:param x: Set the x coordinate of the rectangle
:param y: Set the y coordinate of the rectangle
:param width: Set the width of the rectangle
:param height: Set the height of the rectangle
:return: Nothing
:doc-author: Joel Brogan
```

7.20.3 Member Data Documentation

7.20.3.1 height

```
height
```

7.20.3.2 width

```
width
```

7.20.3.3 x

```
x
```

7.20.3.4 y

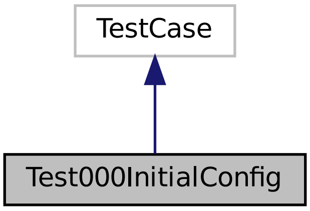
```
y
```

The documentation for this class was generated from the following file:

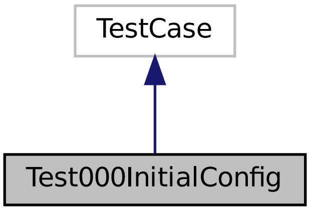
- __init__.py

7.21 Test000InitialConfig Class Reference

Inheritance diagram for Test000InitialConfig:



Collaboration diagram for Test000InitialConfig:



Public Member Functions

- None setUpClass (cls)
- def test_01_config_portlist (self)
- def test_02_port_connections (self)
- def test_03_num_service_ports (self)
- def test_04_num_procs_per_port (self)
- def test_05_num_threads_per_port (self)
- def test_06_correct_database_creation (self)
- def testDatasetDir (self)
- def testOutDir (self)
- def testValidationDir (self)

Public Attributes

- config_reply

7.21.1 Member Function Documentation

7.21.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.21.1.2 test_01_config_portlist()

```
def test_01_config_portlist (
              self )
```

```
Testing configuration port list
```

7.21.1.3 test_02_port_connections()

```
def test_02_port_connections (
              self )
```

```
Testing port connections
```

7.21.1.4 test_03_num_service_ports()

```
def test_03_num_service_ports (
              self )
```

```
Testing number of ports variable
```

7.21.1.5 test_04_num_procs_per_port()

```
def test_04_num_procs_per_port (
              self )
```

7.21.1.6 test_05_num_threads_per_port()

```
def test_05_num_threads_per_port (
              self )
```

7.21.1.7 test_06_correct_database_creation()

```
def test_06_correct_database_creation (
              self )
```

7.21.1.8 testDatasetDir()

```
def testDatasetDir (
              self )
```

7.21.1.9 testOutDir()

```
def testOutDir (
              self )
```

7.21.1.10 testValidationDir()

```
def testValidationDir (
              self )
```

7.21.2 Member Data Documentation

7.21.2.1 config_reply

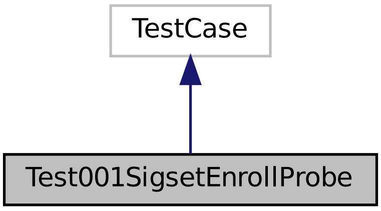
```
config_reply
```

The documentation for this class was generated from the following file:

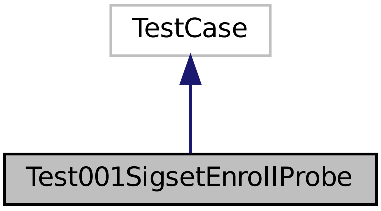
- evaluation/full_evaluation.py

7.22 Test001SigsetEnrollProbe Class Reference

Inheritance diagram for Test001SigsetEnrollProbe:



Collaboration diagram for Test001SigsetEnrollProbe:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_enroll_probe (self)
- def test_02_probe_checkpoint (self)

7.22.1 Member Function Documentation

7.22.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.22.1.2 test_01_sigset_enroll_probe()

```
def test_01_sigset_enroll_probe (
              self )
```

7.22.1.3 test_02_probe_checkpoint()

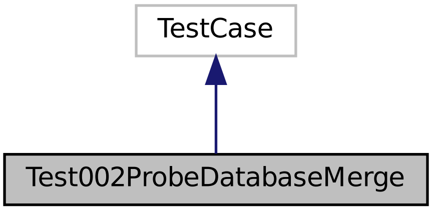
```
def test_02_probe_checkpoint (
              self )
```

The documentation for this class was generated from the following file:

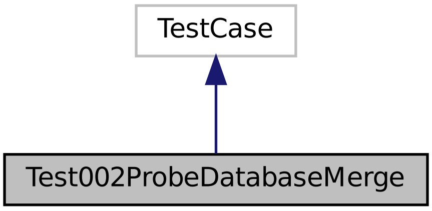
- evaluation/full_evaluation.py

7.23 Test002ProbeDatabaseMerge Class Reference

Inheritance diagram for Test002ProbeDatabaseMerge:



Collaboration diagram for Test002ProbeDatabaseMerge:



Public Member Functions

- None setUpClass (cls)
- def test_02_merge_probe_dbs (self)
- def test_03_checkpoint_merged_probe_db (self)

Public Attributes

- merged_dbinfo
- total_entries
- total_failed
- total_templates

7.23.1 Member Function Documentation

7.23.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.23.1.2 test_02_merge_probe_dbs()

```
def test_02_merge_probe_dbs (
              self )
```

7.23.1.3 test_03_checkpoint_merged_probe_db()

```
def test_03_checkpoint_merged_probe_db (
              self )
```

7.23.2 Member Data Documentation

7.23.2.1 merged_dbinfo

```
merged_dbinfo
```

7.23.2.2 total_entries

```
total_entries
```

7.23.2.3 total_failed

```
total_failed
```

7.23.2.4 total_templates

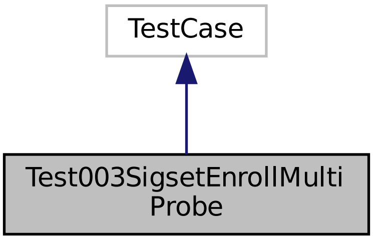
```
total_templates
```

The documentation for this class was generated from the following file:

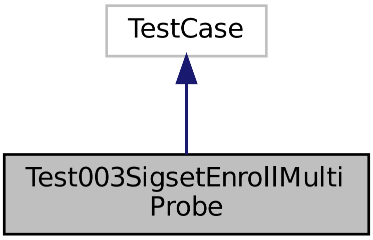
- evaluation/full_evaluation.py

7.24 Test003SigsetEnrollMultiProbe Class Reference

Inheritance diagram for Test003SigsetEnrollMultiProbe:



Collaboration diagram for Test003SigsetEnrollMultiProbe:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_enroll_probe (self)
- def test_02_probe_checkpoint (self)

7.24.1 Member Function Documentation

7.24.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.24.1.2 test_01_sigset_enroll_probe()

```
def test_01_sigset_enroll_probe (
              self )
```

7.24.1.3 test_02_probe_checkpoint()

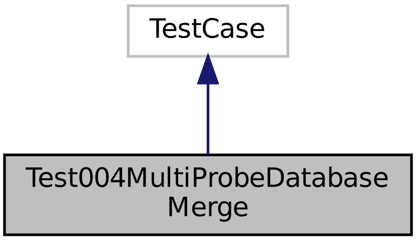
```
def test_02_probe_checkpoint (
              self )
```

The documentation for this class was generated from the following file:

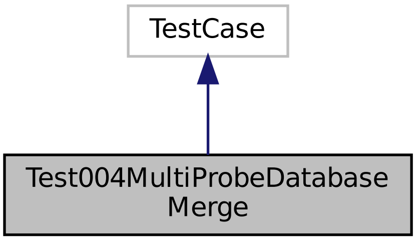
- evaluation/full_evaluation.py

7.25 Test004MultiProbeDatabaseMerge Class Reference

Inheritance diagram for Test004MultiProbeDatabaseMerge:



Collaboration diagram for Test004MultiProbeDatabaseMerge:



Public Member Functions

- None setUpClass (cls)
- def test_02_merge_probe_dbs (self)
- def test_03_checkpoint_merged_probe_db (self)

Public Attributes

- merged_dbinfo
- total_entries
- total_failed
- total_templates

7.25.1 Member Function Documentation

7.25.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.25.1.2 test_02_merge_probe_dbs()

```
def test_02_merge_probe_dbs (
              self )
```

7.25.1.3 test_03_checkpoint_merged_probe_db()

```
def test_03_checkpoint_merged_probe_db (
              self )
```

7.25.2 Member Data Documentation

7.25.2.1 merged_dbinfo

```
merged_dbinfo
```

7.25.2.2 total_entries

```
total_entries
```

7.25.2.3 total_failed

```
total_failed
```

7.25.2.4 total_templates

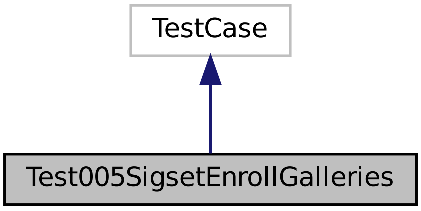
```
total_templates
```

The documentation for this class was generated from the following file:

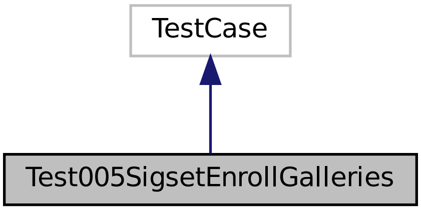
- evaluation/full_evaluation.py

7.26 Test005SigsetEnrollGalleries Class Reference

Inheritance diagram for Test005SigsetEnrollGalleries:



Collaboration diagram for Test005SigsetEnrollGalleries:



Public Member Functions

- def runGallery (self, gal_name, sigset_path)
- None setUpClass (cls)
- def test_01_sigset_enroll_gallery1 (self)
- def test_02_gallery1_partitioned_checkpoint (self)
- def test_03_sigset_enroll_gallery2 (self)
- def test_04_gallery2_partitioned_checkpoint (self)
- def test_05_sigset_enroll_gallery1 (self)
- def test_06_gallery1_partitioned_checkpoint (self)
- def test_07_sigset_enroll_gallery2 (self)
- def test_08_gallery2_partitioned_checkpoint (self)

7.26.1 Member Function Documentation

7.26.1.1 runGallery()

```
def runGallery (
              self,
              gal_name,
              sigset_path )
```

7.26.1.2 setUpClass()

```
None setUpClass (
              cls )
```

7.26.1.3 test_01_sigset_enroll_gallery1()

```
def test_01_sigset_enroll_gallery1 (
              self )
```

7.26.1.4 test_02_gallery1_partitioned_checkpoint()

```
def test_02_gallery1_partitioned_checkpoint (
              self )
```

7.26.1.5 test_03_sigset_enroll_gallery2()

```
def test_03_sigset_enroll_gallery2 (
              self )
```

7.26.1.6 test_04_gallery2_partitioned_checkpoint()

```
def test_04_gallery2_partitioned_checkpoint (
              self )
```

7.26.1.7 test_05_sigset_enroll_gallery1()

```
def test_05_sigset_enroll_gallery1 (
              self )
```

7.26.1.8 test_06_gallery1_partitioned_checkpoint()

```
def test_06_gallery1_partitioned_checkpoint (
              self )
```

7.26.1.9 test_07_sigset_enroll_gallery2()

```
def test_07_sigset_enroll_gallery2 (
              self )
```

7.26.1.10 test_08_gallery2_partitioned_checkpoint()

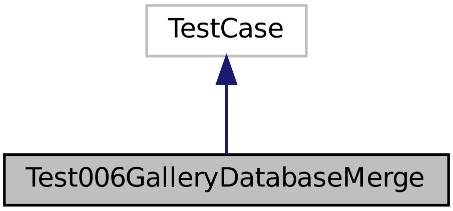
```
def test_08_gallery2_partitioned_checkpoint (
              self )
```

The documentation for this class was generated from the following file:

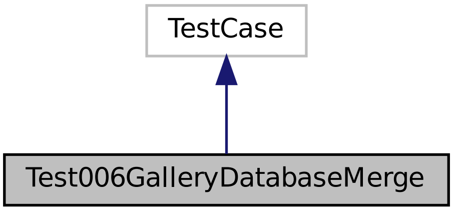
- evaluation/full_evaluation.py

7.27 Test006GalleryDatabaseMerge Class Reference

Inheritance diagram for Test006GalleryDatabaseMerge:



Collaboration diagram for Test006GalleryDatabaseMerge:



Public Member Functions

- def merge_db_func (self, gallery_sigset, gallery_db_base_name)
- None setUpClass (cls)
- def test_01_merge_gallery1_dbs (self)
- def test_02_merge_gallery2_dbs (self)
- def test_03_merge_blended_gallery2_dbs (self)
- def test_04_merge_blended_gallery2_dbs (self)
- def test_05_finalized_merged_gallery1_db (self)
- def test_06_finalized_merged_gallery1_db (self)
- def test_07_finalized_merged_gallery1_db (self)
- def test_08_finalized_merged_gallery1_db (self)

7.27.1 Member Function Documentation

7.27.1.1 merge_db_func()

```
def merge_db_func (
              self,
              gallery_sigset,
              gallery_db_base_name )
```

7.27.1.2 setUpClass()

```
None setUpClass (
              cls )
```

7.27.1.3 test_01_merge_gallery1_dbs()

```
def test_01_merge_gallery1_dbs (
              self )
```

7.27.1.4 test_02_merge_gallery2_dbs()

```
def test_02_merge_gallery2_dbs (
              self )
```

7.27.1.5 test_03_merge_blended_gallery2_dbs()

```
def test_03_merge_blended_gallery2_dbs (
              self )
```

7.27.1.6 test_04_merge_blended_gallery2_dbs()

```
def test_04_merge_blended_gallery2_dbs (
              self )
```

7.27.1.7 test_05_finalized_merged_gallery1_db()

```
def test_05_finalized_merged_gallery1_db (
              self )
```

7.27.1.8 test_06_finalized_merged_gallery1_db()

```
def test_06_finalized_merged_gallery1_db (
              self )
```

7.27.1.9 test_07_finalized_merged_gallery1_db()

```
def test_07_finalized_merged_gallery1_db (
              self )
```

7.27.1.10 test_08_finalized_merged_gallery1_db()

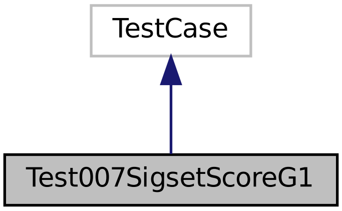
```
def test_08_finalized_merged_gallery1_db (
              self )
```

The documentation for this class was generated from the following file:

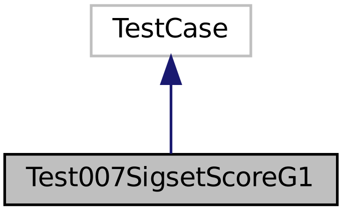
- evaluation/full_evaluation.py

7.28 Test007SigsetScoreG1 Class Reference

Inheritance diagram for Test007SigsetScoreG1:



Collaboration diagram for Test007SigsetScoreG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_probe_gallery1 (self)
- def test_02_sigset_search_probe_gallery1 (self)

7.28.1 Member Function Documentation

7.28.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.28.1.2 test_01_sigset_verify_probe_gallery1()

```
def test_01_sigset_verify_probe_gallery1 (
              self )
```

7.28.1.3 test_02_sigset_search_probe_gallery1()

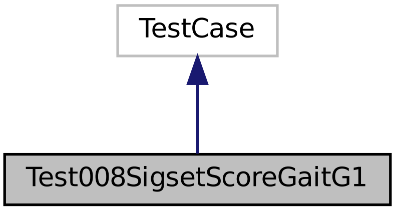
```
def test_02_sigset_search_probe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

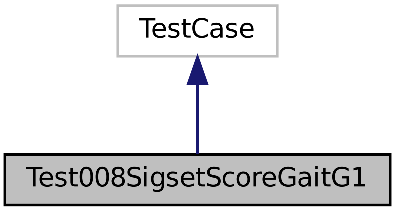
- evaluation/full_evaluation.py

7.29 Test008SigsetScoreGaitG1 Class Reference

Inheritance diagram for Test008SigsetScoreGaitG1:



Collaboration diagram for Test008SigsetScoreGaitG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_probe_gallery1 (self)
- def test_02_sigset_search_gait_probe_gallery1 (self)

7.29.1 Member Function Documentation

7.29.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.29.1.2 test_01_sigset_verify_gait_probe_gallery1()

```
def test_01_sigset_verify_gait_probe_gallery1 (
              self )
```

7.29.1.3 test_02_sigset_search_gait_probe_gallery1()

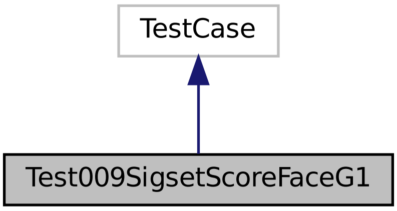
```
def test_02_sigset_search_gait_probe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

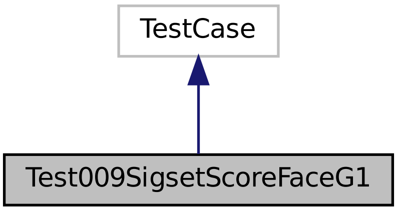
- evaluation/full_evaluation.py

7.30 Test009SigsetScoreFaceG1 Class Reference

Inheritance diagram for Test009SigsetScoreFaceG1:



Collaboration diagram for Test009SigsetScoreFaceG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_probe_gallery1 (self)
- def test_02_sigset_search_face_probe_gallery1 (self)

7.30.1 Member Function Documentation

7.30.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.30.1.2 test_01_sigset_verify_face_probe_gallery1()

```
def test_01_sigset_verify_face_probe_gallery1 (
              self )
```

7.30.1.3 test_02_sigset_search_face_probe_gallery1()

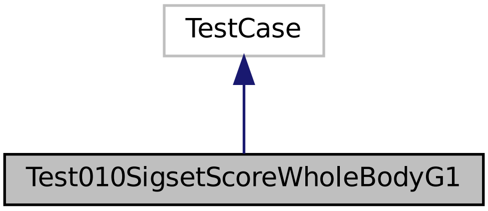
```
def test_02_sigset_search_face_probe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

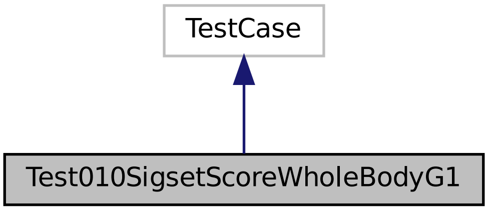
- evaluation/full_evaluation.py

7.31 Test010SigsetScoreWholeBodyG1 Class Reference

Inheritance diagram for Test010SigsetScoreWholeBodyG1:



Collaboration diagram for Test010SigsetScoreWholeBodyG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_probe_gallery1 (self)
- def test_02_sigset_search_WB_probe_gallery1 (self)

7.31.1 Member Function Documentation

7.31.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.31.1.2 test_01_sigset_verify_WB_probe_gallery1()

```
def test_01_sigset_verify_WB_probe_gallery1 (
              self )
```

7.31.1.3 test_02_sigset_search_WB_probe_gallery1()

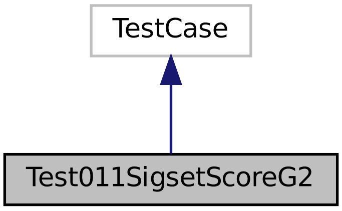
```
def test_02_sigset_search_WB_probe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

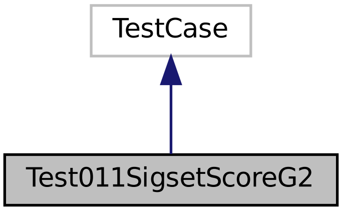
- evaluation/full_evaluation.py

7.32 Test011SigsetScoreG2 Class Reference

Inheritance diagram for Test011SigsetScoreG2:



Collaboration diagram for Test011SigsetScoreG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_probe_gallery2 (self)
- def test_02_sigset_search_probe_gallery2 (self)

7.32.1 Member Function Documentation

7.32.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.32.1.2 test_01_sigset_verify_probe_gallery2()

```
def test_01_sigset_verify_probe_gallery2 (
              self )
```

7.32.1.3 test_02_sigset_search_probe_gallery2()

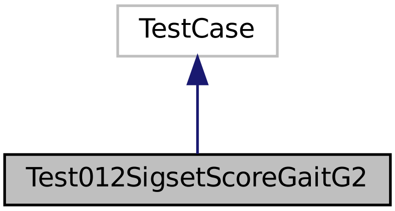
```
def test_02_sigset_search_probe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

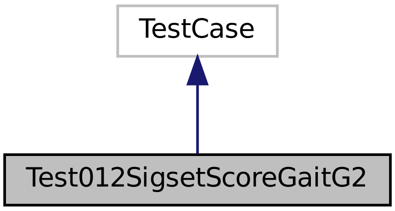
- evaluation/full_evaluation.py

7.33 Test012SigsetScoreGaitG2 Class Reference

Inheritance diagram for Test012SigsetScoreGaitG2:



Collaboration diagram for Test012SigsetScoreGaitG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_probe_gallery2 (self)
- def test_02_sigset_search_gait_probe_gallery2 (self)

7.33.1 Member Function Documentation

7.33.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.33.1.2 test_01_sigset_verify_gait_probe_gallery2()

```
def test_01_sigset_verify_gait_probe_gallery2 (
              self )
```

7.33.1.3 test_02_sigset_search_gait_probe_gallery2()

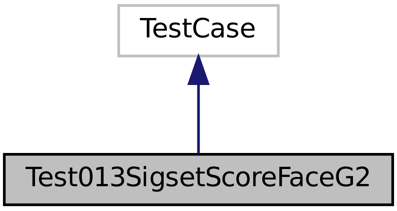
```
def test_02_sigset_search_gait_probe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

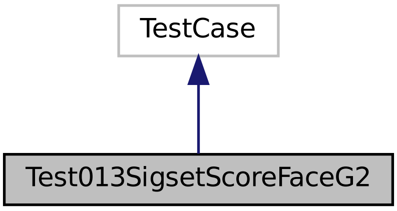
- evaluation/full_evaluation.py

7.34 Test013SigsetScoreFaceG2 Class Reference

Inheritance diagram for Test013SigsetScoreFaceG2:



Collaboration diagram for Test013SigsetScoreFaceG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_probe_gallery2 (self)
- def test_02_sigset_search_face_probe_gallery2 (self)

7.34.1 Member Function Documentation

7.34.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.34.1.2 test_01_sigset_verify_face_probe_gallery2()

```
def test_01_sigset_verify_face_probe_gallery2 (
              self )
```

7.34.1.3 test_02_sigset_search_face_probe_gallery2()

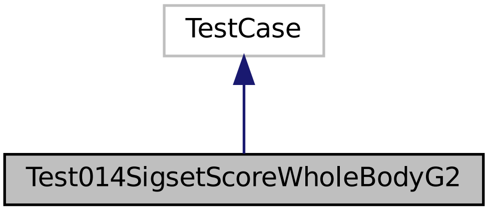
```
def test_02_sigset_search_face_probe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

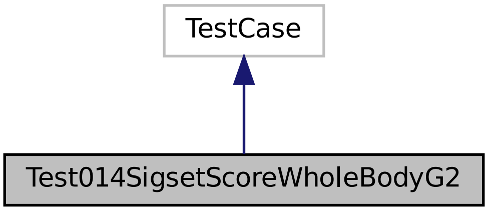
- evaluation/full_evaluation.py

7.35 Test014SigsetScoreWholeBodyG2 Class Reference

Inheritance diagram for Test014SigsetScoreWholeBodyG2:



Collaboration diagram for Test014SigsetScoreWholeBodyG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_probe_gallery2 (self)
- def test_02_sigset_search_WB_probe_gallery2 (self)

7.35.1 Member Function Documentation

7.35.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.35.1.2 test_01_sigset_verify_WB_probe_gallery2()

```
def test_01_sigset_verify_WB_probe_gallery2 (
              self )
```

7.35.1.3 test_02_sigset_search_WB_probe_gallery2()

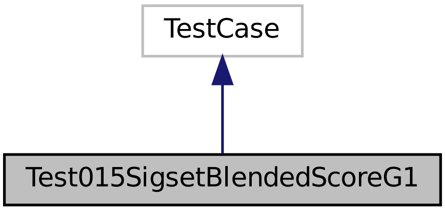
```
def test_02_sigset_search_WB_probe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

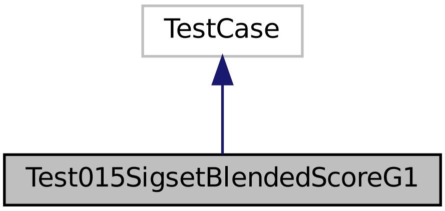
- evaluation/full_evaluation.py

7.36 Test015SigsetBlendedScoreG1 Class Reference

Inheritance diagram for Test015SigsetBlendedScoreG1:



Collaboration diagram for Test015SigsetBlendedScoreG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_probe_blended_gallery1 (self)
- def test_02_sigset_search_probe_blended_gallery1 (self)

7.36.1 Member Function Documentation

7.36.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.36.1.2 test_01_sigset_verify_probe_blended_gallery1()

```
def test_01_sigset_verify_probe_blended_gallery1 (
              self )
```

7.36.1.3 test_02_sigset_search_probe_blended_gallery1()

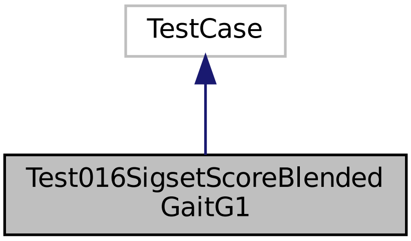
```
def test_02_sigset_search_probe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

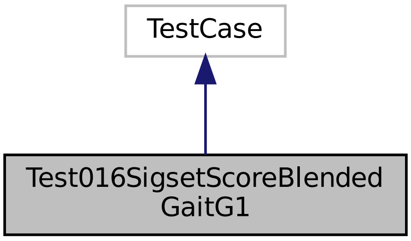
- evaluation/full_evaluation.py

7.37 Test016SigsetScoreBlendedGaitG1 Class Reference

Inheritance diagram for Test016SigsetScoreBlendedGaitG1:



Collaboration diagram for Test016SigsetScoreBlendedGaitG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_probe_blended_gallery1 (self)
- def test_02_sigset_search_gait_probe_blended_gallery1 (self)

7.37.1 Member Function Documentation

7.37.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.37.1.2 test_01_sigset_verify_gait_probe_blended_gallery1()

```
def test_01_sigset_verify_gait_probe_blended_gallery1 (
              self )
```

7.37.1.3 test_02_sigset_search_gait_probe_blended_gallery1()

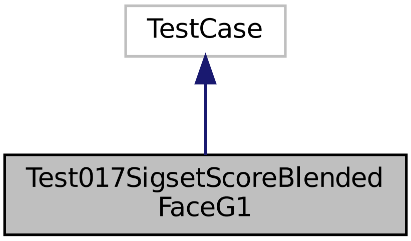
```
def test_02_sigset_search_gait_probe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

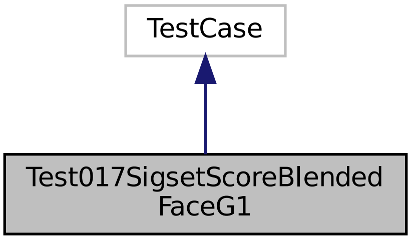
- evaluation/full_evaluation.py

7.38 Test017SigsetScoreBlendedFaceG1 Class Reference

Inheritance diagram for Test017SigsetScoreBlendedFaceG1:



Collaboration diagram for Test017SigsetScoreBlendedFaceG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_probe_blended_gallery1 (self)
- def test_02_sigset_search_face_probe_blended_gallery1 (self)

7.38.1 Member Function Documentation

7.38.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.38.1.2 test_01_sigset_verify_face_probe_blended_gallery1()

```
def test_01_sigset_verify_face_probe_blended_gallery1 (
              self )
```

7.38.1.3 test_02_sigset_search_face_probe_blended_gallery1()

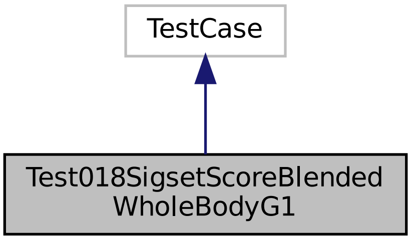
```
def test_02_sigset_search_face_probe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

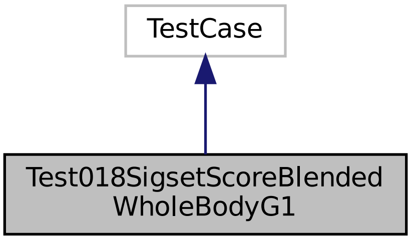
- evaluation/full_evaluation.py

7.39 Test018SigsetScoreBlendedWholeBodyG1 Class Reference

Inheritance diagram for Test018SigsetScoreBlendedWholeBodyG1:



Collaboration diagram for Test018SigsetScoreBlendedWholeBodyG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_probe_blended_gallery1 (self)
- def test_02_sigset_search_WB_probe_blended_gallery1 (self)

7.39.1 Member Function Documentation

7.39.1.1 setUpClass()

```
None setUpClass (
             cls )
```

7.39.1.2 test_01_sigset_verify_WB_probe_blended_gallery1()

```
def test_01_sigset_verify_WB_probe_blended_gallery1 (
              self )
```

7.39.1.3 test_02_sigset_search_WB_probe_blended_gallery1()

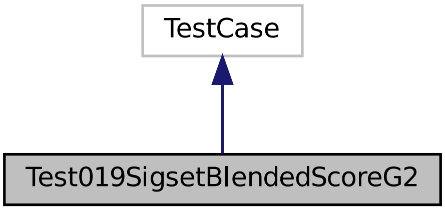
```
def test_02_sigset_search_WB_probe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

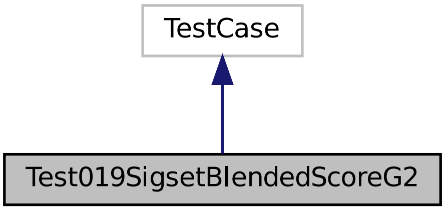
- evaluation/full_evaluation.py

7.40 Test019SigsetBlendedScoreG2 Class Reference

Inheritance diagram for Test019SigsetBlendedScoreG2:



Collaboration diagram for Test019SigsetBlendedScoreG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_probe_blended_gallery2 (self)
- def test_02_sigset_search_probe_blended_gallery2 (self)

7.40.1 Member Function Documentation

7.40.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.40.1.2 test_01_sigset_verify_probe_blended_gallery2()

```
def test_01_sigset_verify_probe_blended_gallery2 (
              self )
```

7.40.1.3 test_02_sigset_search_probe_blended_gallery2()

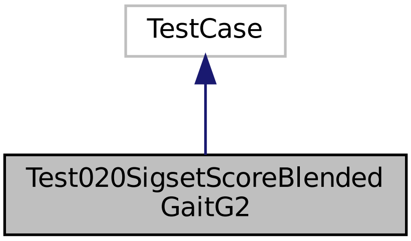
```
def test_02_sigset_search_probe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

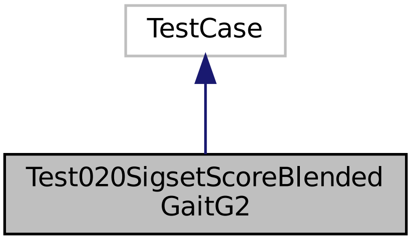
- evaluation/full_evaluation.py

7.41 Test020SigsetScoreBlendedGaitG2 Class Reference

Inheritance diagram for Test020SigsetScoreBlendedGaitG2:



Collaboration diagram for Test020SigsetScoreBlendedGaitG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_probe_blended_gallery2 (self)
- def test_02_sigset_search_gait_probe_blended_gallery2 (self)

7.41.1 Member Function Documentation

7.41.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.41.1.2 test_01_sigset_verify_gait_probe_blended_gallery2()

```
def test_01_sigset_verify_gait_probe_blended_gallery2 (
              self )
```

7.41.1.3 test_02_sigset_search_gait_probe_blended_gallery2()

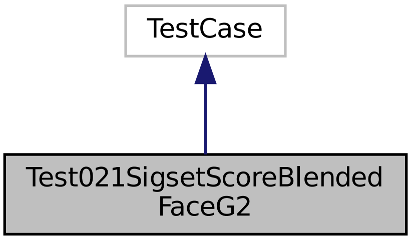
```
def test_02_sigset_search_gait_probe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

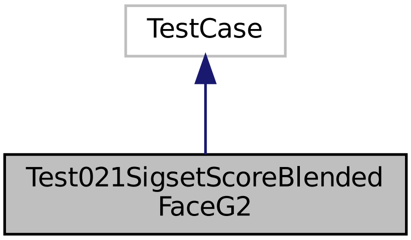
- evaluation/full_evaluation.py

7.42 Test021SigsetScoreBlendedFaceG2 Class Reference

Inheritance diagram for Test021SigsetScoreBlendedFaceG2:



Collaboration diagram for Test021SigsetScoreBlendedFaceG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_probe_blended_gallery2 (self)
- def test_02_sigset_search_face_probe_blended_gallery2 (self)

7.42.1 Member Function Documentation

7.42.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.42.1.2 test_01_sigset_verify_face_probe_blended_gallery2()

```
def test_01_sigset_verify_face_probe_blended_gallery2 (
              self )
```

7.42.1.3 test_02_sigset_search_face_probe_blended_gallery2()

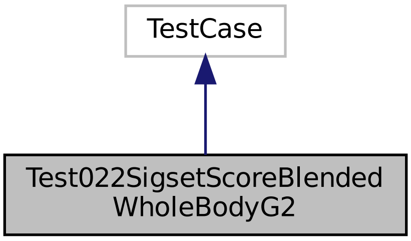
```
def test_02_sigset_search_face_probe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

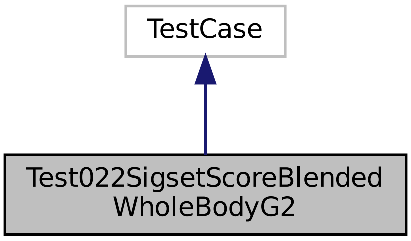
- evaluation/full_evaluation.py

7.43 Test022SigsetScoreBlendedWholeBodyG2 Class Reference

Inheritance diagram for Test022SigsetScoreBlendedWholeBodyG2:



Collaboration diagram for Test022SigsetScoreBlendedWholeBodyG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_probe_blended_gallery2 (self)
- def test_02_sigset_search_WB_probe_blended_gallery2 (self)

7.43.1 Member Function Documentation

7.43.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.43.1.2 test_01_sigset_verify_WB_probe_blended_gallery2()

```
def test_01_sigset_verify_WB_probe_blended_gallery2 (
              self )
```

7.43.1.3 test_02_sigset_search_WB_probe_blended_gallery2()

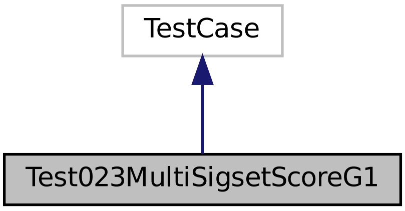
```
def test_02_sigset_search_WB_probe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

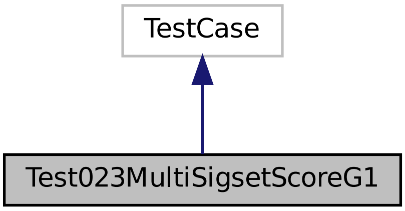
- evaluation/full_evaluation.py

7.44 Test023MultiSigsetScoreG1 Class Reference

Inheritance diagram for Test023MultiSigsetScoreG1:



Collaboration diagram for Test023MultiSigsetScoreG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_multiprobe_gallery1 (self)
- def test_02_sigset_search_multiprobe_gallery1 (self)

7.44.1 Member Function Documentation

7.44.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.44.1.2 test_01_sigset_verify_multiprobe_gallery1()

```
def test_01_sigset_verify_multiprobe_gallery1 (
              self )
```

7.44.1.3 test_02_sigset_search_multiprobe_gallery1()

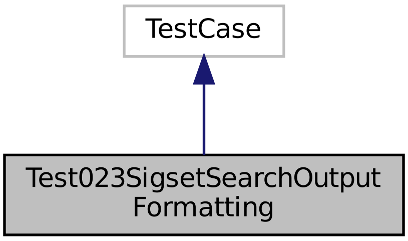
```
def test_02_sigset_search_multiprobe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

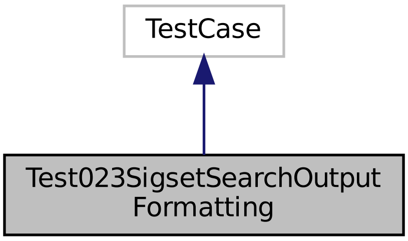
- evaluation/full_evaluation.py

7.45 Test023SigsetSearchOutputFormatting Class Reference

Inheritance diagram for Test023SigsetSearchOutputFormatting:



Collaboration diagram for Test023SigsetSearchOutputFormatting:



Public Member Functions

- def search_file_check (self, modality='', blended=False, galnumber=1)
- def test_01_sigset_search_pickle_fileG1 (self)
- def test_02_sigset_search_gait_pickle_fileG1 (self)
- def test_03_sigset_search_face_pickle_fileG1 (self)
- def test_04_sigset_search_wb_pickle_fileG1 (self)
- def test_05_sigset_search_pickle_fileG2 (self)
- def test_06_sigset_search_gait_pickle_fileG2 (self)
- def test_07_sigset_search_face_pickle_fileG2 (self)
- def test_08_sigset_search_wb_pickle_fileG2 (self)
- def test_09_sigset_blended_search_pickle_fileG1 (self)
- def test_10_sigset_blended_search_gait_pickle_fileG1 (self)
- def test_11_sigset_blended_search_face_pickle_fileG1 (self)
- def test_12_sigset_blended_search_wb_pickle_fileG1 (self)
- def test_13_sigset_blended_search_pickle_fileG2 (self)
- def test_14_sigset_blended_search_gait_pickle_fileG2 (self)
- def test_15_sigset_blended_search_face_pickle_fileG2 (self)
- def test_16_sigset_blended_search_wb_pickle_fileG2 (self)

7.45.1 Member Function Documentation

7.45.1.1 search_file_check()

```
def search_file_check (
              self,
              modality = '',
              blended = False,
              galnumber = 1 )
```

7.45.1.2 test_01_sigset_search_pickle_fileG1()

```
def test_01_sigset_search_pickle_fileG1 (
              self )
```

7.45.1.3 test_02_sigset_search_gait_pickle_fileG1()

```
def test_02_sigset_search_gait_pickle_fileG1 (
              self )
```

7.45.1.4 test_03_sigset_search_face_pickle_fileG1()

```
def test_03_sigset_search_face_pickle_fileG1 (
              self )
```

7.45.1.5 test_04_sigset_search_wb_pickle_fileG1()

```
def test_04_sigset_search_wb_pickle_fileG1 (
              self )
```

7.45.1.6 test_05_sigset_search_pickle_fileG2()

```
def test_05_sigset_search_pickle_fileG2 (
              self )
```

7.45.1.7 test_06_sigset_search_gait_pickle_fileG2()

```
def test_06_sigset_search_gait_pickle_fileG2 (
              self )
```

7.45.1.8 test_07_sigset_search_face_pickle_fileG2()

```
def test_07_sigset_search_face_pickle_fileG2 (
              self )
```

7.45.1.9 test_08_sigset_search_wb_pickle_fileG2()

```
def test_08_sigset_search_wb_pickle_fileG2 (
              self )
```

7.45.1.10 test_09_sigset_blended_search_pickle_fileG1()

```
def test_09_sigset_blended_search_pickle_fileG1 (
              self )
```

7.45.1.11 test_10_sigset_blended_search_gait_pickle_fileG1()

```
def test_10_sigset_blended_search_gait_pickle_fileG1 (
              self )
```

7.45.1.12 test_11_sigset_blended_search_face_pickle_fileG1()

```
def test_11_sigset_blended_search_face_pickle_fileG1 (
              self )
```

7.45.1.13 test_12_sigset_blended_search_wb_pickle_fileG1()

```
def test_12_sigset_blended_search_wb_pickle_fileG1 (
              self )
```

7.45.1.14 test_13_sigset_blended_search_pickle_fileG2()

```
def test_13_sigset_blended_search_pickle_fileG2 (
              self )
```

7.45.1.15 test_14_sigset_blended_search_gait_pickle_fileG2()

```
def test_14_sigset_blended_search_gait_pickle_fileG2 (
              self )
```

7.45.1.16 test_15_sigset_blended_search_face_pickle_fileG2()

```
def test_15_sigset_blended_search_face_pickle_fileG2 (
              self )
```

7.45.1.17 test_16_sigset_blended_search_wb_pickle_fileG2()

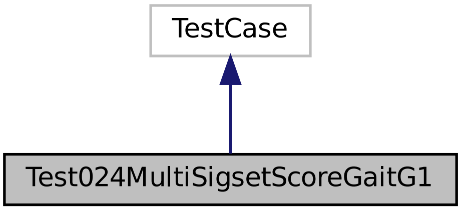
```
def test_16_sigset_blended_search_wb_pickle_fileG2 (
              self )
```

The documentation for this class was generated from the following file:

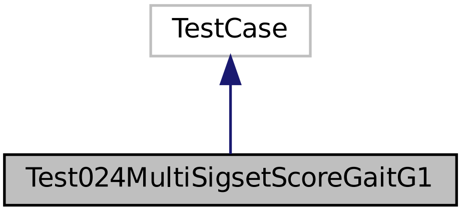
- evaluation/full_evaluation.py

7.46 Test024MultiSigsetScoreGaitG1 Class Reference

Inheritance diagram for Test024MultiSigsetScoreGaitG1:



Collaboration diagram for Test024MultiSigsetScoreGaitG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_multiprobe_gallery1 (self)
- def test_02_sigset_search_gait_multiprobe_gallery1 (self)

7.46.1 Member Function Documentation

7.46.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.46.1.2 test_01_sigset_verify_gait_multiprobe_gallery1()

```
def test_01_sigset_verify_gait_multiprobe_gallery1 (
              self )
```

7.46.1.3 test_02_sigset_search_gait_multiprobe_gallery1()

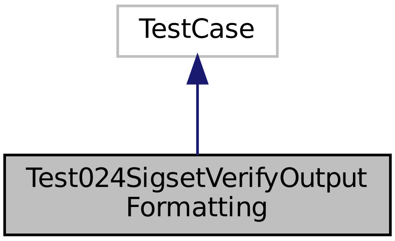
```
def test_02_sigset_search_gait_multiprobe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

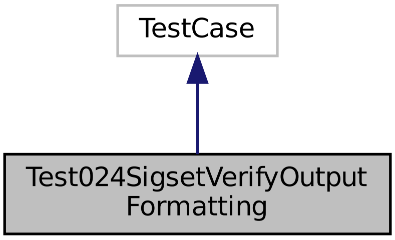
- evaluation/full_evaluation.py

7.47 Test024SigsetVerifyOutputFormatting Class Reference

Inheritance diagram for Test024SigsetVerifyOutputFormatting:



Collaboration diagram for Test024SigsetVerifyOutputFormatting:



Public Member Functions

- def score_file_check (self, modality='', blended=False, galnumber=1)
- def test_01_sigset_verify_pickle_fileG1 (self)
- def test_02_sigset_verify_gait_pickle_fileG1 (self)
- def test_03_sigset_verify_face_pickle_fileG1 (self)
- def test_04_sigset_verify_wb_pickle_fileG1 (self)
- def test_05_sigset_verify_pickle_fileG2 (self)
- def test_06_sigset_verify_gait_pickle_fileG2 (self)
- def test_07_sigset_verify_face_pickle_fileG2 (self)
- def test_08_sigset_verify_wb_pickle_fileG2 (self)
- def test_09_sigset_blended_verify_pickle_fileG1 (self)
- def test_10_sigset_blended_verify_gait_pickle_fileG1 (self)
- def test_11_sigset_blended_verify_face_pickle_fileG1 (self)
- def test_12_sigset_blended_verify_wb_pickle_fileG1 (self)
- def test_13_sigset_blended_verify_pickle_fileG2 (self)
- def test_14_sigset_blended_verify_gait_pickle_fileG2 (self)
- def test_15_sigset_blended_verify_face_pickle_fileG2 (self)
- def test_16_sigset_blended_verify_wb_pickle_fileG2 (self)

7.47.1 Member Function Documentation

7.47.1.1 score_file_check()

```
def score_file_check (
              self,
              modality = '',
              blended = False,
              galnumber = 1 )
```

7.47.1.2 test_01_sigset_verify_pickle_fileG1()

```
def test_01_sigset_verify_pickle_fileG1 (
              self )
```

7.47.1.3 test_02_sigset_verify_gait_pickle_fileG1()

```
def test_02_sigset_verify_gait_pickle_fileG1 (
              self )
```

7.47.1.4 test_03_sigset_verify_face_pickle_fileG1()

```
def test_03_sigset_verify_face_pickle_fileG1 (
              self )
```

7.47.1.5 test_04_sigset_verify_wb_pickle_fileG1()

```
def test_04_sigset_verify_wb_pickle_fileG1 (
              self )
```

7.47.1.6 test_05_sigset_verify_pickle_fileG2()

```
def test_05_sigset_verify_pickle_fileG2 (
              self )
```

7.47.1.7 test_06_sigset_verify_gait_pickle_fileG2()

```
def test_06_sigset_verify_gait_pickle_fileG2 (
              self )
```

7.47.1.8 test_07_sigset_verify_face_pickle_fileG2()

```
def test_07_sigset_verify_face_pickle_fileG2 (
              self )
```

7.47.1.9 test_08_sigset_verify_wb_pickle_fileG2()

```
def test_08_sigset_verify_wb_pickle_fileG2 (
              self )
```

7.47.1.10 test_09_sigset_blended_verify_pickle_fileG1()

```
def test_09_sigset_blended_verify_pickle_fileG1 (
              self )
```

7.47.1.11 test_10_sigset_blended_verify_gait_pickle_fileG1()

```
def test_10_sigset_blended_verify_gait_pickle_fileG1 (
              self )
```

7.47.1.12 test_11_sigset_blended_verify_face_pickle_fileG1()

```
def test_11_sigset_blended_verify_face_pickle_fileG1 (
              self )
```

7.47.1.13 test_12_sigset_blended_verify_wb_pickle_fileG1()

```
def test_12_sigset_blended_verify_wb_pickle_fileG1 (
              self )
```

7.47.1.14 test_13_sigset_blended_verify_pickle_fileG2()

```
def test_13_sigset_blended_verify_pickle_fileG2 (
              self )
```

7.47.1.15 test_14_sigset_blended_verify_gait_pickle_fileG2()

```
def test_14_sigset_blended_verify_gait_pickle_fileG2 (
              self )
```

7.47.1.16 test_15_sigset_blended_verify_face_pickle_fileG2()

```
def test_15_sigset_blended_verify_face_pickle_fileG2 (
              self )
```

7.47.1.17 test_16_sigset_blended_verify_wb_pickle_fileG2()

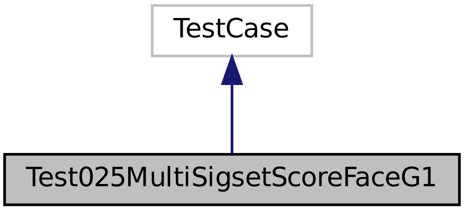
```
def test_16_sigset_blended_verify_wb_pickle_fileG2 (
              self )
```

The documentation for this class was generated from the following file:

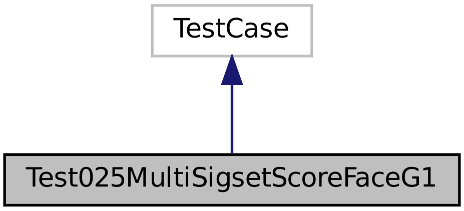
- evaluation/full_evaluation.py

7.48 Test025MultiSigsetScoreFaceG1 Class Reference

Inheritance diagram for Test025MultiSigsetScoreFaceG1:



Collaboration diagram for Test025MultiSigsetScoreFaceG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_multiprobe_gallery1 (self)
- def test_02_sigset_search_face_multiprobe_gallery1 (self)

7.48.1 Member Function Documentation

7.48.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.48.1.2 test_01_sigset_verify_face_multiprobe_gallery1()

```
def test_01_sigset_verify_face_multiprobe_gallery1 (
              self )
```

7.48.1.3 test_02_sigset_search_face_multiprobe_gallery1()

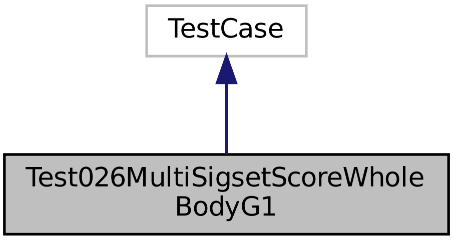
```
def test_02_sigset_search_face_multiprobe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

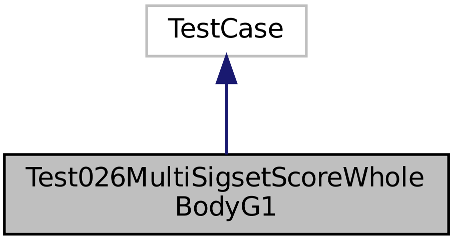
- evaluation/full_evaluation.py

7.49 Test026MultiSigsetScoreWholeBodyG1 Class Reference

Inheritance diagram for Test026MultiSigsetScoreWholeBodyG1:



Collaboration diagram for Test026MultiSigsetScoreWholeBodyG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_multiprobe_gallery1 (self)
- def test_02_sigset_search_WB_multiprobe_gallery1 (self)

7.49.1 Member Function Documentation

7.49.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.49.1.2 test_01_sigset_verify_WB_multiprobe_gallery1()

```
def test_01_sigset_verify_WB_multiprobe_gallery1 (
              self )
```

7.49.1.3 test_02_sigset_search_WB_multiprobe_gallery1()

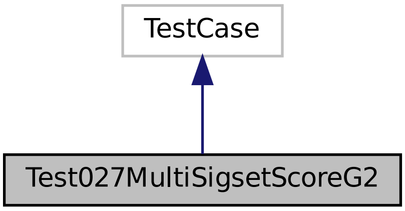
```
def test_02_sigset_search_WB_multiprobe_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

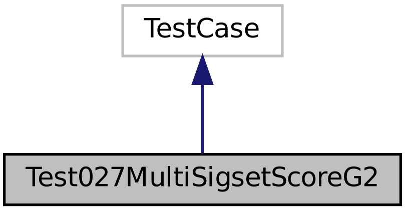
- evaluation/full_evaluation.py

7.50 Test027MultiSigsetScoreG2 Class Reference

Inheritance diagram for Test027MultiSigsetScoreG2:



Collaboration diagram for Test027MultiSigsetScoreG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_multiprobe_gallery2 (self)
- def test_02_sigset_search_multiprobe_gallery2 (self)

7.50.1 Member Function Documentation

7.50.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.50.1.2 test_01_sigset_verify_multiprobe_gallery2()

```
def test_01_sigset_verify_multiprobe_gallery2 (
              self )
```

7.50.1.3 test_02_sigset_search_multiprobe_gallery2()

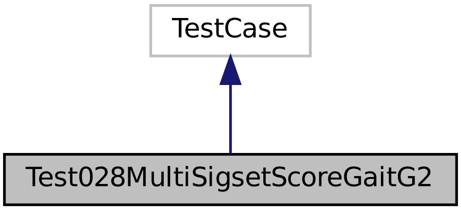
```
def test_02_sigset_search_multiprobe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

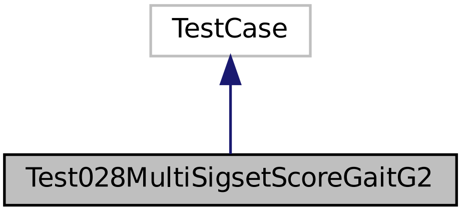
- evaluation/full_evaluation.py

7.51 Test028MultiSigsetScoreGaitG2 Class Reference

Inheritance diagram for Test028MultiSigsetScoreGaitG2:



Collaboration diagram for Test028MultiSigsetScoreGaitG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_multiprobe_gallery2 (self)
- def test_02_sigset_search_gait_multiprobe_gallery2 (self)

7.51.1 Member Function Documentation

7.51.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.51.1.2 test_01_sigset_verify_gait_multiprobe_gallery2()

```
def test_01_sigset_verify_gait_multiprobe_gallery2 (
              self )
```

7.51.1.3 test_02_sigset_search_gait_multiprobe_gallery2()

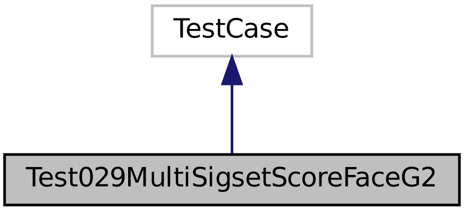
```
def test_02_sigset_search_gait_multiprobe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

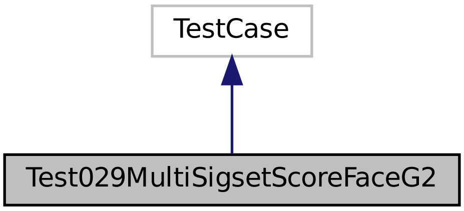
- evaluation/full_evaluation.py

7.52 Test029MultiSigsetScoreFaceG2 Class Reference

Inheritance diagram for Test029MultiSigsetScoreFaceG2:



Collaboration diagram for Test029MultiSigsetScoreFaceG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_multiprobe_gallery2 (self)
- def test_02_sigset_search_face_multiprobe_gallery2 (self)

7.52.1 Member Function Documentation

7.52.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.52.1.2 test_01_sigset_verify_face_multiprobe_gallery2()

```
def test_01_sigset_verify_face_multiprobe_gallery2 (
              self )
```

7.52.1.3 test_02_sigset_search_face_multiprobe_gallery2()

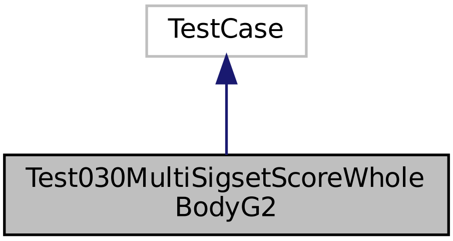
```
def test_02_sigset_search_face_multiprobe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

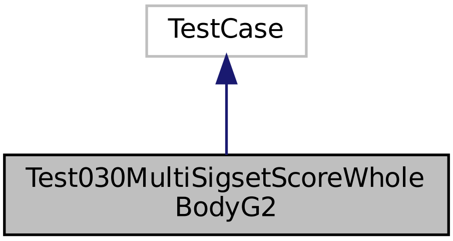
- evaluation/full_evaluation.py

7.53 Test030MultiSigsetScoreWholeBodyG2 Class Reference

Inheritance diagram for Test030MultiSigsetScoreWholeBodyG2:



Collaboration diagram for Test030MultiSigsetScoreWholeBodyG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_multiprobe_gallery2 (self)
- def test_02_sigset_search_WB_multiprobe_gallery2 (self)

7.53.1 Member Function Documentation

7.53.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.53.1.2 test_01_sigset_verify_WB_multiprobe_gallery2()

```
def test_01_sigset_verify_WB_multiprobe_gallery2 (
              self )
```

7.53.1.3 test_02_sigset_search_WB_multiprobe_gallery2()

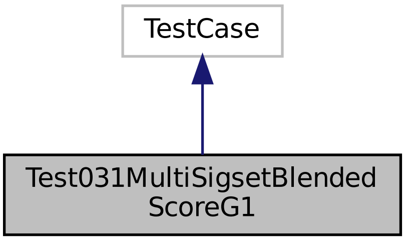
```
def test_02_sigset_search_WB_multiprobe_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

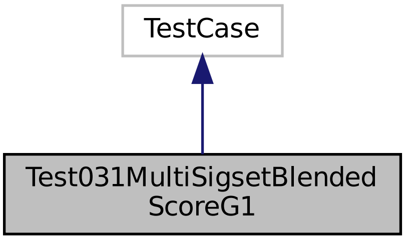
- evaluation/full_evaluation.py

7.54 Test031MultiSigsetBlendedScoreG1 Class Reference

Inheritance diagram for Test031MultiSigsetBlendedScoreG1:



Collaboration diagram for Test031MultiSigsetBlendedScoreG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_multiprobe_blended_gallery1 (self)
- def test_02_sigset_search_multiprobe_blended_gallery1 (self)

7.54.1 Member Function Documentation

7.54.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.54.1.2 test_01_sigset_verify_multiprobe_blended_gallery1()

```
def test_01_sigset_verify_multiprobe_blended_gallery1 (
              self )
```

7.54.1.3 test_02_sigset_search_multiprobe_blended_gallery1()

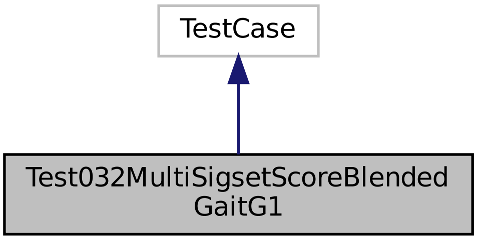
```
def test_02_sigset_search_multiprobe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

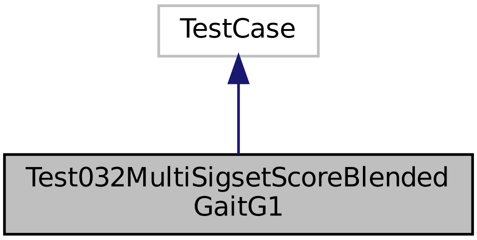
- evaluation/full_evaluation.py

7.55 Test032MultiSigsetScoreBlendedGaitG1 Class Reference

Inheritance diagram for Test032MultiSigsetScoreBlendedGaitG1:



Collaboration diagram for Test032MultiSigsetScoreBlendedGaitG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_multiprobe_blended_gallery1 (self)
- def test_02_sigset_search_gait_multiprobe_blended_gallery1 (self)

7.55.1 Member Function Documentation

7.55.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.55.1.2 test_01_sigset_verify_gait_multiprobe_blended_gallery1()

```
def test_01_sigset_verify_gait_multiprobe_blended_gallery1 (
              self )
```

7.55.1.3 test_02_sigset_search_gait_multiprobe_blended_gallery1()

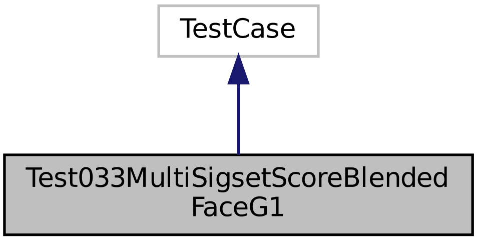
```
def test_02_sigset_search_gait_multiprobe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

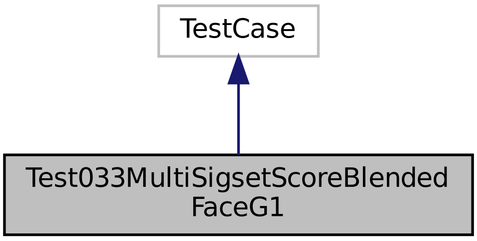
- evaluation/full_evaluation.py

7.56 Test033MultiSigsetScoreBlendedFaceG1 Class Reference

Inheritance diagram for Test033MultiSigsetScoreBlendedFaceG1:



Collaboration diagram for Test033MultiSigsetScoreBlendedFaceG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_multiprobe_blended_gallery1 (self)
- def test_02_sigset_search_face_multiprobe_blended_gallery1 (self)

7.56.1 Member Function Documentation

7.56.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.56.1.2 test_01_sigset_verify_face_multiprobe_blended_gallery1()

```
def test_01_sigset_verify_face_multiprobe_blended_gallery1 (
              self )
```

7.56.1.3 test_02_sigset_search_face_multiprobe_blended_gallery1()

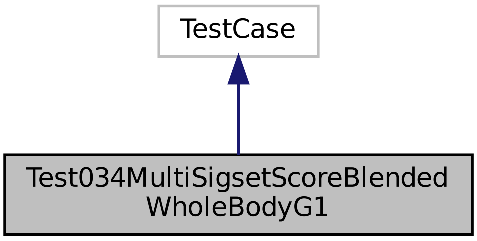
```
def test_02_sigset_search_face_multiprobe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

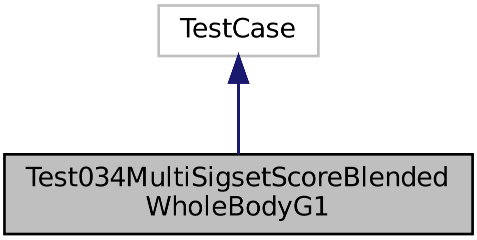
- evaluation/full_evaluation.py

7.57 Test034MultiSigsetScoreBlendedWholeBodyG1 Class Reference

Inheritance diagram for Test034MultiSigsetScoreBlendedWholeBodyG1:



Collaboration diagram for Test034MultiSigsetScoreBlendedWholeBodyG1:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_multiprobe_blended_gallery1 (self)
- def test_02_sigset_search_WB_multiprobe_blended_gallery1 (self)

7.57.1 Member Function Documentation

7.57.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.57.1.2 test_01_sigset_verify_WB_multiprobe_blended_gallery1()

```
def test_01_sigset_verify_WB_multiprobe_blended_gallery1 (
              self )
```

7.57.1.3 test_02_sigset_search_WB_multiprobe_blended_gallery1()

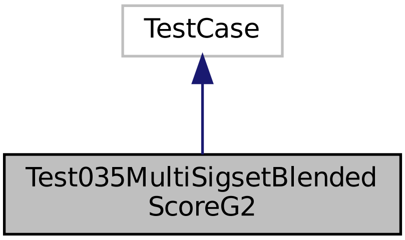
```
def test_02_sigset_search_WB_multiprobe_blended_gallery1 (
              self )
```

The documentation for this class was generated from the following file:

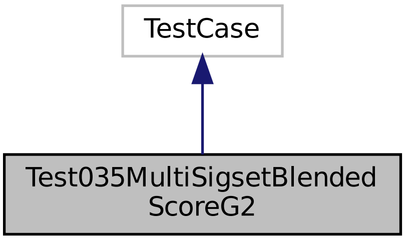
- evaluation/full_evaluation.py

7.58 Test035MultiSigsetBlendedScoreG2 Class Reference

Inheritance diagram for Test035MultiSigsetBlendedScoreG2:



Collaboration diagram for Test035MultiSigsetBlendedScoreG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_multiprobe_blended_gallery2 (self)
- def test_02_sigset_search_multiprobe_blended_gallery2 (self)

7.58.1 Member Function Documentation

7.58.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.58.1.2 test_01_sigset_verify_multiprobe_blended_gallery2()

```
def test_01_sigset_verify_multiprobe_blended_gallery2 (
              self )
```

7.58.1.3 test_02_sigset_search_multiprobe_blended_gallery2()

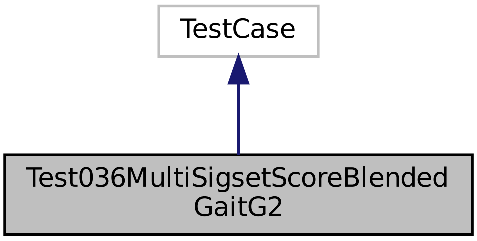
```
def test_02_sigset_search_multiprobe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

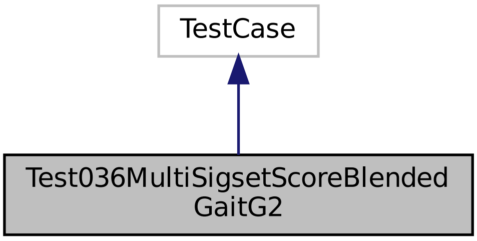
- evaluation/full_evaluation.py

7.59 Test036MultiSigsetScoreBlendedGaitG2 Class Reference

Inheritance diagram for Test036MultiSigsetScoreBlendedGaitG2:



Collaboration diagram for Test036MultiSigsetScoreBlendedGaitG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_gait_multiprobe_blended_gallery2 (self)
- def test_02_sigset_search_gait_multiprobe_blended_gallery2 (self)

7.59.1 Member Function Documentation

7.59.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.59.1.2 test_01_sigset_verify_gait_multiprobe_blended_gallery2()

```
def test_01_sigset_verify_gait_multiprobe_blended_gallery2 (
              self )
```

7.59.1.3 test_02_sigset_search_gait_multiprobe_blended_gallery2()

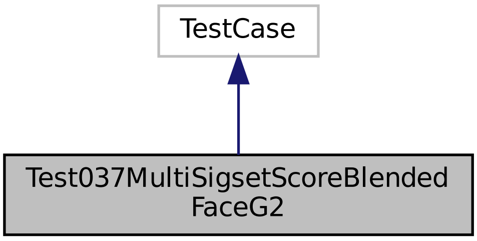
```
def test_02_sigset_search_gait_multiprobe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

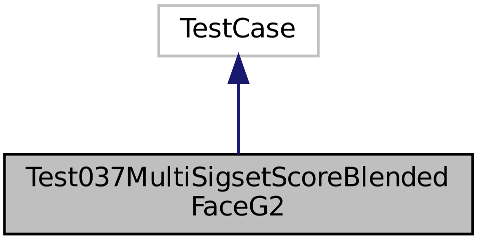
- evaluation/full_evaluation.py

7.60 Test037MultiSigsetScoreBlendedFaceG2 Class Reference

Inheritance diagram for Test037MultiSigsetScoreBlendedFaceG2:



Collaboration diagram for Test037MultiSigsetScoreBlendedFaceG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_face_multiprobe_blended_gallery2 (self)
- def test_02_sigset_search_face_multiprobe_blended_gallery2 (self)

7.60.1 Member Function Documentation

7.60.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.60.1.2 test_01_sigset_verify_face_multiprobe_blended_gallery2()

```
def test_01_sigset_verify_face_multiprobe_blended_gallery2 (
              self )
```

7.60.1.3 test_02_sigset_search_face_multiprobe_blended_gallery2()

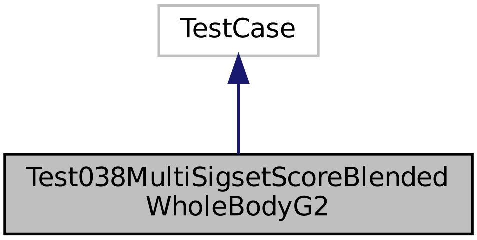
```
def test_02_sigset_search_face_multiprobe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

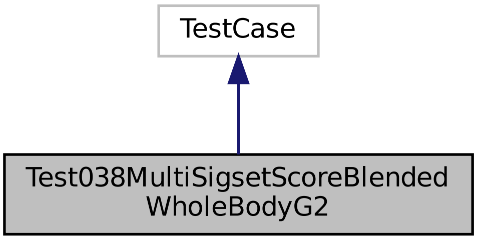
- evaluation/full_evaluation.py

7.61 Test038MultiSigsetScoreBlendedWholeBodyG2 Class Reference

Inheritance diagram for Test038MultiSigsetScoreBlendedWholeBodyG2:



Collaboration diagram for Test038MultiSigsetScoreBlendedWholeBodyG2:



Public Member Functions

- None setUpClass (cls)
- def test_01_sigset_verify_WB_multiprobe_blended_gallery2 (self)
- def test_02_sigset_search_WB_multiprobe_blended_gallery2 (self)

7.61.1 Member Function Documentation

7.61.1.1 setUpClass()

```
None setUpClass (
              cls )
```

7.61.1.2 test_01_sigset_verify_WB_multiprobe_blended_gallery2()

```
def test_01_sigset_verify_WB_multiprobe_blended_gallery2 (
              self )
```

7.61.1.3 test_02_sigset_search_WB_multiprobe_blended_gallery2()

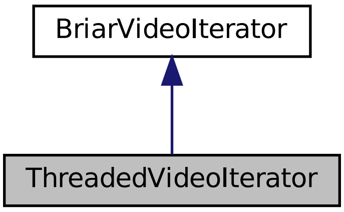
```
def test_02_sigset_search_WB_multiprobe_blended_gallery2 (
              self )
```

The documentation for this class was generated from the following file:

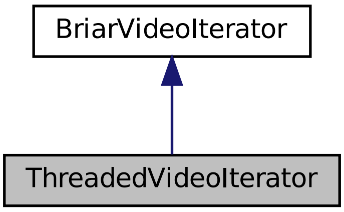
- evaluation/full_evaluation.py

7.62 ThreadedVideoIterator Class Reference

Inheritance diagram for ThreadedVideoIterator:



Collaboration diagram for ThreadedVideoIterator:



Public Member Functions

- def __aiter__ (self)
- def __anext__ (self)
- def __init__ (self, filepath, start=None, stop=None, unit=None, debug_empty=False, options=None)
- def __iter__ (self)
- def __len__ (self)
- def __next__ (self)
- def stop_iteration (self, exception_type)

Public Attributes

- cap
- debug_empty
- filepath
- fps
- frame_count
- frame_height
- frame_width
- i
- isOpened
- isStarted
- length
- msec
- options
- pos
- processed
- start_frame
- stop_frame
- stream

7.62.1 Constructor & Destructor Documentation

7.62.1.1 __init__()

```
def __init__ (
              self,
              filepath,
              start = None,
              stop = None,
              unit = None,
              debug_empty = False,
              options = None )
```

```
The __init__ function is called when the class is instantiated.
It sets up the instance of the class, and defines all its attributes.
The __init__ function takes in arguments that are passed to it by whoever creates an instance of this class,
and assigns these arguments to self variables so they can be used throughout this object.

:param self: Represent the instance of the class
:param filepath: Specify the path to the video file
:param start: Specify the start frame of the video
:param stop: Set the last frame to be read from the video
:param unit: Specify the unit of start and stop, choices: frame, time in seconds, NA (defaults to full video)
:param debug_empty: specified for creating a debug video iterator object that passes empty frames for testing
:return: Nothing
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.62.2 Member Function Documentation

7.62.2.1 __aiter__()

```
def __aiter__ (
              self )
```

```
The __aiter__ function is used to define an asynchronous iterator.

:param self: Refer to the current instance of a class
:return: The __iter__ function
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.62.2.2 __anext__()

```
def __anext__ (
              self )
```

```
The __anext__ function is the asynchronous iterator protocol.
It allows you to use async for loops, which are a lot more efficient than regular for loops.
The __anext__ function should return an awaitable object that resolves to the next item in your sequence.

:param self: Represent the instance of the class
:return: A frame, which is a numpy array
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.62.2.3 __iter__()

```
def __iter__ (
              self )
```

```
The __iter__ function is called when an iterator is required for a container.
This function should return a new iterator object that can iterate over all the objects in the container.
For mappings, it should iterate over the keys of the container, and should also be made available as the metho

:param self: Represent the instance of the class
:return: Self
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.62.2.4 __len__()

```
def __len__ (
              self )
```

```
The __len__ function is used to determine the length of an object.
For example, if you have a list with 5 items in it, calling len(my_list) will return 5.
The __len__ function is called when using the built-in len() function.

:param self: Allow an object to refer to itself
:return: The length of the iterator
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.62.2.5 __next__()

```
def __next__ (
              self )
```

```
The __next__ function is called by the for loop to get each item from the iterator.
The __next__ function should raise a StopIteration exception when there are no more items in the container.


:param self: Represent the instance of the class
:return: A frame from the video
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.62.2.6 stop_iteration()

```
def stop_iteration (
              self,
              exception_type )
```

7.62.3 Member Data Documentation

7.62.3.1 cap

```
cap
```

7.62.3.2 debug_empty

```
debug_empty
```

7.62.3.3 filepath

```
filepath
```

7.62.3.4 fps

```
fps
```

7.62.3.5 frame_count

```
frame_count
```

7.62.3.6 frame_height

```
frame_height
```

7.62.3.7 frame_width

```
frame_width
```

7.62.3.8 i

```
i
```

7.62.3.9 isOpened

```
isOpened
```

7.62.3.10 isStarted

```
isStarted
```

7.62.3.11 length

```
length
```

7.62.3.12 msec

```
msec
```

7.62.3.13 options

```
options
```

7.62.3.14 pos

```
pos
```

7.62.3.15 processed

```
processed
```

7.62.3.16 start_frame

```
start_frame
```

7.62.3.17 stop_frame

```
stop_frame
```

7.62.3.18 stream

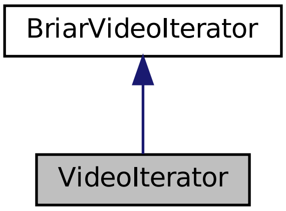
```
stream
```

The documentation for this class was generated from the following file:

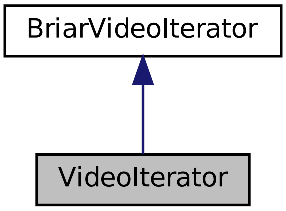
- media/__init__.py

7.63 VideoIterator Class Reference

Inheritance diagram for VideoIterator:



Collaboration diagram for VideoIterator:



Public Member Functions

- def __aiter__ (self)
- def __anext__ (self)
- def __init__ (self, filepath, start=None, stop=None, unit=None, debug_empty=False, options=None)
- def __iter__ (self)
- def __len__ (self)
- def __next__ (self)

Public Attributes

- cap
- debug_empty
- filepath
- fps
- frame_count
- frame_height
- frame_width
- i
- isOpened
- length
- msec
- pos
- processed
- start_frame
- stop_frame

7.63.1 Constructor & Destructor Documentation

7.63.1.1 __init__()

```
def __init__ (
              self,
              filepath,
              start = None,
              stop = None,
              unit = None,
              debug_empty = False,
              options = None )
```

```
The __init__ function is called when the class is instantiated.
It sets up the instance of the class, and defines all its attributes.
The __init__ function takes in arguments that are passed to it by whoever creates an instance of this class,
and assigns these arguments to self variables so they can be used throughout this object.

:param self: Represent the instance of the class
:param filepath: Specify the path to the video file
:param start: Specify the start frame of the video
:param stop: Set the last frame to be read from the video
:param unit: Specify the unit of start and stop, choices: frame, time in seconds, NA (defaults to full video)
:param debug_empty: specified for creating a debug video iterator object that passes empty frames for testing
:return: Nothing
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.63.2 Member Function Documentation

7.63.2.1 __aiter__()

```
def __aiter__ (
              self )
```

```
The __aiter__ function is used to define an asynchronous iterator.

:param self: Refer to the current instance of a class
:return: The __iter__ function
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.63.2.2 __anext__()

```
def __anext__ (
              self )
```

```
The __anext__ function is the asynchronous iterator protocol.
It allows you to use async for loops, which are a lot more efficient than regular for loops.
The __anext__ function should return an awaitable object that resolves to the next item in your sequence.

:param self: Represent the instance of the class
:return: A frame, which is a numpy array
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.63.2.3 __iter__()

```
def __iter__ (
              self )
```

```
The __iter__ function is called when an iterator is required for a container.
This function should return a new iterator object that can iterate over all the objects in the container.
For mappings, it should iterate over the keys of the container, and should also be made available as the metho

:param self: Represent the instance of the class
:return: Self
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.63.2.4 __len__()

```
def __len__ (
              self )
```

```
Calculates the length of the iterator.

Returns:
    The length of the iterator.

Raises:
    Exception: If there is an error in calculating the length.
```

Reimplemented from BriarVideoIterator.

7.63.2.5 __next__()

```
def __next__ (
              self )
```

```
The __next__ function is called by the for loop to get each item from the iterator.
The __next__ function should raise a StopIteration exception when there are no more items in the container.


:param self: Represent the instance of the class
:return: A frame from the video
:doc-author: Joel Brogan, BRIAR team, Trelent
```

Reimplemented from BriarVideoIterator.

7.63.3 Member Data Documentation

7.63.3.1 cap

```
cap
```

7.63.3.2 debug_empty

```
debug_empty
```

7.63.3.3 filepath

```
filepath
```

7.63.3.4 fps

```
fps
```

7.63.3.5 frame_count

```
frame_count
```

7.63.3.6 frame_height

```
frame_height
```

7.63.3.7 frame_width

```
frame_width
```

7.63.3.8 i

```
i
```

7.63.3.9 isOpened

```
isOpened
```

7.63.3.10 length

```
length
```

7.63.3.11 msec

```
msec
```

7.63.3.12 pos

```
pos
```

7.63.3.13 processed

```
processed
```

7.63.3.14 start_frame

```
start_frame
```

7.63.3.15 stop_frame

```
stop_frame
```

The documentation for this class was generated from the following file:

- media/__init__.py

Chapter 8

File Documentation

8.1 __init__.py File Reference

Classes

- class Rect

Namespaces

- briar

Functions

- def _wait_forever (server)
- def CLIServe (serviceClass, add_custom_options=None)
- def dyn_import (name)
- def get_process_number ()
- def get_thread_number ()
- def multiproc_serve (serviceClass, options=None, serve_port=None)
- def parse_ports (options)
- def serve (serviceClass, options=None, serve_port=None)

Variables

- string __version__ = '2.3.6'
- _ONE_DAY = datetime.timedelta(days=1)
- int DEFAULT_MAX_MESSAGE_SIZE = 64 ∗ 1024 ∗ 1024 ∗8
- string DEFAULT_PORT = "0.0.0.0:50051"
- string DEFAULT_SERVE_PORT = '[::]:50051'
- string PLATFORM = "UNKNOWN"

8.2 cli/__init__.py File Reference

Namespaces

- briar.cli

  The modules contained in the CLI package each contain a function (detect, extract, enroll, etc.) which is a command within the broader CLI toolkit along with the assorted helper functions.

8.3 cli/database/__init__.py File Reference

Namespaces

- briar.cli.database

8.4 evaluation/__init__.py File Reference

Namespaces

- briar.evaluation

Functions

- def runStages (stages, report=False)

8.5 media/__init__.py File Reference

Classes

- class BriarProgress
- class BriarVideoIterator
- class ImageIterator
- class MediaSetIterator
- class ThreadedVideoIterator
- class VideoIterator

Namespaces

- briar.media

Functions

- def aenumerate (asequence, start=0)
- def decodeMedia (media_pb, newsource=None)

  Convert protobuf media into a numpy array.
- def enroll_frames_iter (database_name, video, detect_options=None, extract_options=None, enroll_↩ options=None, det_list_list=None, whole_image=False, request_start=-1)

  Iterates the paths in the media file list, loading them one by one and yielding grpc enroll requests.
- def enroll_frames_iter_async (database_name, video, detect_options=None, extract_options=None, enroll↩ _options=None, det_list_list=None, whole_image=False, request_start=-1)

  Asyncronously Iterates the paths in the media file list, loading them one by one and yielding grpc enroll requests.
- def file_iter (list[str] media_files, briar_pb2.DetectionOptions clientoptions=None, dict options_dict=None, str database_name=None, bool verbose=False, int request_start=-1, callable requestConstructor=None)
- def frame_iter (media_iterator, clientoptions=None, dict options_dict={}, database_name=None, det_list_↩ list=None, whole_image=False, request_start=-1, requestConstructor=None)
- def ImageGenerator (filepath, start=None, stop=None, unit=None, options=None)
- def isFinalFrame (request)
- def single_frame_generate (frame, frame_num, ittype, filepath, start_frame, stop_frame, fps, video_length, clientoptions=None, dict options_dict={}, database_name=None, det_list_list=None, whole_image=False, request_start=-1, frame_load_time_start=-1, file_level_client_time_end=-1, requestConstructor=None)

8.6 sigset/__init__.py File Reference

Namespaces

- briar.sigset

8.7 timing/__init__.py File Reference

Namespaces

- briar.timing

Functions

- def end_duration (reply)
- def generate_progress (frame_id, media)
- def loadDurationsFolder (durations_directory)
- def parseDurations (durationsperfile_dictionary)
- def print_duration (name, duration)
- def print_durations (durations)
- def save_durations (media_file, durations_list, options, operation, modality=None)
- def start_duration (request, reply)
- def timeElapsed (duration)
- def timestamp ()

Variables

- string DURATION_FILE_EXT = ".durations"

8.8 __main__.py File Reference

Namespaces

- briar.__main__

8.9 briar_cli.py File Reference

Namespaces

- briar.briar_cli

  Created on 2021 at Oak Ridge National Laboratory.

Functions

- def briar_command_line ()

  Entry point for the CLI - switches on the first command line argument (such as 'status', 'detect', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.
- def briar_database_command_line ()

  Entry point for the Database CLI - switches on the second command line argument (such as 'delete', 'merge', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.
- def briar_test_command_line ()

  Entry point for the Test CLI - switches on the second command line argument (such as 'delete', 'merge', etc) and builds the parser and help messages based upon the callback defined in COMMANDS.
- def incomplete ()

Variables

- dictionary COMMANDS
- dictionary DATABASE_COMMANDS
- string DETECTION_FILE_EXT = ".detection"
- int FACE_COUNT = 0
- string MATCHES_FILE_EXT = '.matches'
- string TEMPLATE_FILE_EXT = '.template'

8.10 briar_client.py File Reference

Classes

- class BriarClient

  Provide a client to a BRIAR service.

Namespaces

- briar.briar_client

  Copyright 2021 Oak Ridge National Laboratory.

Functions

- def _initialize_worker (server_address, proc_number, thread_number, count_q)
- def _shutdown_worker ()

Variables

- _client_identifier_singleton = None
- _worker_channel_singleton = None
- _worker_port_singleton = None
- _worker_proccess_position_singleton = None
- _worker_stub_singleton = None
- _worker_thread_position_singleton = None

8.11 briar_media.py File Reference

Classes

- class BriarMedia
- class MediaStream

Namespaces

- briar.briar_media

  Defines a media class which acts as a wrapper for image and video files.

Functions

- def briar_media_from_pb2 (pb2_object)
- def briar_media_to_pb2 (media)
- def load_media_from_folder (folder_path, recursive=False)
- def load_media_from_image (image_path)
- def load_media_from_numpy (numpy_array)

8.12 cli/connection.py File Reference

Namespaces

- briar.cli.connection

Functions

- def addConnectionOptions (parser)

  Accumulatively add options for connecting to the Briar API service.

Variables

- int DEFAULT_MAX_ASYNC = 8
- int DEFAULT_MAX_MESSAGE_SIZE = 64 ∗ 1024 ∗ 1024 ∗ 8

8.13 cli/database/checkpoint.py File Reference

Namespaces

- briar.cli.database.checkpoint

Functions

- def database_checkpoint (options=None, args=None, input_command=None, ret=False)

8.14 cli/database/checkpoint_subject.py File Reference

Namespaces

- briar.cli.database.checkpoint_subject

Functions

- def database_checkpoint_subject (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseCheckpointSubjectOptions (inputCommand=None)

  Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

8.15 cli/database/common.py File Reference

Namespaces

- briar.cli.database.common

Functions

- def db_no_exist (name)

8.16 cli/database/compute_scores.py File Reference

Namespaces

- briar.cli.database.compute_scores

Functions

- def addDatabaseComputeScore_options2proto (options)
- def addDatabaseComputeScoreOptions (parser)

  Add options for search of a database using a database.
- def database_compute_verify (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a batch verification betweeen the specified databases using specified probe database.
- def parseDatabaseComputeScoreOptions (inputCommand=None)

  Generate options for matching databases against other databases, and parse command line arguments into the API call.

8.17 cli/database/compute_search.py File Reference

Namespaces

- briar.cli.database.compute_search

Functions

- def database_compute_search (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a search within the specified database using specified probe database.

8.18 cli/database/create.py File Reference

Namespaces

- briar.cli.database.create

Functions

- def database_create (options=None, args=None, input_command=None, ret=False)

8.19 cli/database/delete.py File Reference

Namespaces

- briar.cli.database.delete

Functions

- def database_delete (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseDeleteOptions (inputCommand=None)

  Generate options for Deleting a pre-existing database and parse command line arguments into API call.

8.20 cli/database/finalize.py File Reference

Namespaces

- briar.cli.database.finalize

Functions

- def database_finalize (options=None, args=None, input_command=None, ret=False)

  Parses the command line options and saves the database to disk.
- def finalizeParseOptions (inputCommand=None)

  Generate options for running 'finalize' (saving the loaded databases) and parse command line arguments into them.

8.21 cli/finalize.py File Reference

Namespaces

- briar.cli.finalize

Functions

- def database_finalize (options=None, args=None, input_command=None, ret=False)

  Parses the command line options and saves the database to disk.
- def finalizeParseOptions (inputCommand=None)

  Generate options for running 'finalize' (saving the loaded databases) and parse command line arguments into them.

8.22 cli/database/info.py File Reference

Namespaces

- briar.cli.database.info

Functions

- def database_info (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseInfoOptions (inputCommand=None)

  Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

8.23 cli/database/list.py File Reference

Namespaces

- briar.cli.database.list

Functions

- def database_list (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseListOptions (inputCommand=None)

  Generate options for listing all pre-existing databases and parse command line arguments into them.

8.24 cli/database/list_entries.py File Reference

Namespaces

- briar.cli.database.list_entries

Functions

- def database_list_entries (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseListEntriesOptions (inputCommand=None)

  Generate options for Listing entries within a pre-existing database and parse command line arguments into an API call.

8.25 cli/database/load.py File Reference

Namespaces

- briar.cli.database.load

Functions

- def database_load ()

8.26 cli/database/merge.py File Reference

Namespaces

- briar.cli.database.merge

Functions

- def database_merge (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseMergeOptions (inputCommand=None)

  Generate options for merging databases and parse command line arguments into the API call.

8.27 cli/database/move_entry.py File Reference

Namespaces

- briar.cli.database.move_entry

Functions

- def database_move_entry (options=None, args=None, input_command=None, ret=False, client=None)
- def parseDatabaseMoveEntryOptions (inputCommand=None)

  Generate options for moving database entries and parse command line arguments into the API call.

8.28 cli/database/refresh.py File Reference

Namespaces

- briar.cli.database.refresh

Functions

- def database_checkpoint (options=None, args=None, input_command=None, ret=False)
- def database_refresh (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseRefreshOptions (inputCommand=None)

  Generate options for getting information about a pre-existing database and parse command line arguments into an API call.

8.29 cli/database/rename.py File Reference

Namespaces

- briar.cli.database.rename

Functions

- def database_load ()
- def database_rename (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseRenameOptions (inputCommand=None)

  Generate options for Renaming a pre-existing database to a new name and parse command line arguments into API call.

8.30 cli/database/retrieve.py File Reference

Namespaces

- briar.cli.database.retrieve

Functions

- def database_retrieve (options=None, args=None, input_command=None, ret=False)
- def parseDatabaseRetrieveOptions (inputCommand=None)

  Generate options for retrieving a pre-existing database and parse command line arguments API call.

8.31 cli/detect.py File Reference

Namespaces

- briar.cli.detect

Functions

- def addDetectorOptions (parser)

  Add options for running detections to the parser.
- def addTrackingOptions (parser)

  Add options for running detections to the parser.
- def detect (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a detection on the specified files.
- def detect_options2proto (options)
- def detectParseOptions (inputCommand=None)

  Generate options for running detections and parse command line arguments into them.
- def detectRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)
- def get_detection_path (media_file, options, i, modality=None, media_id=None)
- def isFinalReply (srvc_pb2.DetectReply reply)
- def save_detections (media_file, reply, options, i, modality=None, media_id=None)
- def tracking_options2proto (options)

Variables

- string DETECTION_FILE_EXT = ".detection"

8.32 cli/enhance.py File Reference

Namespaces

- briar.cli.enhance

Functions

- def addEnhanceOptions (parser)

  Add options for running detections to the parser.
- def enhance (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a detection on the specified files.
- def enhance_options2proto (options)
- def enhanceParseOptions (inputCommand=None)

  Generate options for running enhancement and parse command line arguments into them.
- def enhanceRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options↩ _dict={}, det_list_list=None, str database_name=None)
- def save_Enhancement (media_file, reply, options, i, modality=None)

Variables

- string ENHANCE_FILE_EXT = '.enh'

8.33 cli/enroll.py File Reference

Namespaces

- briar.cli.enroll

Functions

- def addEnrollOptions (parser)

  Add options for enrollment into a database.
- def enroll (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs an enroll on the specified files.
- def enroll_options2proto (options)
- def enrollParseOptions (inputCommand=None)

  Generate options for running enrollments and parse command line arguments into them.
- def enrollRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)

8.34 cli/extract.py File Reference

Namespaces

- briar.cli.extract

Functions

- def addExtractOptions (parser)

  Add options for extractions to the parser.
- def extract (options=None, args=None, inputCommand=None, ret=False)

  Using the options specified in the command line, runs an extract on the specified files.
- def extract_options2proto (options)
- def extractParseOptions (inputCommand=None)

  Generate options for running extracts and parse command line arguments into them.
- def extractRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)
- def save_extractions (media_file, templates, options, i, modality=None, media_id=None)

Variables

- string TEMPLATE_FILE_EXT = '.template'

8.35 cli/media.py File Reference

Namespaces

- briar.cli.media

Functions

- def addMediaOptions (parser)

  Add options for running detections to the parser.
- def collect_files (args, options, extension=None)

  Take the paths specified by 'args' and find all the media files that they define: folders will be searched for all media files contained inside.
- def hasExtension (f, extension)

Variables

- int DEFAULT_MAX_SIZE = 1920

8.36 cli/search.py File Reference

Namespaces

- briar.cli.search

Functions

- def addSearchOptions (parser)

  Add options for search of a database.
- def search (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a search within the specified database using specified probe template(s).
- def search_options2proto (options)
- def searchParseOptions (inputCommand=None)

  Generate options for running searches and parse command line arguments into them.
- def searchRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)

Variables

- string MATCHES_FILE_EXT = '.matches'

8.37 cli/sigset.py File Reference

Namespaces

- briar.cli.sigset

Functions

- def checkpoint_subject_threaded (obj)
- def df_batch_consumer (batch_queue, identifier, port, server_configuration, id_queue, progress_queue)
- def df_batch_generator (list_of_dfs, start, dataset_dir, detect_options, extract_options, enroll_options, options)
- def df_batch_producer (batch_queue, identifier, progress_position, list_of_dataframes, start, dataset_dir, detect_options, extract_options, enroll_options, options)
- def df_row_generator (rowiter, start, dataset_dir, detect_options, extract_options, enroll_options, options)
- def enroll_call_threaded (input)
- def inner_pool_mapper (batch_obj, local_pool, progress_queue, port)
- def parseSigsetEnrollOptions (inputCommand=None)
- def parseSigsetStatsOptions (inputCommand=None)
- def progress_consumer (progress_queue, progress_position, total_len, options)
- def sigset_enroll (input_command=None)
- def sigset_stats (options=None, args=None)

Variables

- my_pool = None
- proc_number = None
- service_address_number = None

8.38 cli/status.py File Reference

Namespaces

- briar.cli.status

Functions

- def get_service_configuration (options=None, args=None, input_command=None, ret=False)
- def print_service_configuration (options=None, args=None)
- def status (options=None, args=None, input_command=None, ret=False)

  Conects to the server and gets status information.
- def statusParseOptions (inputCommand=None)

  Generate options for getting status and parse command line arguments into them.

8.39 cli/test.py File Reference

Classes

- class BriarTest
- class BriarTestResult
- class DatabaseTest
- class DetectTest
- class EnrollTest
- class ExtractTest
- class Fore

Namespaces

- briar.cli.test

Functions

- def detection_output_tests (detection_obj_loaded, testimage, return_media)
- def extraction_output_tests (template_obj_loaded, testimage, return_media)

Variables

- bool use_colorama = False

8.40 cli/track.py File Reference

Namespaces

- briar.cli.track

Functions

- def get_tracklet_path (media_file, options, i, modality=None, media_id=None)
- def save_tracklets (media_file, tracklets, options, i, verbose=False, modality=None, media_id=None)
- def track (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs a detection on the specified files.
- def trackRequestConstructor (briar_pb2.BriarMedia media, briar_pb2.BriarDurations durations, options_↩ dict={}, det_list_list=None, str database_name=None)

Variables

- string TRACKLET_FILE_EXT = ".tracklet"

8.41 cli/verify.py File Reference

Namespaces

- briar.cli.verify

Functions

- def addVerifyOptions (parser)

  Add options for verification to the parser.
- def save_verifications (media_file, reply, options, i, modality=None, media_id=None)
- def verify (options=None, args=None, input_command=None, ret=False)

  Using the options specified in the command line, runs an extract on the specified files.
- def verify_options2proto (options)
- def verifyParseOptions (inputCommand=None)

  Generate options for running verifications and parse command line arguments into them.

Variables

- string VERIFICATION_FILE_EXT = ".verification"

8.42 cli/viz.py File Reference

Namespaces

- briar.cli.viz

Functions

- def viz ()

  Using the options specified in the command line, runs visualization on the specified files.
- def vizParseOptions ()

  Generate options for running detections and parse command line arguments into them.

8.43 evaluation/full_evaluation.py File Reference

Classes

- class Test000InitialConfig
- class Test001SigsetEnrollProbe
- class Test002ProbeDatabaseMerge
- class Test003SigsetEnrollMultiProbe
- class Test004MultiProbeDatabaseMerge
- class Test005SigsetEnrollGalleries
- class Test006GalleryDatabaseMerge
- class Test007SigsetScoreG1
- class Test008SigsetScoreGaitG1
- class Test009SigsetScoreFaceG1
- class Test010SigsetScoreWholeBodyG1
- class Test011SigsetScoreG2
- class Test012SigsetScoreGaitG2
- class Test013SigsetScoreFaceG2
- class Test014SigsetScoreWholeBodyG2
- class Test015SigsetBlendedScoreG1
- class Test016SigsetScoreBlendedGaitG1
- class Test017SigsetScoreBlendedFaceG1
- class Test018SigsetScoreBlendedWholeBodyG1
- class Test019SigsetBlendedScoreG2
- class Test020SigsetScoreBlendedGaitG2
- class Test021SigsetScoreBlendedFaceG2
- class Test022SigsetScoreBlendedWholeBodyG2
- class Test023MultiSigsetScoreG1
- class Test023SigsetSearchOutputFormatting
- class Test024MultiSigsetScoreGaitG1
- class Test024SigsetVerifyOutputFormatting
- class Test025MultiSigsetScoreFaceG1

- class Test026MultiSigsetScoreWholeBodyG1
- class Test027MultiSigsetScoreG2
- class Test028MultiSigsetScoreGaitG2
- class Test029MultiSigsetScoreFaceG2
- class Test030MultiSigsetScoreWholeBodyG2
- class Test031MultiSigsetBlendedScoreG1
- class Test032MultiSigsetScoreBlendedGaitG1
- class Test033MultiSigsetScoreBlendedFaceG1
- class Test034MultiSigsetScoreBlendedWholeBodyG1
- class Test035MultiSigsetBlendedScoreG2
- class Test036MultiSigsetScoreBlendedGaitG2
- class Test037MultiSigsetScoreBlendedFaceG2
- class Test038MultiSigsetScoreBlendedWholeBodyG2

Namespaces

- briar.evaluation.full_evaluation

Functions

- def compute_search (probe_db_name, gal_db_name, probe_sigset_path, output_path, modality=None, blended=False)
- def compute_verify (probe_db_name, gal_db_name, probe_sigset_path, gal_sigset_path, output_path, csv←_path, modality=None, blended=False)
- def get_info (self, db_name)
- def get_multi_info (self, base_db_name)
- def merge_dbs (self, db_name)
- def run_on_multi (self, base_db_name, mapped_function)
- def runall ()
- None setUpClass_main (cls)
- def setUpModule ()

Variables

- string args_string = " --progress "
- string blended_gallery_1_filename = "sigsets_gallery/Blended_Gallery_1.xml"
- string blended_gallery_2_filename = "sigsets_gallery/Blended_Gallery_2.xml"
- category
- string database_blended_gallery_1_name = 'db_eval_phase2_blended_gallery_1'
- string database_blended_gallery_2_name = 'db_eval_phase2_blended_gallery_2'
- string database_gallery_1_name = 'db_eval_phase2_gallery_1'
- string database_gallery_2_name = 'db_eval_phase2_gallery_2'
- string database_multi_probe_name = 'db_eval_phase2_multisubject_probe'
- string database_probe_name = 'db_eval_phase2_probe'
- DATABASE_SUFFIX_FLAG = os.getenv('BRIAR_DATABASE_SUFFIX_FLAG')
- DATASET_DIR = os.getenv('BRIAR_DATASET_DIR')
- string enroll_args = " --auto-create-database "
- EVAL_PHASE = os.getenv('BRIAR_EVAL_PHASE')
- EVALUATION_DIR = os.getenv('BRIAR_EVALUATION_DIR')
- EVALUATION_MULTISUBJECT_DIR = os.getenv('BRIAR_MULTISUBJECT_EVALUATION_DIR')
- gallery1_blended_sigset
- gallery1_sigset

- gallery2_blended_sigset
- gallery2_sigset
- gallery_1_blended_sigset_path
- string gallery_1_filename = "sigsets_gallery/Gallery_1.xml"
- gallery_1_sigset_path
- gallery_2_blended_sigset_path
- string gallery_2_filename = "sigsets_gallery/Gallery_2.xml"
- gallery_2_sigset_path
- generate_report = os.environ.get('REPORT', False)
- string media_args = " --no-save "
- module
- string multisubject_probe_filename = "sigsets_multiperson/Probe_BTS_briar-rd_multi.xml"
- int number_of_partitions = 1
- OUTPUT_DIR = os.getenv('BRIAR_EVALUATION_OUTPUT_DIR')
- list port_list = [ ]
- string probe_filename = "sigsets_main/Probe_BTS_briar-rd_ALL.xml"
- probe_multisubject_sigset
- probe_multisubject_sigset_path
- probe_sigset
- probe_sigset_path
- bool requires_database_merge = False
- run_multisubject_evaluation
- RUN_STAGES = os.getenv('RUN_STAGES',None)
- sortTestMethodsUsing
- list stages_temp = [ ]
- USE_SINGLE_SUBJECT = os.getenv('BRIAR_USE_SINGLE_SUBJECT')
- USES_FRONTEND_MERGING = os.getenv('BRIAR_USE_FRONTEND_MERGING')

8.44 evaluation/stage1_probe_enroll.py File Reference

Namespaces

- briar.evaluation.stage1_probe_enroll

Variables

- generate_report = os.environ.get('REPORT',False)
- main = unittest.TestProgram

8.45 evaluation/stage2.1_gallery1_simple_enroll.py File Reference

Namespaces

- briar.evaluation.stage2

Variables

- generate_report = os.environ.get('REPORT', False)
- main = unittest.TestProgram

8.46 evaluation/stage2.2_gallery2_simple_enroll.py File Reference

Namespaces

- briar.evaluation.stage2

8.47 evaluation/stage2.3_gallery1_blended_enroll.py File Reference

Namespaces

- briar.evaluation.stage2

8.48 evaluation/stage2.4_gallery2_blended_enroll.py File Reference

Namespaces

- briar.evaluation.stage2

8.49 evaluation/stage3_result_scoring.py File Reference

Namespaces

- briar.evaluation.stage3_result_scoring

Variables

- generate_report = os.environ.get('REPORT', False)
- main = unittest.TestProgram

8.50 grpc_json.py File Reference

Classes

- class GrpcDecoder

  Object which extends the JSONDecoded to allow it to read saved gRPC files.
- class GrpcEncoder

  Encoder class which extends the normal JSON encoder to allow the encoding of gRPC objects.

Namespaces

- briar.grpc_json

  I couldn't find a good library to write gRPC/protobuf objects to disk so this is my own implementation of a generalized json converter for said objects.

Functions

- def dict_to_proto_obj (obj_dict, options=None)

  *Take the object dictionary, read the dict which is saved in in the '**class**' key, and initialize it with values stored in the dictionary's key/value pairs.*
- def load (load_path, options=None)

  Load the json file at the given directory, reloading dictionaries with "__class__" fields into the specified objects and initializing them with values defined by key/value pairs within the dictionary.
- def proto_obj_to_dict (obj, options=None)

  Takes a general gRPC/protobuf object, eliminates the unnecessary fields, and stores the data in a dict.
- def save (json_obj, save_path, options=None)

  Save a list or dictionary containing protobuf classes to a json file.

Variables

- list ATTRIB_IGNORE

8.51 media/VideoStream.py File Reference

Classes

- class FileVideoStream_cv2
- class FileVideoStream_imageio

Namespaces

- briar.media.VideoStream

8.52 media/visualize.py File Reference

Classes

- class match_matrix_visualizer

Namespaces

- briar.media.visualize

Functions

- def decode_track (tracklet, framenum=None, newsource=None)
- def get_frame (vidfile)
- def playVideo (vidfiles, titles=None, attributes=None, isvideo=True)
- def update_annot (ind, visualizer, pltloc, playvid=False)
- def update_annot_filename_only (ind, visualizer, pltloc)
- def visualize_detection (detection_path)
- def visualize_matches (matches_path)
- def visualize_track (track_path, options)
- def windowclick (event, visualizer)
- def windowhover (event, visualizer, playvid=False)
- def windowhover_filename_only (event, visualizer)

Variables

- string fdir = "/Users/2r6/Projects/briar/briar-api/media/test_probe/clinton3.matches"
- list files = [os.path.join(fdir, f) for f in os.listdir(fdir)]

8.53 media_converters.py File Reference

Namespaces

- briar.media_converters

  Contained in this are functions for converting numpy arrays into various protobuf objects and back again since numpy arrays cannot be sent directly over gRPC.

Functions

- def attribute_find (key, List[briar_pb2.Attribute] attributes)
- def attribute_proto2val (briar_pb2.Attribute attribute)
- def attribute_retrieve (briar_pb2.Attribute attribute, return_type=False)
- def attribute_val2proto (key, val, briar_pb2.BriarDataType override_type=None)
- def check_if_delete_request (srvc_pb2.DatabaseInsertRequest request)
- def check_if_delete_request_due_to_error (srvc_pb2.DatabaseInsertRequest request)
- def get_entry_id_list (srvc_pb2.DatabaseInsertRequest request)
- def image_cv2proto (im, compression='uint8', quality=99, flip_channels=True)

  Convert a cv2 numpy array to a protobuf format.
- def image_file2proto (imfile, path_map={})
- def image_np2proto (im, compression='uint8', quality=99, flip_channels=True)

  Convert a numpy array to a protobuf format.
- def image_proto2cv (pb_data, flip_channels=False)

  Convert a protobuf BriarMedia image to a cv2 numpy array.
- def image_proto2np (pb_data, flip_channels=True)

  Convert a protobuf image to a numpy array.
- def matrix_np2proto (mat)

  Convert a numpy matrix into a BriarMatrix.
- def matrix_proto2np (protomat)

  Convert a protobuf matrix into a numpy matrix.
- def modality_proto2string (modality)
- def modality_string2proto (modality)
- def pathmap_path2remotepath (path, path_map, exclude_cases_containing_folder=['mugshots'])
- def pathmap_str2dict (path_map)
- def subjectID_int2str (subjectid)
- def subjectID_str2int (subjectid)
- def subjectList_list2string (subject_list_str, chomp=True)
- def subjectList_string2list (subject_list)
- def tracklet_list2proto (track_list)
- def vector_np2proto (vec)

  Convert a 1 dimensional np array into a BriarVector.
- def vector_proto2np (protovec)

  Convert a protobuf vector into a numpy array.
- def video_file2proto (vidfile, start, end, path_map={})

Variables

- dictionary attribute_type_name_map = {'int': 'ivalue', 'float': 'fvalue','string':'text'}
- dictionary modalityDict
- dictionary reverseModalityDict = {modalityDict[k]: k for k in modalityDict}

8.54 readme-cli.md File Reference

8.55 sigset/parse.py File Reference

Namespaces

- briar.sigset.parse

Functions

- def create_test_sigset (str sigset_probe_file, str sigset_gallery_file, str base_dir, output_dir)
- def expandTree (root, level=0, spaces=3)
- def parseBriarSigset (filename)

Variables

- args = sys.argv

Index