

BRIAR Client Documentation

Generated by Doxygen 1.8.17

1 BRIAR Command Line Interface (CLI) and Client	1
2 Namespace Index	3
2.1 Packages	3
3 Hierarchical Index	5
3.1 Class Hierarchy	5
4 Class Index	7
4.1 Class List	7
5 File Index	9
5.1 File List	9
6 Namespace Documentation	11
6.1 briar Namespace Reference	11
6.1.1 Function Documentation	12
6.1.1.1 dyn_import()	12
6.1.1.2 serve()	12
6.1.2 Variable Documentation	12
6.1.2.1 __version__	12
6.1.2.2 DEFAULT_MAX_MESSAGE_SIZE	12
6.1.2.3 DEFAULT_PORT	12
6.1.2.4 DEFAULT_SERVE_PORT	13
6.2 briar.__main__ Namespace Reference	13
6.3 briar.briar_cli Namespace Reference	13
6.3.1 Detailed Description	13
6.3.2 Function Documentation	13
6.3.2.1 briar_command_line()	14
6.3.2.2 briar_database_command_line()	14
6.3.2.3 briar_test_command_line()	14
6.3.2.4 incomplete()	14
6.3.3 Variable Documentation	15
6.3.3.1 COMMANDS	15
6.3.3.2 DATABASE_COMMANDS	15
6.3.3.3 DETECTION_FILE_EXT	15
6.3.3.4 FACE_COUNT	15
6.3.3.5 MATCHES_FILE_EXT	16
6.3.3.6 TEMPLATE_FILE_EXT	16
6.4 briar.briar_client Namespace Reference	16
6.4.1 Detailed Description	16
6.5 briar.briar_media Namespace Reference	16
6.5.1 Detailed Description	17
6.5.2 Function Documentation	17

6.5.2.1 briar_media_from_pb2()	17
6.5.2.2 briar_media_to_pb2()	17
6.5.2.3 load_media_from_folder()	17
6.5.2.4 load_media_from_image()	17
6.5.2.5 load_media_from_numpy()	17
6.6 briar.cli Namespace Reference	18
6.6.1 Detailed Description	18
6.7 briar.cli.annotate Namespace Reference	18
6.7.1 Function Documentation	18
6.7.1.1 annotate()	18
6.8 briar.cli.connection Namespace Reference	19
6.8.1 Function Documentation	19
6.8.1.1 addConnectionOptions()	19
6.8.2 Variable Documentation	19
6.8.2.1 DEFAULT_MAX_ASYNC	19
6.8.2.2 DEFAULT_MAX_MESSAGE_SIZE	19
6.9 briar.cli.database Namespace Reference	20
6.9.1 Function Documentation	20
6.9.1.1 database_delete()	20
6.9.1.2 database_info()	20
6.9.1.3 database_list()	21
6.9.1.4 database_list_entries()	21
6.9.1.5 database_merge()	21
6.9.1.6 database_rename()	21
6.9.1.7 database_retrieve()	21
6.9.1.8 db_no_exist()	22
6.9.1.9 parseDatabaseDeleteOptions()	22
6.9.1.10 parseDatabaseInfoOptions()	22
6.9.1.11 parseDatabaseListEntriesOptions()	22
6.9.1.12 parseDatabaseListOptions()	23
6.9.1.13 parseDatabaseMergeOptions()	23
6.9.1.14 parseDatabaseRenameOptions()	23
6.9.1.15 parseDatabaseRetrieveOptions()	23
6.10 briar.cli.detect Namespace Reference	24
6.10.1 Function Documentation	24
6.10.1.1 addDetectorOptions()	24
6.10.1.2 addTrackingOptions()	24
6.10.1.3 detect()	25
6.10.1.4 detect_file_iter()	25
6.10.1.5 detect_options2proto()	25
6.10.1.6 detectParseOptions()	26
6.10.1.7 save_detections()	26

6.10.1.8 tracking_options2proto()	26
6.10.2 Variable Documentation	26
6.10.2.1 DETECTION_FILE_EXT	26
6.11 briar.cli.enhance Namespace Reference	26
6.11.1 Function Documentation	27
6.11.1.1 addEnhanceOptions()	27
6.11.1.2 enhance()	27
6.11.1.3 enhance_file_iter()	27
6.11.1.4 enhance_options2proto()	28
6.11.1.5 enhanceParseOptions()	28
6.11.1.6 save_Enhancement()	28
6.12 briar.cli.enroll Namespace Reference	28
6.12.1 Function Documentation	28
6.12.1.1 addEnrollOptions()	28
6.12.1.2 enroll()	29
6.12.1.3 enroll_options2proto()	29
6.12.1.4 enrollParseOptions()	29
6.13 briar.cli.extract Namespace Reference	30
6.13.1 Function Documentation	30
6.13.1.1 addExtractOptions()	30
6.13.1.2 extract()	30
6.13.1.3 extract_options2proto()	31
6.13.1.4 extractParseOptions()	31
6.13.1.5 save_extractions()	31
6.13.2 Variable Documentation	31
6.13.2.1 TEMPLATE_FILE_EXT	31
6.14 briar.cli.finalize Namespace Reference	31
6.14.1 Function Documentation	32
6.14.1.1 database_finalize()	32
6.14.1.2 finalizeParseOptions()	32
6.15 briar.cli.media Namespace Reference	32
6.15.1 Function Documentation	32
6.15.1.1 addMediaOptions()	32
6.15.1.2 collect_files()	33
6.15.1.3 hasExtension()	33
6.15.2 Variable Documentation	33
6.15.2.1 DEFAULT_MAX_SIZE	33
6.16 briar.cli.search Namespace Reference	34
6.16.1 Function Documentation	34
6.16.1.1 addSearchOptions()	34
6.16.1.2 search()	34
6.16.1.3 search_options2proto()	35

6.16.1.4 searchParseOptions()	35
6.16.2 Variable Documentation	35
6.16.2.1 MATCHES_FILE_EXT	35
6.17 briar.cli.sigset Namespace Reference	35
6.17.1 Function Documentation	35
6.17.1.1 parseSigsetEnrollOptions()	36
6.17.1.2 parseSigsetStatsOptions()	36
6.17.1.3 sigset_enroll()	36
6.17.1.4 sigset_stats()	36
6.18 briar.cli.status Namespace Reference	36
6.18.1 Function Documentation	36
6.18.1.1 status()	37
6.18.1.2 statusParseOptions()	37
6.19 briar.cli.test Namespace Reference	37
6.19.1 Function Documentation	37
6.19.1.1 detection_output_tests()	38
6.19.1.2 extraction_output_tests()	38
6.20 briar.cli.track Namespace Reference	38
6.20.1 Function Documentation	38
6.20.1.1 save_tracklets()	38
6.20.1.2 track()	39
6.20.2 Variable Documentation	39
6.20.2.1 TRACKLET_FILE_EXT	39
6.21 briar.cli.viz Namespace Reference	39
6.21.1 Function Documentation	39
6.21.1.1 viz()	39
6.21.1.2 vizParseOptions()	40
6.22 briar.functions Namespace Reference	40
6.22.1 Function Documentation	40
6.22.1.1 new_uuid()	40
6.23 briar.grpc_json Namespace Reference	40
6.23.1 Detailed Description	41
6.23.2 Function Documentation	41
6.23.2.1 dict_to_proto_obj()	41
6.23.2.2 load()	42
6.23.2.3 proto_obj_to_dict()	42
6.23.2.4 save()	42
6.23.3 Variable Documentation	43
6.23.3.1 ATTRIB_IGNORE	43
6.24 briar.ibriar Namespace Reference	43
6.24.1 Variable Documentation	43
6.24.1.1 bcl	43

6.24.1.2 DEFAULT_ARGS	44
6.25 briar.media Namespace Reference	44
6.25.1 Function Documentation	44
6.25.1.1 decodeMedia()	44
6.25.1.2 ImageGenerator()	44
6.26 briar.media.visualize Namespace Reference	45
6.26.1 Function Documentation	45
6.26.1.1 visualize_detection()	45
6.26.1.2 visualize_matches()	45
6.26.1.3 visualize_track()	45
6.26.2 Variable Documentation	45
6.26.2.1 fdir	45
6.26.2.2 files	46
6.27 briar.media_converters Namespace Reference	46
6.27.1 Detailed Description	46
6.27.2 Function Documentation	46
6.27.2.1 image_cv2proto()	46
6.27.2.2 image_np2proto()	47
6.27.2.3 image_proto2cv()	47
6.27.2.4 image_proto2np()	47
6.27.2.5 matrix_np2proto()	48
6.27.2.6 matrix_proto2np()	48
6.27.2.7 modality_proto2string()	48
6.27.2.8 modality_string2proto()	49
6.27.2.9 vector_np2proto()	49
6.27.2.10 vector_proto2np()	49
6.27.3 Variable Documentation	49
6.27.3.1 modalityDict	49
6.27.3.2 reverseModalityDict	50
6.28 briar.sigset Namespace Reference	50
6.29 briar.sigset.parse Namespace Reference	50
6.29.1 Function Documentation	50
6.29.1.1 expandTree()	50
6.29.1.2 parseBriarSigset()	50
6.30 briar.timing Namespace Reference	51
6.30.1 Function Documentation	51
6.30.1.1 end_duration()	51
6.30.1.2 generate_progress()	51
6.30.1.3 print_duration()	51
6.30.1.4 print_durations()	52
6.30.1.5 save_durations()	52
6.30.1.6 start_duration()	52

6.30.1.7 timeElapsed()	52
6.30.1.8 timestamp()	52
6.30.2 Variable Documentation	52
6.30.2.1 DURATION_FILE_EXT	52
7 Class Documentation	53
7.1 BriarCLI Class Reference	53
7.1.1 Constructor & Destructor Documentation	53
7.1.1.1 __init__()	53
7.1.2 Member Function Documentation	54
7.1.2.1 _cmd_line()	54
7.1.3 Member Data Documentation	54
7.1.3.1 _base_prompt_text	54
7.1.3.2 _completer	54
7.1.3.3 _prompt_hist_path	54
7.1.3.4 _PROMPT_TXT	54
7.2 BriarCLICompleter Class Reference	55
7.2.1 Constructor & Destructor Documentation	56
7.2.1.1 __init__()	56
7.2.2 Member Function Documentation	56
7.2.2.1 _complete_base_cmd()	56
7.2.2.2 _complete_cmds()	56
7.2.2.3 _complete_from_schema()	56
7.2.2.4 _complete_kwargs()	56
7.2.2.5 _get_completions()	57
7.2.2.6 _suggest_cmds()	57
7.2.2.7 _suggest_kwargs()	57
7.2.2.8 get_completions()	57
7.3 BriarClient Class Reference	58
7.3.1 Detailed Description	61
7.3.2 Constructor & Destructor Documentation	61
7.3.2.1 __init__()	61
7.3.3 Member Function Documentation	61
7.3.3.1 _enroll_frames_iter()	61
7.3.3.2 _enroll_frames_iter2()	62
7.3.3.3 database_create()	62
7.3.3.4 database_insert()	62
7.3.3.5 database_list_templates()	63
7.3.3.6 database_remove_templates()	63
7.3.3.7 database_retrieve()	64
7.3.3.8 detect()	64
7.3.3.9 detect_files()	65

7.3.3.10 detect_files_iter()	65
7.3.3.11 detect_frames()	65
7.3.3.12 detect_frames_iter()	66
7.3.3.13 enhance()	66
7.3.3.14 enroll()	66
7.3.3.15 enroll_file_iter()	67
7.3.3.16 enroll_files()	67
7.3.3.17 enroll_frames()	68
7.3.3.18 enroll_frames2()	68
7.3.3.19 enroll_frames_iter()	70
7.3.3.20 extract()	71
7.3.3.21 extract_file_iter()	71
7.3.3.22 extract_files()	72
7.3.3.23 extract_frames()	72
7.3.3.24 extract_frames_iter()	73
7.3.3.25 finalize()	73
7.3.3.26 get_database_names()	73
7.3.3.27 get_status()	74
7.3.3.28 load_database()	74
7.3.3.29 print_verbose()	74
7.3.3.30 retrieve_req_iter()	75
7.3.3.31 search()	75
7.3.3.32 search_file_iter()	76
7.3.3.33 search_files()	77
7.3.3.34 track()	78
7.3.3.35 track_file_iter()	78
7.3.3.36 track_files()	79
7.3.3.37 verify()	79
7.3.4 Member Data Documentation	80
7.3.4.1 channel	80
7.3.4.2 DEFAULT_PORT	80
7.3.4.3 options	80
7.3.4.4 stub	80
7.4 BriarMedia Class Reference	80
7.4.1 Constructor & Destructor Documentation	81
7.4.1.1 __init__()	81
7.4.2 Member Data Documentation	81
7.4.2.1 channels	81
7.4.2.2 DATA_TYPES	81
7.4.2.3 datetime	81
7.4.2.4 description	82
7.4.2.5 fps	82

7.4.2.6 height	82
7.4.2.7 IMAGE_FORMATS	82
7.4.2.8 len	82
7.4.2.9 metadata	82
7.4.2.10 source	82
7.4.2.11 VIDEO_FORMATS	83
7.4.2.12 width	83
7.5 BriarProgress Class Reference	83
7.5.1 Constructor & Destructor Documentation	83
7.5.1.1 __init__()	83
7.5.2 Member Function Documentation	84
7.5.2.1 update()	84
7.5.3 Member Data Documentation	84
7.5.3.1 desc	84
7.5.3.2 enabled	84
7.5.3.3 leave	84
7.5.3.4 name	84
7.5.3.5 pbar	84
7.5.3.6 position	85
7.5.3.7 prevstep	85
7.5.3.8 tqdm	85
7.6 BriarTest Class Reference	85
7.6.1 Constructor & Destructor Documentation	86
7.6.1.1 __init__()	86
7.6.2 Member Function Documentation	86
7.6.2.1 description()	86
7.6.2.2 run()	86
7.6.2.3 test()	86
7.6.3 Member Data Documentation	86
7.6.3.1 testdata_folder	86
7.7 BriarResult Class Reference	87
7.7.1 Constructor & Destructor Documentation	87
7.7.1.1 __init__()	87
7.7.2 Member Data Documentation	87
7.7.2.1 level	87
7.7.2.2 name	87
7.7.2.3 passed	87
7.7.2.4 reason	88
7.8 CLICommands Class Reference	88
7.8.1 Member Data Documentation	88
7.8.1.1 ARGS	88
7.8.1.2 CMD_SCHEMA	88

7.8.1.3 CMD_SYNTAX	88
7.8.1.4 COMMAND	89
7.8.1.5 DETECT	89
7.8.1.6 ENROLL	89
7.8.1.7 EXTRACT	89
7.8.1.8 MODE	89
7.9 DatabaseTest Class Reference	90
7.9.1 Member Function Documentation	90
7.9.1.1 test()	90
7.10 DetectTest Class Reference	91
7.10.1 Member Function Documentation	92
7.10.1.1 description()	92
7.10.1.2 test_1_detection_image()	92
7.10.1.3 test_2_detection_image_output()	92
7.10.1.4 test_3_detection_image_withreturn()	92
7.10.1.5 test_4_detection_image_output_withreturn()	92
7.10.2 Member Data Documentation	92
7.10.2.1 detection_file_path	93
7.10.2.2 output_path	93
7.10.2.3 testim_path	93
7.11 EnrollTest Class Reference	93
7.11.1 Member Function Documentation	94
7.11.1.1 test()	94
7.12 ExtractTest Class Reference	94
7.12.1 Member Function Documentation	95
7.12.1.1 description()	95
7.12.1.2 test_1_extraction_image()	95
7.12.1.3 test_2_extraction_image_output()	95
7.12.2 Member Data Documentation	96
7.12.2.1 detection_file_path	96
7.12.2.2 output_path	96
7.12.2.3 template_file_path	96
7.12.2.4 testim_path	96
7.13 GrpcDecoder Class Reference	96
7.13.1 Detailed Description	97
7.13.2 Constructor & Destructor Documentation	97
7.13.2.1 __init__()	97
7.13.3 Member Function Documentation	97
7.13.3.1 default()	97
7.13.4 Member Data Documentation	98
7.13.4.1 options	98
7.14 GrpcEncoder Class Reference	98

7.14.1 Detailed Description	99
7.14.2 Constructor & Destructor Documentation	99
7.14.2.1 __init__()	99
7.14.3 Member Function Documentation	99
7.14.3.1 default()	99
7.14.4 Member Data Documentation	100
7.14.4.1 options	100
7.15 ImageIterator Class Reference	100
7.15.1 Constructor & Destructor Documentation	101
7.15.1.1 __init__()	101
7.15.2 Member Function Documentation	101
7.15.2.1 __iter__()	101
7.15.2.2 __len__()	102
7.15.2.3 __next__()	102
7.15.3 Member Data Documentation	102
7.15.3.1 filepath	102
7.15.3.2 fps	102
7.15.3.3 frame	102
7.15.3.4 frame_count	102
7.15.3.5 frame_height	102
7.15.3.6 frame_width	103
7.15.3.7 i	103
7.15.3.8 isOpened	103
7.15.3.9 length	103
7.15.3.10 msec	103
7.15.3.11 pos	103
7.15.3.12 processed	103
7.15.3.13 start_frame	103
7.15.3.14 stop_frame	104
7.16 MediaStream Class Reference	104
7.16.1 Constructor & Destructor Documentation	104
7.16.1.1 __init__()	104
7.16.2 Member Function Documentation	104
7.16.2.1 __iter__()	104
7.16.3 Member Data Documentation	104
7.16.3.1 _media_list	105
7.17 Rect Class Reference	105
7.17.1 Detailed Description	105
7.17.2 Constructor & Destructor Documentation	105
7.17.2.1 __init__()	105
7.17.3 Member Data Documentation	105
7.17.3.1 height	106

7.17.3.2 width	106
7.17.3.3 x	106
7.17.3.4 y	106
7.18 Videoliterator Class Reference	106
7.18.1 Constructor & Destructor Documentation	107
7.18.1.1 __init__()	107
7.18.2 Member Function Documentation	108
7.18.2.1 __iter__()	108
7.18.2.2 __len__()	108
7.18.2.3 __next__()	108
7.18.3 Member Data Documentation	108
7.18.3.1 cap	108
7.18.3.2 filepath	108
7.18.3.3 fps	108
7.18.3.4 frame_count	109
7.18.3.5 frame_height	109
7.18.3.6 frame_width	109
7.18.3.7 i	109
7.18.3.8 isOpened	109
7.18.3.9 length	109
7.18.3.10 msec	109
7.18.3.11 pos	109
7.18.3.12 processed	110
7.18.3.13 start_frame	110
7.18.3.14 stop_frame	110
8 File Documentation	111
8.1 __init__.py File Reference	111
8.2 cli/__init__.py File Reference	111
8.3 media/__init__.py File Reference	112
8.4 sigset/__init__.py File Reference	112
8.5 timing/__init__.py File Reference	112
8.6 __main__.py File Reference	113
8.7 briar_cli.py File Reference	113
8.8 briar_client.py File Reference	113
8.9 briar_media.py File Reference	114
8.10 cli/annotate.py File Reference	114
8.11 cli/connection.py File Reference	114
8.12 cli/database.py File Reference	115
8.13 cli/detect.py File Reference	115
8.14 cli/enhance.py File Reference	116
8.15 cli/enroll.py File Reference	116

8.16 cli/extract.py File Reference	116
8.17 cli/finalize.py File Reference	117
8.18 cli/media.py File Reference	117
8.19 cli/search.py File Reference	118
8.20 cli/sigset.py File Reference	118
8.21 cli/status.py File Reference	118
8.22 cli/test.py File Reference	119
8.23 cli/track.py File Reference	119
8.24 cli/viz.py File Reference	119
8.25 functions.py File Reference	120
8.26 grpc_json.py File Reference	120
8.27 ibriar.py File Reference	121
8.28 media/visualize.py File Reference	121
8.29 media_converters.py File Reference	121
8.30 readme-cli.md File Reference	122
8.31 sigset/parse.py File Reference	122
Index	123

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 tools, 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 thoroughly in the briar protobuf and stubs documentation

Chapter 2

Namespace Index

2.1 Packages

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

briar	11
briar.__main__	13
briar.briar_cli	13
briar.briar_client	16
briar.briar_media	16
briar.cli	18
briar.cli.annotate	18
briar.cli.connection	19
briar.cli.database	20
briar.cli.detect	24
briar.cli.enhance	26
briar.cli.enroll	28
briar.cli.extract	30
briar.cli.finalize	31
briar.cli.media	32
briar.cli.search	34
briar.cli.sigset	35
briar.cli.status	36
briar.cli.test	37
briar.cli.track	38
briar.cli.viz	39
briar.functions	40
briar.grpc_json	40
briar.ibriar	43
briar.media	44
briar.media.visualize	45
briar.media_converters	46
briar.sigset	50
briar.sigset.parse	50
briar.timing	51

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

BriarCLI	53
BriarMedia	80
BriarProgress	83
BriarTest	85
DatabaseTest	90
DetectTest	91
EnrollTest	93
ExtractTest	94
BriarTestResult	87
CLICommands	88
JSONDecoder	
GrpcDecoder	96
JSONEncoder	
GrpcEncoder	98
MediaStream	104
object	
BriarClient	58
ImageIterator	100
VideoIterator	106
Rect	105
Completer	
BriarCLICompleter	55

Chapter 4

Class Index

4.1 Class List

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

BriarCLI	53
BriarCLICompleter	55
BriarClient	
Provide a client to a BRIAR service	58
BriarMedia	80
BriarProgress	83
BriarTest	85
BriarTestResult	87
CLICommands	88
DatabaseTest	90
DetectTest	91
EnrollTest	93
ExtractTest	94
GrpcDecoder	
Object which extends the JSONDecoded to allow it to read saved gRPC files	96
GrpcEncoder	
Encoder class which extends the normal JSON encoder to allow the encoding of gRPC objects	98
ImageIterator	100
MediaStream	104
Rect	105
VideoIterator	106

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

__init__.py	111
__main__.py	113
briar_cli.py	113
briar_client.py	113
briar_media.py	114
functions.py	120
grpc_json.py	120
ibriar.py	121
media_converters.py	121
cli/__init__.py	111
cli/annotate.py	114
cli/connection.py	114
cli/database.py	115
cli/detect.py	115
cli/enhance.py	116
cli/enroll.py	116
cli/extract.py	116
cli/finalize.py	117
cli/media.py	117
cli/search.py	118
cli/sigset.py	118
cli/status.py	118
cli/test.py	119
cli/track.py	119
cli/viz.py	119
media/__init__.py	112
media/visualize.py	121
sigset/__init__.py	112
sigset/parse.py	122
timing/__init__.py	112

Chapter 6

Namespace Documentation

6.1 briar Namespace Reference

Namespaces

- [__main__](#)
- [briar_cli](#)
- [briar_client](#)
- [briar_media](#)
- [cli](#)
- [functions](#)
- [grpc_json](#)
- [ibriar](#)
- [media](#)
- [media_converters](#)
- [sigset](#)
- [timing](#)

Classes

- class [Rect](#)

Functions

- def [dyn_import](#) (name)
- def [serve](#) (serviceClass, options=None, serve_port=None)
Initialize and run the BRIARService.

Variables

- string [__version__](#) = '1.2.4'
- int [DEFAULT_MAX_MESSAGE_SIZE](#) = 64*1024*1024
- string [DEFAULT_PORT](#) = "0.0.0.0:50051"
- string [DEFAULT_SERVE_PORT](#) = '[::]:50051'

6.1.1 Function Documentation

6.1.1.1 `dyn_import()`

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

6.1.1.2 `serve()`

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

Initialize and run the BRIARService.

Runs until killed

Returns

:

6.1.2 Variable Documentation

6.1.2.1 `__version__`

```
string __version__ = '1.2.4' [private]
```

6.1.2.2 `DEFAULT_MAX_MESSAGE_SIZE`

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

6.1.2.3 `DEFAULT_PORT`

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

6.1.2.4 DEFAULT_SERVE_PORT

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

6.2 briar.__main__ Namespace Reference

6.3 briar.briar_cli Namespace Reference

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

: qdb

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 ( )
```

6.3.3 Variable Documentation

6.3.3.1 COMMANDS

dictionary COMMANDS

Initial value:

```
1 = {
2     '! annotate' : ['Run detection and annotation models and save the results.',incomplete],
3     'detect' : ['Run detection on media files.',detect],
4     'track' : ['Run tracking on media files (videos only).',track],
5     'enhance' : ['Run tracking on media files (videos only).',enhance],
6     'extract' : ['Run feature extraction to generate templates or embeddings.', extract],
7     'enroll' : ['Scan media files and enroll templates into a database.',enroll],
8     'finalize' : ['Finalize a database.', database_finalize],
9     '! template-fuse' : ['Fuse two templates into a single database entry.', incomplete],
10    'search' : ['Search a database for an example identity.',search],
11    'sigset-stats' : ['Convert a sigset to a csv file and compute statistics.', sigset_stats],
12    'sigset-enroll' : ['Process a sigset and enroll in a database.', sigset_enroll],
13    'status' : ['Connects to the server and displays version and status information.',status],
14    '! test-verify' : ['Compare two databases and create a verification score matrix.',incomplete],
15    '! test-search' : ['Compare two databases and create search results.',incomplete],
16    'database' : ['Base call for database-related functions',briar_database_command_line],
17    'vis' : ['Visualize saved results output from API Calls',viz],
18    'test' : ['Base call for API test functions',briar_test_command_line]
19 }
```

6.3.3.2 DATABASE_COMMANDS

dictionary DATABASE_COMMANDS

Initial value:

```
1 = {
2     '!create' : ['Create and initialize a new database.',incomplete],
3     'delete' : ['Delete a database from the service.',database_delete],
4     'rename' : ['Rename a database from the service.',database_rename],
5     '!insert' : ['Insert templates directly into a database.',incomplete],
6     '!load' : ['Load database onto the server.',incomplete],
7     'list' : ['List the names of the databases on this service.',database_list],
8     'ls' : ['List the names of the databases on this service.',database_list],
9     'info' : ['List information about a given database.', database_info],
10    'list-entries' : ['List the entries contained within a database stored on this
    service.',database_list_entries],
11    '!remove' : ['Remove entries from the database',incomplete],
12    'merge' : ['Merge a list of existing databases together',database_merge],
13 }
```

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

Classes

- class [BriarClient](#)

Provide a client to a BRIAR service.

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

6.5 briar.briar_media Namespace Reference

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

Namespaces

- [annotate](#)
- [connection](#)
- [database](#)
- [detect](#)
- [enhance](#)
- [enroll](#)
- [extract](#)
- [finalize](#)
- [media](#)
- [search](#)
- [sigset](#)
- [status](#)
- [test](#)
- [track](#)
- [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.annotate Namespace Reference

Functions

- def [annotate](#) ()

6.7.1 Function Documentation

6.7.1.1 annotate()

```
def briar.cli.annotate.annotate ( )
```


6.8 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

6.8.1 Function Documentation

6.8.1.1 `addConnectionOptions()`

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

Accumulatively add options for connecting to the Briar API service.

Modifies the parser in place

Parameters

<code>parser</code>	optparse.OptionParser: A parser to modify in place by adding connection options
---------------------	---

6.8.2 Variable Documentation

6.8.2.1 `DEFAULT_MAX_ASYNC`

```
int DEFAULT_MAX_ASYNC = 8
```

6.8.2.2 `DEFAULT_MAX_MESSAGE_SIZE`

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

6.9 briar.cli.database Namespace Reference

Functions

- def [database_delete](#) ()
- def [database_info](#) ()
- def [database_list](#) ()
- def [database_list_entries](#) ()
- def [database_merge](#) ()
- def [database_rename](#) ()
- def [database_retrieve](#) ()
- def [db_no_exist](#) (name)
- def [parseDatabaseDeleteOptions](#) ()
Generate options for Deleting a pre-existing database and parse command line arguments into API call.
- def [parseDatabaseInfoOptions](#) ()
Generate options for getting information about a pre-existing database and parse command line arguments into an API call.
- def [parseDatabaseListEntriesOptions](#) ()
Generate options for Listing entries within a pre-existing database and parse command line arguments into an API call.
- def [parseDatabaseListOptions](#) ()
Generate options for listing all pre-existing databases and parse command line arguments into them.
- def [parseDatabaseMergeOptions](#) ()
Generate options for merging databases and parse command line arguments into the API call.
- def [parseDatabaseRenameOptions](#) ()
Generate options for Renaming a pre-existing database to a new name and parse command line arguments into API call.
- def [parseDatabaseRetrieveOptions](#) ()
Generate options for retrieving a pre-existing database and parse command line arguments API call.

6.9.1 Function Documentation

6.9.1.1 database_delete()

```
def briar.cli.database.database_delete ( )
```

delete a database

6.9.1.2 database_info()

```
def briar.cli.database.database_info ( )
```

list the information pertaining to a database

6.9.1.3 database_list()

```
def briar.cli.database.database_list ( )
```

list the names of the databases

6.9.1.4 database_list_entries()

```
def briar.cli.database.database_list_entries ( )
```

list the entries in a database

6.9.1.5 database_merge()

```
def briar.cli.database.database_merge ( )
```

Merge a set of databases

6.9.1.6 database_rename()

```
def briar.cli.database.database_rename ( )
```

rename a database

6.9.1.7 database_retrieve()

```
def briar.cli.database.database_retrieve ( )
```

list the entries in a database

6.9.1.8 db_no_exist()

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

6.9.1.9 parseDatabaseDeleteOptions()

```
def briar.cli.database.parseDatabaseDeleteOptions ( )
```

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

6.9.1.10 parseDatabaseInfoOptions()

```
def briar.cli.database.parseDatabaseInfoOptions ( )
```

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

6.9.1.11 parseDatabaseListEntriesOptions()

```
def briar.cli.database.parseDatabaseListEntriesOptions ( )
```

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

6.9.1.12 parseDatabaseListOptions()

```
def briar.cli.database.parseDatabaseListOptions ( )
```

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

6.9.1.13 parseDatabaseMergeOptions()

```
def briar.cli.database.parseDatabaseMergeOptions ( )
```

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

6.9.1.14 parseDatabaseRenameOptions()

```
def briar.cli.database.parseDatabaseRenameOptions ( )
```

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

6.9.1.15 parseDatabaseRetrieveOptions()

```
def briar.cli.database.parseDatabaseRetrieveOptions ( )
```

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

6.10 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)
Using the options specified in the command line, runs a detection on the specified files.
- def [detect_file_iter](#) (media_files, detect_options=None, verbose=False, request_start=-1)
Iterates the paths in the media file list, loading them one by one and yielding grpc detect requests.
- def [detect_options2proto](#) (options)
- def [detectParseOptions](#) (inputCommand=None)
Generate options for running detections and parse command line arguments into them.
- def [save_detections](#) (media_file, reply, options, i, modality=None)
- def [tracking_options2proto](#) (options)

Variables

- string [DETECTION_FILE_EXT](#) = ".detection"

6.10.1 Function Documentation

6.10.1.1 addDetectorOptions()

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

Add options for running detections to the parser.

Modifies the parser in place

Parameters

<i>parser</i>	optparse.OptionParser: A parser to modify in place by adding options
---------------	--

6.10.1.2 addTrackingOptions()

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

Add options for running detections to the parser.

Modifies the parser in place

Parameters

<i>parser</i>	optparse.OptionParser: A parser to modify in place by adding options
---------------	--

6.10.1.3 detect()

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

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

6.10.1.4 detect_file_iter()

```
def briar.cli.detect.detect_file_iter (
    media_files,
    detect_options = None,
    verbose = False,
    request_start = -1 )
```

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

Parameters

<i>media_files</i>	list(str): Paths to the media files to enroll from
<i>options</i>	briar_pb2.DetectionOptions: Command line options in protobuf format which control detection functionality

@yield: briar_service_pb2.DetectRequest

6.10.1.5 detect_options2proto()

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

Parse command line options and populate a proto object for grpc

6.10.1.6 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

6.10.1.7 save_detections()

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

6.10.1.8 tracking_options2proto()

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

Parse command line options and populate a proto object for grpc

6.10.2 Variable Documentation

6.10.2.1 DETECTION_FILE_EXT

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

6.11 briar.cli.enhance Namespace Reference

Functions

- def [addEnhanceOptions](#) (parser)
Add options for running detections to the parser.
- def [enhance](#) (options=None, args=None)
Using the options specified in the command line, runs a detection on the specified files.
- def [enhance_file_iter](#) (media_files, enhance_options=None, verbose=False, request_start=-1)
Iterates the paths in the media file list, loading them one by one and yielding grpc detect requests.
- def [enhance_options2proto](#) (options)
- def [enhanceParseOptions](#) (inputCommand=None)
Generate options for running enhancement and parse command line arguments into them.
- def [save_Enhancement](#) (media_file, reply, options, i, modality=None)

6.11.1 Function Documentation

6.11.1.1 addEnhanceOptions()

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

Add options for running detections to the parser.

Modifies the parser in place

Parameters

<i>parser</i>	optparse.OptionParser: A parser to modify in place by adding options
---------------	--

6.11.1.2 enhance()

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

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

6.11.1.3 enhance_file_iter()

```
def briar.cli.enhance.enhance_file_iter (
    media_files,
    enhance_options = None,
    verbose = False,
    request_start = -1 )
```

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

Parameters

<i>media_files</i>	list(str): Paths to the media files to enroll from
<i>options</i>	briar_pb2.DetectionOptions: Command line options in protobuf format which control detection functionality

@yield: briar_service_pb2.DetectRequest

6.11.1.4 enhance_options2proto()

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

Parse command line options and populate a proto object for grpc

6.11.1.5 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

6.11.1.6 save_Enhancement()

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

6.12 briar.cli.enroll Namespace Reference

Functions

- def [addEnrollOptions](#) (parser)
Add options for enrollment into a database.
- def [enroll](#) (options=None, args=None)
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.

6.12.1 Function Documentation

6.12.1.1 addEnrollOptions()

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

Add options for enrollment into a database.

Parameters

<i>parser</i>	optparse.OptionParser: A parser to modify in place by adding options
---------------	--

6.12.1.2 enroll()

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

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

6.12.1.3 enroll_options2proto()

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

6.12.1.4 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

6.13 briar.cli.extract Namespace Reference

Functions

- def [addExtractOptions](#) (parser)
Add options for extractions to the parser.
- def [extract](#) (options=None, args=None)
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 [save_extractions](#) (media_file, templates, options, i, modality=None)

Variables

- string [TEMPLATE_FILE_EXT](#) = '.template'

6.13.1 Function Documentation

6.13.1.1 addExtractOptions()

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

Add options for extractions to the parser.

@type parser: optparse.OptionParser

Parameters

<i>parser</i>	A parser to modify in place by adding options
---------------	---

6.13.1.2 extract()

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

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

6.13.1.3 extract_options2proto()

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

6.13.1.4 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

6.13.1.5 save_extractions()

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

6.13.2 Variable Documentation

6.13.2.1 TEMPLATE_FILE_EXT

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

6.14 briar.cli.finalize Namespace Reference

Functions

- def [database_finalize](#) (options=None, args=None)
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.

6.14.1 Function Documentation

6.14.1.1 database_finalize()

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

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

Returns

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

6.14.1.2 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

6.15 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

6.15.1 Function Documentation

6.15.1.1 addMediaOptions()

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

Add options for running detections to the parser.

Modifies the parser in place

Parameters

<i>parser</i>	optparse.OptionParser: A parser to modify in place by adding options
---------------	--

6.15.1.2 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

<i>args</i>	list(str): List of paths to add as/search for media files
<i>options</i>	optparse.Values: Command line options which dictate collect behavior
<i>extension</i>	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'

6.15.1.3 hasExtension()

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

6.15.2 Variable Documentation**6.15.2.1 DEFAULT_MAX_SIZE**

```
int DEFAULT_MAX_SIZE = 1920
```

6.16 briar.cli.search Namespace Reference

Functions

- def [addSearchOptions](#) (parser)
Add options for search of a database.
- def [search](#) (options=None, args=None)
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.

Variables

- string [MATCHES_FILE_EXT](#) = '.matches'

6.16.1 Function Documentation

6.16.1.1 addSearchOptions()

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

Add options for search of a database.

Parameters

<i>parser</i>	optparse.OptionParser: A parser to modify in place by adding options
---------------	--

6.16.1.2 search()

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

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

6.16.1.3 search_options2proto()

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

Parse command line options and populate a proto object for grpc

6.16.1.4 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

6.16.2 Variable Documentation

6.16.2.1 MATCHES_FILE_EXT

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

6.17 briar.cli.sigset Namespace Reference

Functions

- def [parseSigsetEnrollOptions](#) (inputCommand=None)
- def [parseSigsetStatsOptions](#) (inputCommand=None)
- def [sigset_enroll](#) ()
- def [sigset_stats](#) (options=None, args=None)

6.17.1 Function Documentation

6.17.1.1 parseSigsetEnrollOptions()

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

Parse command line arguments.

6.17.1.2 parseSigsetStatsOptions()

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

Parse command line arguments.

6.17.1.3 sigset_enroll()

```
def briar.cli.sigset.sigset_enroll ( )
```

6.17.1.4 sigset_stats()

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

6.18 briar.cli.status Namespace Reference

Functions

- def [status](#) (options=None, args=None)
Connects to the server and gets status information.
- def [statusParseOptions](#) (inputCommand=None)
Generate options for getting status and parse command line arguments into them.

6.18.1 Function Documentation

6.18.1.1 status()

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

Conects to the server and gets status information.

Print results.

Returns

: None - results are printed

6.18.1.2 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

6.19 briar.cli.test Namespace Reference

Classes

- class [BriarTest](#)
- class [BriarTestResult](#)
- class [DatabaseTest](#)
- class [DetectTest](#)
- class [EnrollTest](#)
- class [ExtractTest](#)

Functions

- def [detection_output_tests](#) (detection_obj_loaded, testimage, return_media)
- def [extraction_output_tests](#) (template_obj_loaded, testimage, return_media)

6.19.1 Function Documentation

6.19.1.1 detection_output_tests()

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

6.19.1.2 extraction_output_tests()

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

6.20 briar.cli.track Namespace Reference

Functions

- def [save_tracklets](#) (media_file, tracklets, options, i, verbose=False, modality=None)
- def [track](#) (options=None, args=None)

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

Variables

- string [TRACKLET_FILE_EXT](#) = ".tracklet"

6.20.1 Function Documentation

6.20.1.1 save_tracklets()

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

6.20.1.2 track()

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

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

6.20.2 Variable Documentation

6.20.2.1 TRACKLET_FILE_EXT

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

6.21 briar.cli.viz Namespace Reference

Functions

- def [viz](#) ()
Using the options specified in the command line, runs a detection on the specified files.
- def [vizParseOptions](#) ()
Generate options for running detections and parse command line arguments into them.

6.21.1 Function Documentation

6.21.1.1 viz()

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

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

6.21.1.2 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

6.22 briar.functions Namespace Reference

Functions

- def [new_uuid](#) ()
Create and return a new, 36 character unique id.

6.22.1 Function Documentation

6.22.1.1 new_uuid()

```
def briar.functions.new_uuid ( )
```

Create and return a new, 36 character unique id.

Returns

: str

6.23 briar.grpc_json Namespace 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.

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](#)

6.23.1 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.

6.23.2 Function Documentation

6.23.2.1 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

<i>obj_dict</i>	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**'

6.23.2.2 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

<i>load_path</i>	str: Path to the json file to load
------------------	------------------------------------

Returns

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

6.23.2.3 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

<i>obj</i>	Any gRPC object generated by protobuf files
------------	---

Returns

: A dictionary representing the object

6.23.2.4 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

<i>json_obj</i>	list dict: List or dict containing data to save
<i>save_path</i>	str: Path to the file to save

Returns: None

6.23.3 Variable Documentation

6.23.3.1 ATTRIB_IGNORE

list ATTRIB_IGNORE

Initial value:

```
1 = ['ByteSize', 'Clear', 'ClearExtension', 'ClearField', 'CopyFrom',
2     'DESCRIPTOR', 'DiscardUnknownFields', 'Extensions',
3     'FindInitializationErrors', 'FromString', 'HasExtension',
4     'HasField', 'IsInitialized', 'ListFields', 'MergeFrom',
5     'MergeFromString', 'ParseFromString', 'RegisterExtension',
6     'SerializePartialToString', 'SerializeToString',
7     'SetInParent', 'UnknownFields', 'WhichOneof',
8     '_CheckCalledFromGeneratedFile', '_SetListener',
9     '__deepcopy__', '__delattr__', '__dir__', '__doc__',
10    '_extensions_by_name', '_extensions_by_number', 'EnumTypeWrapper']
```

6.24 briar.ibriar Namespace Reference

Classes

- class [BriarCLI](#)
- class [BriarCLICompleter](#)
- class [CLICommands](#)

Variables

- [bcl](#) = [BriarCLI](#)()
- dictionary [DEFAULT_ARGS](#) = {"in_path":str, "out_path":str, "": "", }

6.24.1 Variable Documentation

6.24.1.1 bcl

`bcl = BriarCLI()`

6.24.1.2 DEFAULT_ARGS

```
dictionary DEFAULT_ARGS = {"in_path":str, "out_path":str, "":"","", }
```

6.25 briar.media Namespace Reference

Namespaces

- [visualize](#)

Classes

- class [BriarProgress](#)
- class [ImageIterator](#)
- class [VideoIterator](#)

Functions

- def [decodeMedia](#) (media_pb)
Convert protobuf media into a numpy array.
- def [ImageGenerator](#) (filepath, start=None, stop=None, unit=None)

6.25.1 Function Documentation

6.25.1.1 decodeMedia()

```
def briar.media.decodeMedia (
    media_pb )
```

Convert protobuf media into a numpy array.

Parameters

<i>media_pb</i>	briar_pb2.BriarMedia
-----------------	----------------------

return: numpy.array

6.25.1.2 ImageGenerator()

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

6.26 briar.media.visualize Namespace Reference

Functions

- def `visualize_detection` (detection_path)
- def `visualize_matches` (matches_path)
- def `visualize_track` (track_path, options)

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`)]

6.26.1 Function Documentation

6.26.1.1 visualize_detection()

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

6.26.1.2 visualize_matches()

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

6.26.1.3 visualize_track()

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

6.26.2 Variable Documentation

6.26.2.1 fdir

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

6.26.2.2 files

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

6.27 briar.media_converters Namespace Reference

Functions

- def [image_cv2proto](#) (im, compression='uint8', quality=99)
Convert a cv2 numpy array to a protobuf format.
- def [image_np2proto](#) (im, compression='uint8', quality=99)
Convert a numpy array to a protobuf format.
- def [image_proto2cv](#) (pb_data)
Convert a protobuf BriarMedia image to a cv2 numpy array.
- def [image_proto2np](#) (pb_data)
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 [vector_np2proto](#) (vec)
Convert a 1 dimensional np array into a BriarVector.
- def [vector_proto2np](#) (protovec)
Convert a protobuf vector into a numpy array.

Variables

- dictionary [modalityDict](#)
- dictionary [reverseModalityDict](#) = {[modalityDict](#)[k]:k for k in [modalityDict](#)}

6.27.1 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.

6.27.2 Function Documentation

6.27.2.1 image_cv2proto()

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

Convert a cv2 numpy array to a protobuf format.

Parameters

<i>img</i>	numpy.array: array containing the image to convert to BriarMedia
<i>compression</i>	str: What compression to use. Can be 'uint8', 'png', 'jpg'
<i>quality</i>	int: 0-100 How much do you want to mutilate the image in the name of saving memory?

return: briar_pb2.BriarMedia

6.27.2.2 image_np2proto()

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

Convert a numpy array to a protobuf format.

Parameters

<i>img</i>	numpy.array: array containing the image to convert to BriarMedia
<i>compression</i>	str: What compression to use. Can be 'uint8', 'png', 'jpg'
<i>quality</i>	int: 0-100 How much do you want to mutilate the image in the name of saving memory?

return: briar_pb2.BriarMedia

6.27.2.3 image_proto2cv()

```
def briar.media_converters.image_proto2cv (
    pb_data )
```

Convert a protobuf BriarMedia image to a cv2 numpy array.

Parameters

<i>pb_data</i>	briar_pb2.BriarMedia: Protobuf object containing image data
----------------	---

Returns

: numpy.array cv2 formatted np array containing image

6.27.2.4 image_proto2np()

```
def briar.media_converters.image_proto2np (
    pb_data )
```

Convert a protobuf image to a numpy array.

Parameters

<i>pb_data</i>	briar_pb2.BriarMedia: Protobuf object containing image data
----------------	---

Returns

: np.array

6.27.2.5 matrix_np2proto()

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

Convert a numpy matrix into a BriarMatrix.

Parameters

<i>mat</i>	numpy.array: Matrix to convert
------------	--------------------------------

Returns

: briar_pb2.BriarMatrix

6.27.2.6 matrix_proto2np()

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

Convert a protobuf matrix into a numpy matrix.

Parameters

<i>protomat</i>	briar_pb2.BriarMatrix: Protobuf matrix to convert
-----------------	---

Returns

: numpy.array

6.27.2.7 modality_proto2string()

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

6.27.2.8 modality_string2proto()

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

6.27.2.9 vector_np2proto()

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

Convert a 1 dimensional np array into a BriarVector.

Parameters

<i>vec</i>	numpy.array: Numpy array containg vector data
------------	---

Returns

: briar_pb2.BriarVector

6.27.2.10 vector_proto2np()

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

Convert a protobuf vector into a numpy array.

Parameters

<i>protovec</i>	briar_pb2.BriarVector: Protobuf object containing vector info
-----------------	---

Returns

: numpy.array

6.27.3 Variable Documentation

6.27.3.1 modalityDict

```
dictionary modalityDict
```

Initial value:

```

1 =  {'whole_body':briar_pb2.WHOLE_BODY,
2     'wholeBody':briar_pb2.WHOLE_BODY,
3     'wholebody':briar_pb2.WHOLE_BODY,
4     'face':briar_pb2.FACE,
5     'gait':briar_pb2.GAIT,
6     'unspecified':briar_pb2.UNSPECIFIED}

```

6.27.3.2 reverseModalityDict

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

6.28 briar.sigset Namespace Reference**Namespaces**

- [parse](#)

6.29 briar.sigset.parse Namespace Reference**Functions**

- def [expandTree](#) (root, level=0, spaces=3)
- def [parseBriarSigset](#) (filename)

6.29.1 Function Documentation**6.29.1.1 expandTree()**

```

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

```

6.29.1.2 parseBriarSigset()

```

def briar.sigset.parse.parseBriarSigset (
    filename )

```


6.30 briar.timing Namespace Reference

Functions

- def `end_duration` (reply)
- def `generate_progress` (frame_id, media)
- 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"

6.30.1 Function Documentation

6.30.1.1 `end_duration()`

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

6.30.1.2 `generate_progress()`

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

6.30.1.3 `print_duration()`

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

6.30.1.4 `print_durations()`

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

6.30.1.5 `save_durations()`

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

6.30.1.6 `start_duration()`

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

6.30.1.7 `timeElapsed()`

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

6.30.1.8 `timestamp()`

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

6.30.2 Variable Documentation

6.30.2.1 `DURATION_FILE_EXT`

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

Chapter 7

Class Documentation

7.1 BriarCLI Class Reference

Public Member Functions

- `def __init__ (self)`

Private Member Functions

- `def _cmd_line (self)`

Private Attributes

- `_base_prompt_text`
- `_completer`
- `_prompt_hist_path`

Static Private Attributes

- `string _PROMPT_TXT = ">"`

7.1.1 Constructor & Destructor Documentation

7.1.1.1 `__init__()`

```
def __init__ (  
    self )
```

7.1.2 Member Function Documentation

7.1.2.1 `_cmd_line()`

```
def _cmd_line (
    self ) [private]
```

7.1.3 Member Data Documentation

7.1.3.1 `_base_prompt_text`

```
_base_prompt_text [private]
```

7.1.3.2 `_completer`

```
_completer [private]
```

7.1.3.3 `_prompt_hist_path`

```
_prompt_hist_path [private]
```

7.1.3.4 `_PROMPT_TXT`

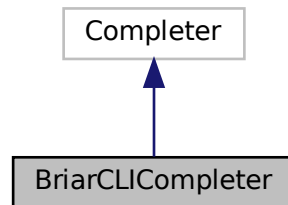
```
string _PROMPT_TXT = ">" [static], [private]
```

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

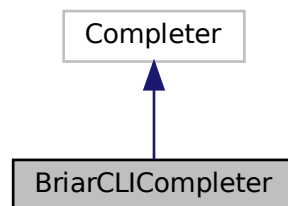
- [ibriar.py](#)

7.2 BriarCLICompleter Class Reference

Inheritance diagram for BriarCLICompleter:



Collaboration diagram for BriarCLICompleter:



Public Member Functions

- def [__init__](#) (self)
- def [get_completions](#) (self, document, complete_event)

Private Member Functions

- def [_complete_base_cmd](#) (self, word_to_complate)
- def [_complete_cmds](#) (self, word_to_complate, base_cmd)
- def [_complete_from_schema](#) (self, cmd_to_cmplt, split_cmds, d)
- def [_complete_kwargs](#) (self, word_to_complete, base_cmd, sub_cmd)
- def [_get_completions](#) (self, document, complete_event, word, suggestions)
- def [_suggest_cmds](#) (self, base_cmd)
- def [_suggest_kwargs](#) (self, base_cmd, sub_cmd)

7.2.1 Constructor & Destructor Documentation

7.2.1.1 `__init__()`

```
def __init__ (
    self )
```

7.2.2 Member Function Documentation

7.2.2.1 `_complete_base_cmd()`

```
def _complete_base_cmd (
    self,
    word_to_complate ) [private]
```

7.2.2.2 `_complete_cmds()`

```
def _complete_cmds (
    self,
    word_to_complate,
    base_cmd ) [private]
```

7.2.2.3 `_complete_from_schema()`

```
def _complete_from_schema (
    self,
    cmd_to_cmplt,
    split_cmds,
    d ) [private]
```

7.2.2.4 `_complete_kwargs()`

```
def _complete_kwargs (
    self,
    word_to_complete,
    base_cmd,
    sub_cmd ) [private]
```

7.2.2.5 `_get_completions()`

```
def _get_completions (
    self,
    document,
    complete_event,
    word,
    suggestions ) [private]
```

7.2.2.6 `_suggest_cmds()`

```
def _suggest_cmds (
    self,
    base_cmd ) [private]
```

7.2.2.7 `_suggest_kwargs()`

```
def _suggest_kwargs (
    self,
    base_cmd,
    sub_cmd ) [private]
```

7.2.2.8 `get_completions()`

```
def get_completions (
    self,
    document,
    complete_event )
```

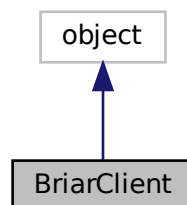
The documentation for this class was generated from the following file:

- [ibriar.py](#)

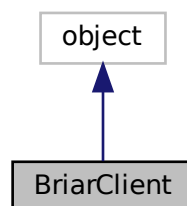
7.3 BriarClient Class Reference

Provide a client to a BRIAR service.

Inheritance diagram for BriarClient:



Collaboration diagram for BriarClient:



Public Member Functions

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

- Run detection on media contained in detect_requests.*

 - def `detect_files` (self, media_file_list, options=None)

Iterator which runs detections on the given file paths, automatically creating and yielding to detect requests initialized from the read images.
 - def `detect_frames` (self, frames, options=None)

Runs detection on an iterable containing pyvision images.
 - def `enhance` (self, enhance_requests, options=None)

Enhancement Functions.
 - def `enroll` (self, enroll_iter)

Enroll images contained in the enroll iterator.
 - def `enroll_file_iter` (self, database_name, media_files, detect_options=None, extract_options=None, enroll_options=None, det_list_list=None, whole_image=False, request_start=-1)

Enroll Functions.
 - def `enroll_files` (self, database, file_list, detect_options=None, extract_options=None, enroll_options=None, det_list_list=None, whole_image=False, request_start=-1)

Iterator which enrolls images from the given file paths, automatically creating and yielding enroll requests initialized from the read images.
 - def `enroll_frames` (self, database_name, frame_list, subject_name=None, subject_id=None, media_id=None, options=None)

Iterator which runs enrolls on the included frames, automatically creating and yielding enroll requests initialized from the read images.
 - def `enroll_frames2` (self, frame_iter, database_name, subject_id, media_id, entity_type, detect_options, extract_options, enroll_options, subject_name=None, record=None)

Iterator which runs enrolls on the included frames, automatically creating and yielding enroll requests initialized from the read images.
 - def `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)

Enroll Functions.
 - def `extract` (self, extract_iter)

Extract images contained in the extract iterator.
 - def `extract_file_iter` (self, media_files, det_list_list=None, detect_options=None, extract_options=None, whole_image=False, request_start=-1)

Extract Functions.
 - def `extract_files` (self, media_list, det_list_list=None, detect_options=None, extract_options=None, whole_image=False, request_start=-1)

Iterator which runs extracts on the given file paths, automatically creating and yielding extract requests initialized from the read images.
 - def `extract_frames` (self, frame_list, det_list_list, whole_image)

Iterator which runs extracts on the given frames, automatically creating and yielding extract requests initialized from the read images.
 - def `extract_frames_iter` (self, frame_list)

Iterates the PyVision frames, yielding extract requests.
 - 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_status` (self, options=None)

Service Functions.
 - def `load_database` (self, database_name)

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

Utility Functions:
 - def `retrieve_req_iter` (self, database_name, template_ids)

Database functions: Retrieve.

- def [search](#) (self, search_iter)

Search Functions.

- def [search_file_iter](#) (self, database_name, media_files, detect_options=None, extract_options=None, search_options=None, search_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 search requests.

- def [search_files](#) (self, database, file_list, detect_options=None, extract_options=None, search_options=None, search_templates=None, det_list_list=None, whole_image=False, request_start=-1)

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

- def [track](#) (self, track_iter)

Tracking Functions.

- def [track_file_iter](#) (self, media_files, detect_options=None, request_start=-1)

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

- def [track_files](#) (self, media_list, detect_options=None, request_start=-1)

Iterator which runs extracts on the given file paths, automatically creating and yielding track requests initialized from the read videos.

- def [verify](#) (self, flag, reference_media=None, verification_media=None, reference_dets=None, verification_dets=None, reference_tmpls=None, verification_tmpls=None)

Verify Functions.

Static Public Member Functions

- def [detect_files_iter](#) (media_file_list)

Detection functions.

- def [detect_frames_iter](#) (frames)

Iterates the PyVision frames, yielding detect requests.

Public Attributes

- [channel](#)
- [options](#)
- [stub](#)

Static Public Attributes

- string [DEFAULT_PORT](#) = "127.0.0.1:50051"

Static Private Member Functions

- def [_enroll_frames_iter](#) (frame_list, database_name, subject_name, subject_id, media_id)

Iterates the PyVision frames, yielding enroll requests.

- def [_enroll_frames_iter2](#) (frame_iter, database_name, subject_id, media_id, detect_options, extract_options, enroll_options, subject_name=None, record=None)

Iterates the PyVision frames, yielding enroll requests.

7.3.1 Detailed Description

Provide a client to a BRIAR service.

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

7.3.2 Constructor & Destructor Documentation

7.3.2.1 `__init__()`

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

Initialize the client and connect it to the specified server.

Attempts a connection to localhost by default

Parameters

<i>options</i>	optparse.Values: Options which define the connection being established
----------------	--

7.3.3 Member Function Documentation

7.3.3.1 `_enroll_frames_iter()`

```
def _enroll_frames_iter (
    frame_list,
    database_name,
    subject_name,
    subject_id,
    media_id ) [static], [private]
```

Iterates the PyVision frames, yielding enroll requests.

Parameters

<i>frame_list</i>	list(pyvision.Image): Raw image data stored in a pyvision object - directly populates requests yielded
<i>database_name</i>	str: Database to enroll int
<i>subject_name</i>	str: Name of subject to enroll
<i>subject_id</i>	str: Uid of subject to enroll
<i>media_id</i>	str: Uid of media being enrolled # TODO media id does nothing

@yield: briar_service_pb2.EnrollRequest

7.3.3.2 _enroll_frames_iter2()

```
def _enroll_frames_iter2 (
    frame_iter,
    database_name,
    subject_id,
    media_id,
    detect_options,
    extract_options,
    enroll_options,
    subject_name = None,
    record = None ) [static], [private]
```

Iterates the PyVision frames, yielding enroll requests.

Parameters

<i>frame_list</i>	list(pyvision.Image): Raw image data stored in a pyvision object - directly populates requests yielded
<i>database_name</i>	str: Database to enroll int
<i>subject_name</i>	str: Name of subject to enroll
<i>subject_id</i>	str: Uid of subject to enroll
<i>media_id</i>	str: Uid of media being enrolled # TODO media id does nothing

@yield: briar_service_pb2.EnrollRequest

7.3.3.3 database_create()

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

Creates an empty database of the given name.

Parameters

<i>database_name</i>	str: Name of database to create
----------------------	---------------------------------

Returns

: briar_pb2.BriarDurations

7.3.3.4 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

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

Returns

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

7.3.3.5 database_list_templates()

```

def database_list_templates (
    self,
    database_name )

```

Lists the templates stored inside the given database name.

Parameters

<i>database_name</i>	str: Name of the database to get the templates from
----------------------	---

Returns

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

7.3.3.6 database_remove_templates()

```

def database_remove_templates (
    self,

```

```

        database_name,
        template_ids )

```

Remove templates matching the ids from the database.

Parameters

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

Returns

: briar_pb2.BriarDurations

7.3.3.7 database_retrieve()

```

def database_retrieve (
    self,
    database_name,
    template_ids )

```

Iteratively grab and return templates matching template_ids from the database.

Parameters

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

Returns

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

7.3.3.8 detect()

```

def detect (
    self,
    detect_requests,
    options = None )

```

Run detection on media contained in detect_requests.

Parameters

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

yield: briar_service_pb2.DetectReply containing results

7.3.3.9 detect_files()

```
def detect_files (
    self,
    media_file_list,
    options = None )
```

Iterator which runs detections on the given file paths, automatically creating and yielding to detect requests initialized from the read images.

Parameters

<i>media_file_list</i>	list(str): List of paths to the image and video files to detect on.
	optparse.Values
<i>options</i>	Additional options to feed to control the detect functions

yield: briar_service_pb2.DetectReply containing results

7.3.3.10 detect_files_iter()

```
def detect_files_iter (
    media_file_list ) [static]
```

Detection functions.

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

Parameters

<i>media_file_list</i>	list of strings @yield: briar_service_pb2.DetectRequest
------------------------	---

7.3.3.11 detect_frames()

```
def detect_frames (
    self,
    frames,
    options = None )
```

Runs detection on an iterable containing pyvision images.

Parameters

<i>frames</i>	pyvision.Video list(pyvision.Image): Iterable which yields pyvision.Images to run detections on
<i>options</i>	optparse.Values
<i>options</i>	Additional options to feed to control the detect functions

yield: briar_service_pb2.DetectReply containing results

7.3.3.12 detect_frames_iter()

```
def detect_frames_iter (
    frames ) [static]
```

Iterates the PyVision frames, yielding detect requests.

Parameters

<i>frames</i>	list(pyvision.Image): Raw image data stored in a pyvision object - directly populates yield requests
---------------	--

@yield: briar_service_pb2.DetectRequest

7.3.3.13 enhance()

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

Enhancement Functions.

Run enhancement on media contained in enhance_requests.

Parameters

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

yield: briar_service_pb2.EnhanceReply containing results

7.3.3.14 enroll()

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

Enroll images contained in the enroll iterator.

Parameters

<i>enroll_iter</i>	Generator: Generator object which yields enroll requests
--------------------	--

Returns

: briar_service_pb2.ExtractReply

7.3.3.15 enroll_file_iter()

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

Enroll Functions.

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

Parameters

<i>database_name</i>	str: Name of the database to enroll templates in
<i>media_files</i>	list(str): Paths to the media files to enroll from
<i>options</i>	briar_pb2.DetectionOptions: Command line options in protobuf format which control detection functionality
<i>options</i>	briar_pb2.ExtractOptions: Command line options in protobuf format which control extraction functionality
<i>options</i>	briar_pb2.EnrollOptions: Command line options in protobuf format which control enrollment functionality
<i>det_list_list</i>	list(list(briar_pb2.Detection)): If not None, it will contains 1 list of detections per media file
<i>boolean</i>	Ignore detections and run an extract on the whole image

@yield: briar_service_pb2.EnrollRequest

7.3.3.16 enroll_files()

```
def enroll_files (
    self,
    database,
    file_list,
    detect_options = None,
    extract_options = None,
    enroll_options = None,
    det_list_list = None,
    whole_image = False,
    request_start = -1 )
```

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

Parameters

<i>database</i>	str: Specifies which database to enroll in
<i>file_list</i>	list(str): List of paths to the image and video files to extract on.
<i>options</i>	optparse.Values: Command line options which control enrollment functionality
<i>det_list_list</i>	list(list(briar_pb2.Detection)): If not None, it will contains 1 list of detections per media file
<i>boolean</i>	Ignore detections and run an extract on the whole image

yield: briar_service_pb2.ExtractReply containing results

7.3.3.17 enroll_frames()

```
def enroll_frames (
    self,
    database_name,
    frame_list,
    subject_name = None,
    subject_id = None,
    media_id = None,
    options = None )
```

Iterator which runs enrolls on the included frames, automatically creating and yielding enroll requests initialized from the read images.

Parameters

<i>database_name</i>	str: Name of the database to enroll into
<i>frame_list</i>	list(pyvision.Image) or other iterable which yields pyvision.Image: List containing data to extract on.
<i>subject_name</i>	str: Name of subject to enroll
<i>subject_id</i>	str: Uid of the subject to enroll
<i>media_id</i>	str: Uid of the media
<i>options</i>	optparse.Values: Command line options which control enrollment functionality

yield: briar_service_pb2.EnrollReply containing results

7.3.3.18 enroll_frames2()

```
def enroll_frames2 (
    self,
    frame_iter,
    database_name,
    subject_id,
    media_id,
    entity_type,
    detect_options,
    extract_options,
    enroll_options,
    subject_name = None,
    record = None )
```

Iterator which runs enrolls on the included frames, automatically creating and yielding enroll requests initialized from the read images.

Parameters

<i>database_name</i>	str: Name of the database to enroll into
<i>frame_iter</i>	list(pyvision.Image) or other iterable which yields pyvision.Image: List containing data to extract on.
<i>subject_name</i>	str: Name of subject to enroll
<i>subject_id</i>	str: Uid of the subject to enroll
<i>media_id</i>	str: Uid of the media
<i>options</i>	optparse.Values: Command line options which control enrollment functionality

yield: briar_service_pb2.EnrollReply containing results

7.3.3.19 enroll_frames_iter()

```
def 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 )
```

Enroll Functions.

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

@type database_name: str

Parameters

<i>database_name</i>	Name of the database to enroll templates in
----------------------	---

@type video: an iterator that generates cv2 frames

Parameters

<i>media_files</i>	Paths to the media files to enroll from
--------------------	---

@type detect_options: briar_pb2.DetectionOptions

Parameters

<i>options</i>	Command line options in protobuf format which control detection functionality
----------------	---

@type options: briar_pb2.ExtractOptions

Parameters

<i>options</i>	Command line options in protobuf format which control extraction functionality
----------------	--

@type options: briar_pb2.EnrollOptions

Parameters

<i>options</i>	Command line options in protobuf format which control enrollment functionality
----------------	--

@type det_list_list: List of list of briar_pb2.Detection

Parameters

<i>det_list_list</i>	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

7.3.3.20 extract()

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

Extract images contained in the extract iterator.

Parameters

<i>extract_iter</i>	Generator: Generator object which yields extract requests
---------------------	---

Returns

: briar_service_pb2.ExtractReply

7.3.3.21 extract_file_iter()

```
def extract_file_iter (
    self,
    media_files,
```

```

    det_list_list = None,
    detect_options = None,
    extract_options = None,
    whole_image = False,
    request_start = -1 )

```

Extract Functions.

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

Parameters

<i>media_file_list</i>	list of strings @yield: briar_service_pb2.ExtractRequest
------------------------	--

7.3.3.22 extract_files()

```

def extract_files (
    self,
    media_list,
    det_list_list = None,
    detect_options = None,
    extract_options = None,
    whole_image = False,
    request_start = -1 )

```

Iterator which runs extracts on the given file paths, automatically creating and yielding extract requests initialized from the read images.

Parameters

<i>media_list</i>	list(str): List of paths to the image and video files to extract on.
<i>det_list_list</i>	list(list(briar_pb2.Detection)): If not None, it will contains 1 list of detections per media file
<i>boolean</i>	Ignore detections and run an extract on the whole image

yield: briar_service_pb2.ExtractReply containing results

7.3.3.23 extract_frames()

```

def extract_frames (
    self,
    frame_list,
    det_list_list,
    whole_image )

```

Iterator which runs extracts on the given frames, automatically creating and yielding extract requests initialized from the read images.

Parameters

<i>frame_list</i>	list(pyvision.Image): List containing data to extract on.
<i>det_list_list</i>	list(list(briar_pb2.Detection)): If not None, it will contains 1 list of detections per media file
<i>boolean</i>	Ignore detections and run an extract on the whole image

yield: briar_service_pb2.ExtractReply containing results

7.3.3.24 extract_frames_iter()

```
def extract_frames_iter (
    self,
    frame_list )
```

Iterates the PyVision frames, yielding extract requests.

Parameters

<i>frame_list</i>	list(pyvision.Image): Raw image data stored in a pyvision object - directly populates requests yielded
-------------------	--

@yield: briar_service_pb2.ExtractRequest

7.3.3.25 finalize()

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

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

Parameters

<i>database_name</i>	str: Name of the database to write to disk
----------------------	--

Returns

: briar_pb2.durations

7.3.3.26 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

: 2 element Tuple (List of str, briar_pb2.Durations)

7.3.3.27 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

<i>options</i>	optparse.↵ Values:
----------------	-----------------------

Returns

: 5 element Tuple of str

7.3.3.28 load_database()

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

Database functions: Load/Create.

Load the database from disk into memory

Parameters

<i>database_name</i>	str: Name of database to load
----------------------	-------------------------------

Returns

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

7.3.3.29 print_verbose()

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

Utility Functions:

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

Parameters

<i>args</i>	tuple(object): Arguments to get passed to print
-------------	---

Returns

: None - outputs to screen

7.3.3.30 retrieve_req_iter()

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

Database functions: Retrieve.

Generator which yields retrieve requests

Parameters

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

Returns

: briar_service_pb2.DatabaseRetrieveRequest

7.3.3.31 search()

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

Search Functions.

Given a probe, search the database and return matches

Parameters

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

Returns

: briar_service_pb2.SearchReply

7.3.3.32 search_file_iter()

```
def search_file_iter (
    self,
    database_name,
    media_files,
    detect_options = None,
    extract_options = None,
    search_options = None,
    search_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 search requests.

@type database_name: str

Parameters

<i>database_name</i>	Name of the database to enroll templates in
----------------------	---

@type media_files: List of strings

Parameters

<i>media_files</i>	Paths to the media files to enroll from
--------------------	---

@type detect_options: briar_pb2.DetectionOptions

Parameters

<i>options</i>	Command line options in protobuf format which control detection functionality
----------------	---

@type extract_options: briar_pb2.ExtractOptions

Parameters

<i>options</i>	Command line options in protobuf format which control extraction functionality
----------------	--

@type earch_optons: briar_pb2.SearchOptions

Parameters

<i>options</i>	Command line options in protobuf format which control search functionality
----------------	--

@type det_list_list: List of list of briar_pb2.Detection

Parameters

<i>det_list_list</i>	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

7.3.3.33 search_files()

```
def search_files (
    self,
    database,
    file_list,
    detect_options = None,
    extract_options = None,
    search_options = None,
    search_templates = None,
    det_list_list = None,
    whole_image = False,
    request_start = -1 )
```

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

@type database: str

Parameters

<i>database</i>	Specifies which database to enroll in
-----------------	---------------------------------------

@type file_list: List of str

Parameters

<i>file_list</i>	List of paths to the image and video files to extract on.
------------------	---

@type options: optparse.Values

Parameters

<i>options</i>	Command line options which control enrollment functionality
----------------	---

@type det_list_list: List of list of briar_pb2.Detection

Parameters

<i>det_list_list</i>	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.ExtractReply containing results

7.3.3.34 track()

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

Tracking Functions.

Track person instances contained in the tracking iterator

Parameters

<i>extract_iter</i>	Generator: Generator object which yields extract requests
---------------------	---

Returns

: briar_service_pb2.ExtractReply

7.3.3.35 track_file_iter()

```
def track_file_iter (
    self,
    media_files,
    detect_options = None,
    request_start = -1 )
```

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

Parameters

<i>media_file_list</i>	list of strings @yield: briar_service_pb2.ExtractRequest
------------------------	--

7.3.3.36 track_files()

```
def track_files (
    self,
    media_list,
    detect_options = None,
    request_start = -1 )
```

Iterator which runs extracts on the given file paths, automatically creating and yielding track requests initialized from the read videos.

Parameters

<i>media_list</i>	list(str): List of paths to the video files to extract on.
<i>detect_options</i>	briar_grpc.DetectionOptions: Detection options proto message (contains TrackOptions

yield: briar_service_pb2.TrackReply containing results

7.3.3.37 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

<i>flag</i>	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
<i>reference_media</i>	briar_pb2.BriarMedia: This media is where the the probe templates will be pulled from
<i>verification_media</i>	briar_pb2.BriarMedia: This media is where the comparison templates will be pulled from
<i>reference_dets</i>	List of briar_pb2.Detection: These are the detections to extract probe templates from
<i>verification_dets</i>	List of briar_pb2.Detection: These are the detections to extract comparison templates from
<i>reference_tmpls</i>	list(briar_pb2.Template): Probe templates
<i>verification_tmpls</i>	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,

7.3.4 Member Data Documentation

7.3.4.1 channel

channel

7.3.4.2 DEFAULT_PORT

```
string DEFAULT_PORT = "127.0.0.1:50051" [static]
```

7.3.4.3 options

options

7.3.4.4 stub

stub

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

- [briar_client.py](#)

7.4 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']

7.4.1 Constructor & Destructor Documentation

7.4.1.1 `__init__()`

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

7.4.2 Member Data Documentation

7.4.2.1 `channels`

`channels`

7.4.2.2 `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)
```

7.4.2.3 `datetime`

`datetime`

7.4.2.4 description

description

7.4.2.5 fps

fps

7.4.2.6 height

height

7.4.2.7 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']
```

7.4.2.8 len

len

7.4.2.9 metadata

metadata

7.4.2.10 source

source

7.4.2.11 VIDEO_FORMATS

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

7.4.2.12 width

width

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

- [briar_media.py](#)

7.5 BriarProgress Class Reference

Public Member Functions

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

Public Attributes

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

7.5.1 Constructor & Destructor Documentation

7.5.1.1 `__init__()`

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

7.5.2 Member Function Documentation

7.5.2.1 update()

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

7.5.3 Member Data Documentation

7.5.3.1 desc

```
desc
```

7.5.3.2 enabled

```
enabled
```

7.5.3.3 leave

```
leave
```

7.5.3.4 name

```
name
```

7.5.3.5 pbar

```
pbar
```

7.5.3.6 position

position

7.5.3.7 prevstep

prevstep

7.5.3.8 tqdm

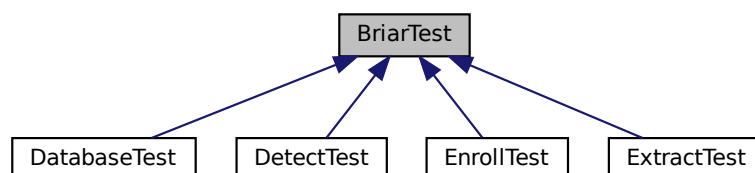
tqdm

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

- [media/___init__.py](#)

7.6 BriarTest Class Reference

Inheritance diagram for BriarTest:



Public Member Functions

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

Public Attributes

- `testdata_folder`

7.6.1 Constructor & Destructor Documentation

7.6.1.1 `__init__()`

```
def __init__ (
    self )
```

7.6.2 Member Function Documentation

7.6.2.1 `description()`

```
def description (
    self )
```

Reimplemented in [ExtractTest](#), and [DetectTest](#).

7.6.2.2 `run()`

```
def run (
    self )
```

7.6.2.3 `test()`

```
def test (
    self )
```

Reimplemented in [DatabaseTest](#), and [EnrollTest](#).

7.6.3 Member Data Documentation

7.6.3.1 `testdata_folder`

```
testdata_folder
```

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

- [cli/test.py](#)

7.7 BriarTestResult Class Reference

Public Member Functions

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

Public Attributes

- `level`
- `name`
- `passed`
- `reason`

7.7.1 Constructor & Destructor Documentation

7.7.1.1 `__init__()`

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

7.7.2 Member Data Documentation

7.7.2.1 `level`

`level`

7.7.2.2 `name`

`name`

7.7.2.3 `passed`

`passed`

7.7.2.4 reason

reason

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

- [cli/test.py](#)

7.8 CLICommands Class Reference

Static Public Attributes

- string [ARGS](#) = "ARGS"
- dictionary [CMD_SCHEMA](#) = {**[DETECT](#), **[EXTRACT](#), **[ENROLL](#)}
- list [CMD_SYNTAX](#) = [[MODE](#), [COMMAND](#), [ARGS](#)]
- string [COMMAND](#) = "COMMAND"
- dictionary [DETECT](#)
- dictionary [ENROLL](#)
- dictionary [EXTRACT](#)
- string [MODE](#) = "MODE"

7.8.1 Member Data Documentation

7.8.1.1 ARGS

```
string ARGS = "ARGS" [static]
```

7.8.1.2 CMD_SCHEMA

```
dictionary CMD_SCHEMA = {**DETECT, **EXTRACT, **ENROLL} [static]
```

7.8.1.3 CMD_SYNTAX

```
list CMD_SYNTAX = [MODE, COMMAND, ARGS] [static]
```

7.8.1.4 COMMAND

```
string COMMAND = "COMMAND" [static]
```

7.8.1.5 DETECT

```
dictionary DETECT [static]
```

Initial value:

```
= {  
    "detect":  
    {  
        "run": {"args": ["input", "output"],  
                 "kwargs": ["kwarg1", "kwarg2", "asdf", "foo", "bar"]}  
    }  
}
```

7.8.1.6 ENROLL

```
dictionary ENROLL [static]
```

Initial value:

```
= {  
    "enroll":  
    {  
        "run": {"args": ["input", "output"],  
                 "kwargs": ["kwarg1", "kwarg2", "asdf", "foo", "bar"]}  
    }  
}
```

7.8.1.7 EXTRACT

```
dictionary EXTRACT [static]
```

Initial value:

```
= {  
    "extract":  
    {  
        "run": {"args": ["input", "output"],  
                 "kwargs": ["kwarg1", "kwarg2", "asdf", "foo", "bar"]}  
    }  
}
```

7.8.1.8 MODE

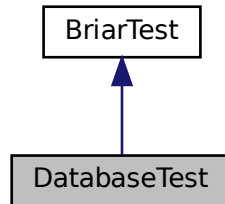
```
string MODE = "MODE" [static]
```

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

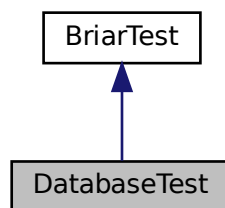
- [ibriar.py](#)

7.9 DatabaseTest Class Reference

Inheritance diagram for DatabaseTest:



Collaboration diagram for DatabaseTest:



Public Member Functions

- def [test](#) (self)

Additional Inherited Members

7.9.1 Member Function Documentation

7.9.1.1 test()

```
def test (
    self )
```

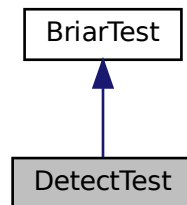
Reimplemented from [BriarTest](#).

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

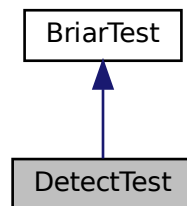
- cli/[test.py](#)

7.10 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"`

7.10.1 Member Function Documentation

7.10.1.1 description()

```
def description (
    self )
```

Reimplemented from [BriarTest](#).

7.10.1.2 test_1_detection_image()

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

7.10.1.3 test_2_detection_image_output()

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

7.10.1.4 test_3_detection_image_withreturn()

```
def test_3_detection_image_withreturn (
    self )
```

7.10.1.5 test_4_detection_image_output_withreturn()

```
def test_4_detection_image_output_withreturn (
    self )
```

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 testim_path

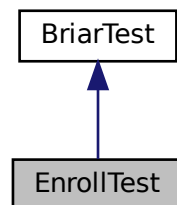
```
string testim_path = "testdata/BTS1/distractors/G00038/controlled/images_jpg/face/G00038_↵  
set2_face0_03_45_662fb70a.jpg" [static]
```

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

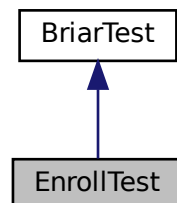
- cli/[test.py](#)

7.11 EnrollTest Class Reference

Inheritance diagram for EnrollTest:



Collaboration diagram for EnrollTest:



Public Member Functions

- def [test](#) (self)

Additional Inherited Members

7.11.1 Member Function Documentation

7.11.1.1 test()

```
def test (
    self )
```

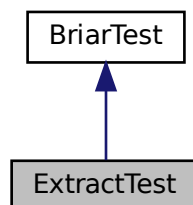
Reimplemented from [BriarTest](#).

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

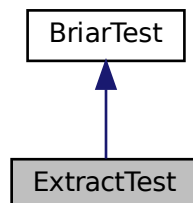
- cli/[test.py](#)

7.12 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"`

7.12.1 Member Function Documentation

7.12.1.1 [description\(\)](#)

```
def description (  
    self )
```

Reimplemented from [BriarTest](#).

7.12.1.2 [test_1_extraction_image\(\)](#)

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

7.12.1.3 [test_2_extraction_image_output\(\)](#)

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

7.12.2 Member Data Documentation

7.12.2.1 detection_file_path

detection_file_path

7.12.2.2 output_path

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

7.12.2.3 template_file_path

template_file_path

7.12.2.4 testim_path

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

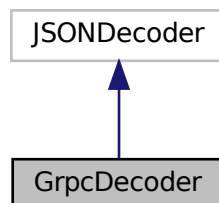
The documentation for this class was generated from the following file:

- cli/[test.py](#)

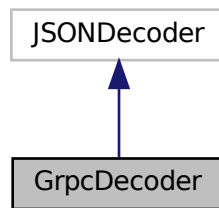
7.13 GrpcDecoder Class Reference

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`

7.13.1 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

7.13.2 Constructor & Destructor Documentation

7.13.2.1 `__init__()`

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

7.13.3 Member Function Documentation

7.13.3.1 `default()`

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

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

Parameters

<i>obj</i>	object dict: Dictionary which represents an object.
------------	--

Returns

: object

7.13.4 Member Data Documentation

7.13.4.1 options

`options`

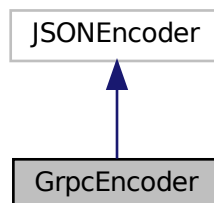
The documentation for this class was generated from the following file:

- [grpc_json.py](#)

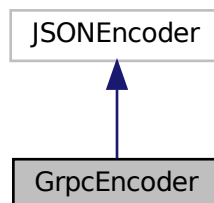
7.14 GrpcEncoder Class Reference

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`

7.14.1 Detailed Description

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

Inherits from `json.JSONEncoder`

7.14.2 Constructor & Destructor Documentation

7.14.2.1 `__init__()`

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

7.14.3 Member Function Documentation

7.14.3.1 `default()`

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

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

Parameters

<i>obj</i>	Any: General object to convert to a json-serializable object
------------	--

Returns

: Json-Serializable object

7.14.4 Member Data Documentation

7.14.4.1 options

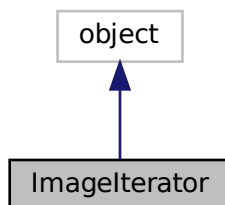
`options`

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

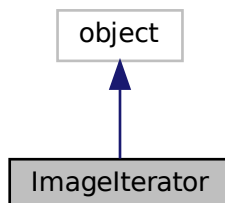
- [grpc_json.py](#)

7.15 Imageliterator Class Reference

Inheritance diagram for Imageliterator:



Collaboration diagram for Imageliterator:



Public Member Functions

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

Public Attributes

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

7.15.1 Constructor & Destructor Documentation

7.15.1.1 `__init__()`

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

7.15.2 Member Function Documentation

7.15.2.1 `__iter__()`

```
def __iter__ (
    self )
```

7.15.2.2 `__len__()`

```
def __len__ (
    self )
```

7.15.2.3 `__next__()`

```
def __next__ (
    self )
```

7.15.3 Member Data Documentation

7.15.3.1 `filepath`

`filepath`

7.15.3.2 `fps`

`fps`

7.15.3.3 `frame`

`frame`

7.15.3.4 `frame_count`

`frame_count`

7.15.3.5 `frame_height`

`frame_height`

7.15.3.6 frame_width

frame_width

7.15.3.7 i

i

7.15.3.8 isOpened

isOpened

7.15.3.9 length

length

7.15.3.10 msec

msec

7.15.3.11 pos

pos

7.15.3.12 processed

processed

7.15.3.13 start_frame

start_frame

7.15.3.14 stop_frame

stop_frame

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

- [media/___init__.py](#)

7.16 MediaStream Class Reference

Public Member Functions

- [def ___init___](#) (self, briar_media)
- [def ___iter___](#) (self, request_type)

Private Attributes

- [_media_list](#)

7.16.1 Constructor & Destructor Documentation

7.16.1.1 ___init__()

```
def ___init___ (
    self,
    briar_media )
```

7.16.2 Member Function Documentation

7.16.2.1 ___iter__()

```
def ___iter___ (
    self,
    request_type )
```

7.16.3 Member Data Documentation

7.16.3.1 `_media_list`

```
_media_list [private]
```

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

- [briar_media.py](#)

7.17 Rect Class Reference

Public Member Functions

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

Public Attributes

- `height`
- `width`
- `x`
- `y`

7.17.1 Detailed Description

Basic rectangle for storing ROIs without needing to mess with the gRPC BriarRect

7.17.2 Constructor & Destructor Documentation

7.17.2.1 `__init__()`

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

7.17.3 Member Data Documentation

7.17.3.1 height

height

7.17.3.2 width

width

7.17.3.3 x

x

7.17.3.4 y

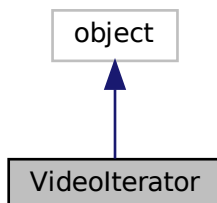
y

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

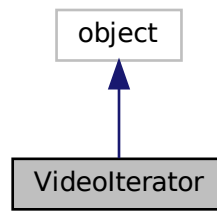
- [__init__.py](#)

7.18 Videolterator Class Reference

Inheritance diagram for Videolterator:



Collaboration diagram for Videolterator:



Public Member Functions

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

Public Attributes

- [cap](#)
- [filepath](#)
- [fps](#)
- [frame_count](#)
- [frame_height](#)
- [frame_width](#)
- [i](#)
- [isOpened](#)
- [length](#)
- [msec](#)
- [pos](#)
- [processed](#)
- [start_frame](#)
- [stop_frame](#)

7.18.1 Constructor & Destructor Documentation

7.18.1.1 `__init__()`

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

7.18.2 Member Function Documentation

7.18.2.1 `__iter__()`

```
def __iter__ (
    self )
```

7.18.2.2 `__len__()`

```
def __len__ (
    self )
```

7.18.2.3 `__next__()`

```
def __next__ (
    self )
```

7.18.3 Member Data Documentation

7.18.3.1 `cap`

`cap`

7.18.3.2 `filepath`

`filepath`

7.18.3.3 `fps`

`fps`

7.18.3.4 frame_count

frame_count

7.18.3.5 frame_height

frame_height

7.18.3.6 frame_width

frame_width

7.18.3.7 i

i

7.18.3.8 isOpened

isOpened

7.18.3.9 length

length

7.18.3.10 msec

msec

7.18.3.11 pos

pos

7.18.3.12 processed

`processed`

7.18.3.13 start_frame

`start_frame`

7.18.3.14 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 [dyn_import](#) (name)
- def [serve](#) (serviceClass, options=None, serve_port=None)

Initialize and run the BRIARService.

Variables

- string [__version__](#) = '1.2.4'
- int [DEFAULT_MAX_MESSAGE_SIZE](#) = 64*1024*1024
- string [DEFAULT_PORT](#) = "0.0.0.0:50051"
- string [DEFAULT_SERVE_PORT](#) = ':::50051'

8.2 `cli/__init__.py` File Reference

Namespaces

- [briar.cli](#)

8.3 media/___init___py File Reference

Classes

- class [BriarProgress](#)
- class [ImageIterator](#)
- class [VideoIterator](#)

Namespaces

- [briar.media](#)

Functions

- def [decodeMedia](#) (media_pb)
Convert protobuf media into a numpy array.
- def [ImageGenerator](#) (filepath, start=None, stop=None, unit=None)

8.4 sigset/___init___py File Reference

Namespaces

- [briar.sigset](#)

8.5 timing/___init___py File Reference

Namespaces

- [briar.timing](#)

Functions

- def [end_duration](#) (reply)
- def [generate_progress](#) (frame_id, media)
- 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"

8.6 __main__.py File Reference

Namespaces

- [briar.__main__](#)

8.7 briar_cli.py File Reference

Namespaces

- [briar.briar_cli](#)

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'

8.8 briar_client.py File Reference

Classes

- class [BriarClient](#)
Provide a client to a BRIAR service.

Namespaces

- [briar.briar_client](#)

8.9 briar_media.py File Reference

Classes

- class [BriarMedia](#)
- class [MediaStream](#)

Namespaces

- [briar.briar_media](#)

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)

8.10 cli/annotate.py File Reference

Namespaces

- [briar.cli.annotate](#)

Functions

- def [annotate](#) ()

8.11 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

8.12 cli/database.py File Reference

Namespaces

- [briar.cli.database](#)

Functions

- def [database_delete](#) ()
- def [database_info](#) ()
- def [database_list](#) ()
- def [database_list_entries](#) ()
- def [database_merge](#) ()
- def [database_rename](#) ()
- def [database_retrieve](#) ()
- def [db_no_exist](#) (name)
- def [parseDatabaseDeleteOptions](#) ()
Generate options for Deleting a pre-existing database and parse command line arguments into API call.
- def [parseDatabaseInfoOptions](#) ()
Generate options for getting information about a pre-existing database and parse command line arguments into an API call.
- def [parseDatabaseListEntriesOptions](#) ()
Generate options for Listing entries within a pre-existing database and parse command line arguments into an API call.
- def [parseDatabaseListOptions](#) ()
Generate options for listing all pre-existing databases and parse command line arguments into them.
- def [parseDatabaseMergeOptions](#) ()
Generate options for merging databases and parse command line arguments into the API call.
- def [parseDatabaseRenameOptions](#) ()
Generate options for Renaming a pre-existing database to a new name and parse command line arguments into API call.
- def [parseDatabaseRetrieveOptions](#) ()
Generate options for retrieving a pre-existing database and parse command line arguments API call.

8.13 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)
Using the options specified in the command line, runs a detection on the specified files.
- def [detect_file_iter](#) (media_files, detect_options=None, verbose=False, request_start=-1)
Iterates the paths in the media file list, loading them one by one and yielding grpc detect requests.
- def [detect_options2proto](#) (options)
- def [detectParseOptions](#) (inputCommand=None)
Generate options for running detections and parse command line arguments into them.
- def [save_detections](#) (media_file, reply, options, i, modality=None)
- def [tracking_options2proto](#) (options)

Variables

- string [DETECTION_FILE_EXT](#) = ".detection"

8.14 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)
Using the options specified in the command line, runs a detection on the specified files.
- def [enhance_file_iter](#) (media_files, enhance_options=None, verbose=False, request_start=-1)
Iterates the paths in the media file list, loading them one by one and yielding grpc detect requests.
- def [enhance_options2proto](#) (options)
- def [enhanceParseOptions](#) (inputCommand=None)
Generate options for running enhancement and parse command line arguments into them.
- def [save_Enhancement](#) (media_file, reply, options, i, modality=None)

8.15 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)
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.

8.16 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)
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 [save_extractions](#) (media_file, templates, options, i, modality=None)

Variables

- string [TEMPLATE_FILE_EXT](#) = '.template'

8.17 cli/finalize.py File Reference

Namespaces

- [briar.cli.finalize](#)

Functions

- def [database_finalize](#) (options=None, args=None)
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.

8.18 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

8.19 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)
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.

Variables

- string [MATCHES_FILE_EXT](#) = '.matches'

8.20 cli/sigset.py File Reference

Namespaces

- [briar.cli.sigset](#)

Functions

- def [parseSigsetEnrollOptions](#) (inputCommand=None)
- def [parseSigsetStatsOptions](#) (inputCommand=None)
- def [sigset_enroll](#) ()
- def [sigset_stats](#) (options=None, args=None)

8.21 cli/status.py File Reference

Namespaces

- [briar.cli.status](#)

Functions

- def [status](#) (options=None, args=None)
Connects to the server and gets status information.
- def [statusParseOptions](#) (inputCommand=None)
Generate options for getting status and parse command line arguments into them.

8.22 cli/test.py File Reference

Classes

- class [BriarTest](#)
- class [BriarTestResult](#)
- class [DatabaseTest](#)
- class [DetectTest](#)
- class [EnrollTest](#)
- class [ExtractTest](#)

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)

8.23 cli/track.py File Reference

Namespaces

- [briar.cli.track](#)

Functions

- def [save_tracklets](#) (media_file, tracklets, options, i, verbose=False, modality=None)
- def [track](#) (options=None, args=None)

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

Variables

- string [TRACKLET_FILE_EXT](#) = ".tracklet"

8.24 cli/viz.py File Reference

Namespaces

- [briar.cli.viz](#)

Functions

- def [viz](#) ()
Using the options specified in the command line, runs a detection on the specified files.
- def [vizParseOptions](#) ()
Generate options for running detections and parse command line arguments into them.

8.25 functions.py File Reference

Namespaces

- [briar.functions](#)

Functions

- def [new_uuid](#) ()
Create and return a new, 36 character unique id.

8.26 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](#)

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](#)

8.27 ibriar.py File Reference

Classes

- class [BriarCLI](#)
- class [BriarCLICompleter](#)
- class [CLICommands](#)

Namespaces

- [briar.ibriar](#)

Variables

- [bcl](#) = [BriarCLI](#)()
- dictionary [DEFAULT_ARGS](#) = {"in_path":str, "out_path":str, "":"", }

8.28 media/visualize.py File Reference

Namespaces

- [briar.media.visualize](#)

Functions

- def [visualize_detection](#) (detection_path)
- def [visualize_matches](#) (matches_path)
- def [visualize_track](#) (track_path, options)

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)]

8.29 media_converters.py File Reference

Namespaces

- [briar.media_converters](#)

Functions

- def [image_cv2proto](#) (im, compression='uint8', quality=99)
Convert a cv2 numpy array to a protobuf format.
- def [image_np2proto](#) (im, compression='uint8', quality=99)
Convert a numpy array to a protobuf format.
- def [image_proto2cv](#) (pb_data)
Convert a protobuf BriarMedia image to a cv2 numpy array.
- def [image_proto2np](#) (pb_data)
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 [vector_np2proto](#) (vec)
Convert a 1 dimensional np array into a BriarVector.
- def [vector_proto2np](#) (protovec)
Convert a protobuf vector into a numpy array.

Variables

- dictionary [modalityDict](#)
- dictionary [reverseModalityDict](#) = {modalityDict[k]:k for k in modalityDict}

8.30 readme-cli.md File Reference

8.31 sigset/parse.py File Reference

Namespaces

- [briar.sigset.parse](#)

Functions

- def [expandTree](#) (root, level=0, spaces=3)
- def [parseBriarSigset](#) (filename)

Index

- `_PROMPT_TXT`
 - BriarCLI, [54](#)
- `__init__`
 - BriarCLI, [53](#)
 - BriarCLICompleter, [56](#)
 - BriarClient, [61](#)
 - BriarMedia, [81](#)
 - BriarProgress, [83](#)
 - BriarTest, [86](#)
 - BriarTestResult, [87](#)
 - GrpcDecoder, [97](#)
 - GrpcEncoder, [99](#)
 - Imageliterator, [101](#)
 - MediaStream, [104](#)
 - Rect, [105](#)
 - Videoliterator, [107](#)
- `__init__.py`, [111](#)
- `__iter__`
 - Imageliterator, [101](#)
 - MediaStream, [104](#)
 - Videoliterator, [108](#)
- `__len__`
 - Imageliterator, [101](#)
 - Videoliterator, [108](#)
- `__main__.py`, [113](#)
- `__next__`
 - Imageliterator, [102](#)
 - Videoliterator, [108](#)
- `__version__`
 - briar, [12](#)
- `_base_prompt_text`
 - BriarCLI, [54](#)
- `_cmd_line`
 - BriarCLI, [54](#)
- `_complete_base_cmd`
 - BriarCLICompleter, [56](#)
- `_complete_cmds`
 - BriarCLICompleter, [56](#)
- `_complete_from_schema`
 - BriarCLICompleter, [56](#)
- `_complete_kwargs`
 - BriarCLICompleter, [56](#)
- `_completer`
 - BriarCLI, [54](#)
- `_enroll_frames_iter`
 - BriarClient, [61](#)
- `_enroll_frames_iter2`
 - BriarClient, [62](#)
- `_get_completions`
 - BriarCLICompleter, [56](#)
- `_media_list`
 - MediaStream, [104](#)
- `_prompt_hist_path`
 - BriarCLI, [54](#)
- `_suggest_cmds`
 - BriarCLICompleter, [57](#)
- `_suggest_kwargs`
 - BriarCLICompleter, [57](#)
- `addConnectionOptions`
 - briar.cli.connection, [19](#)
- `addDetectorOptions`
 - briar.cli.detect, [24](#)
- `addEnhanceOptions`
 - briar.cli.enhance, [27](#)
- `addEnrollOptions`
 - briar.cli.enroll, [28](#)
- `addExtractOptions`
 - briar.cli.extract, [30](#)
- `addMediaOptions`
 - briar.cli.media, [32](#)
- `addSearchOptions`
 - briar.cli.search, [34](#)
- `addTrackingOptions`
 - briar.cli.detect, [24](#)
- `annotate`
 - briar.cli.annotate, [18](#)
- `ARGS`
 - CLICommands, [88](#)
- `ATTRIB_IGNORE`
 - briar.grpc_json, [43](#)
- `bcl`
 - briar.ibriar, [43](#)
- `briar`, [11](#)
 - `__version__`, [12](#)
 - `DEFAULT_MAX_MESSAGE_SIZE`, [12](#)
 - `DEFAULT_PORT`, [12](#)
 - `DEFAULT_SERVE_PORT`, [12](#)
 - `dyn_import`, [12](#)
 - `serve`, [12](#)
- `briar.__main__`, [13](#)
- `briar.briar_cli`, [13](#)
 - `briar_command_line`, [13](#)
 - `briar_database_command_line`, [14](#)
 - `briar_test_command_line`, [14](#)
 - `COMMANDS`, [15](#)
 - `DATABASE_COMMANDS`, [15](#)
 - `DETECTION_FILE_EXT`, [15](#)

- FACE_COUNT, 15
- incomplete, 14
- MATCHES_FILE_EXT, 15
- TEMPLATE_FILE_EXT, 16
- briar.briar_client, 16
- briar.briar_media, 16
 - briar_media_from_pb2, 17
 - briar_media_to_pb2, 17
 - load_media_from_folder, 17
 - load_media_from_image, 17
 - load_media_from_numpy, 17
- briar.cli, 18
- briar.cli.annotate, 18
 - annotate, 18
- briar.cli.connection, 19
 - addConnectionOptions, 19
 - DEFAULT_MAX_ASYNC, 19
 - DEFAULT_MAX_MESSAGE_SIZE, 19
- briar.cli.database, 20
 - database_delete, 20
 - database_info, 20
 - database_list, 20
 - database_list_entries, 21
 - database_merge, 21
 - database_rename, 21
 - database_retrieve, 21
 - db_no_exist, 21
 - parseDatabaseDeleteOptions, 22
 - parseDatabaseInfoOptions, 22
 - parseDatabaseListEntriesOptions, 22
 - parseDatabaseListOptions, 22
 - parseDatabaseMergeOptions, 23
 - parseDatabaseRenameOptions, 23
 - parseDatabaseRetrieveOptions, 23
- briar.cli.detect, 24
 - addDetectorOptions, 24
 - addTrackingOptions, 24
 - detect, 25
 - detect_file_iter, 25
 - detect_options2proto, 25
 - DETECTION_FILE_EXT, 26
 - detectParseOptions, 25
 - save_detections, 26
 - tracking_options2proto, 26
- briar.cli.enhance, 26
 - addEnhanceOptions, 27
 - enhance, 27
 - enhance_file_iter, 27
 - enhance_options2proto, 28
 - enhanceParseOptions, 28
 - save_Enhancement, 28
- briar.cli.enroll, 28
 - addEnrollOptions, 28
 - enroll, 29
 - enroll_options2proto, 29
 - enrollParseOptions, 29
- briar.cli.extract, 30
 - addExtractOptions, 30
 - extract, 30
 - extract_options2proto, 30
 - extractParseOptions, 31
 - save_extractions, 31
 - TEMPLATE_FILE_EXT, 31
- briar.cli.finalize, 31
 - database_finalize, 32
 - finalizeParseOptions, 32
- briar.cli.media, 32
 - addMediaOptions, 32
 - collect_files, 33
 - DEFAULT_MAX_SIZE, 33
 - hasExtension, 33
- briar.cli.search, 34
 - addSearchOptions, 34
 - MATCHES_FILE_EXT, 35
 - search, 34
 - search_options2proto, 34
 - searchParseOptions, 35
- briar.cli.sigset, 35
 - parseSigsetEnrollOptions, 35
 - parseSigsetStatsOptions, 36
 - sigset_enroll, 36
 - sigset_stats, 36
- briar.cli.status, 36
 - status, 36
 - statusParseOptions, 37
- briar.cli.test, 37
 - detection_output_tests, 37
 - extraction_output_tests, 38
- briar.cli.track, 38
 - save_tracklets, 38
 - track, 38
 - TRACKLET_FILE_EXT, 39
- briar.cli.viz, 39
 - viz, 39
 - vizParseOptions, 39
- briar.functions, 40
 - new_uuid, 40
- briar.grpc_json, 40
 - ATTRIB_IGNORE, 43
 - dict_to_proto_obj, 41
 - load, 41
 - proto_obj_to_dict, 42
 - save, 42
- briar.ibriar, 43
 - bcl, 43
 - DEFAULT_ARGS, 43
- briar.media, 44
 - decodeMedia, 44
 - ImageGenerator, 44
- briar.media.visualize, 45
 - fdir, 45
 - files, 45
 - visualize_detection, 45
 - visualize_matches, 45
 - visualize_track, 45
- briar.media_converters, 46

- image_cv2proto, 46
- image_np2proto, 47
- image_proto2cv, 47
- image_proto2np, 47
- matrix_np2proto, 48
- matrix_proto2np, 48
- modality_proto2string, 48
- modality_string2proto, 48
- modalityDict, 49
- reverseModalityDict, 50
- vector_np2proto, 49
- vector_proto2np, 49
- briar.sigset, 50
- briar.sigset.parse, 50
 - expandTree, 50
 - parseBriarSigset, 50
- briar.timing, 51
 - DURATION_FILE_EXT, 52
 - end_duration, 51
 - generate_progress, 51
 - print_duration, 51
 - print_durations, 51
 - save_durations, 52
 - start_duration, 52
 - timeElapsed, 52
 - timestamp, 52
- briar_cli.py, 113
- briar_client.py, 113
- briar_command_line
 - briar.briar_cli, 13
- briar_database_command_line
 - briar.briar_cli, 14
- briar_media.py, 114
- briar_media_from_pb2
 - briar.briar_media, 17
- briar_media_to_pb2
 - briar.briar_media, 17
- briar_test_command_line
 - briar.briar_cli, 14
- BriarCLI, 53
 - _PROMPT_TXT, 54
 - __init__, 53
 - _base_prompt_text, 54
 - _cmd_line, 54
 - _completer, 54
 - _prompt_hist_path, 54
- BriarCLICompleter, 55
 - __init__, 56
 - _complete_base_cmd, 56
 - _complete_cmds, 56
 - _complete_from_schema, 56
 - _complete_kwargs, 56
 - _get_completions, 56
 - _suggest_cmds, 57
 - _suggest_kwargs, 57
 - get_completions, 57
- BriarClient, 58
 - __init__, 61
 - _enroll_frames_iter, 61
 - _enroll_frames_iter2, 62
 - channel, 80
 - database_create, 62
 - database_insert, 62
 - database_list_templates, 63
 - database_remove_templates, 63
 - database_retrieve, 64
 - DEFAULT_PORT, 80
 - detect, 64
 - detect_files, 65
 - detect_files_iter, 65
 - detect_frames, 65
 - detect_frames_iter, 66
 - enhance, 66
 - enroll, 66
 - enroll_file_iter, 67
 - enroll_files, 67
 - enroll_frames, 68
 - enroll_frames2, 68
 - enroll_frames_iter, 70
 - extract, 71
 - extract_file_iter, 71
 - extract_files, 72
 - extract_frames, 72
 - extract_frames_iter, 73
 - finalize, 73
 - get_database_names, 73
 - get_status, 73
 - load_database, 74
 - options, 80
 - print_verbose, 74
 - retrieve_req_iter, 75
 - search, 75
 - search_file_iter, 76
 - search_files, 77
 - stub, 80
 - track, 78
 - track_file_iter, 78
 - track_files, 78
 - verify, 79
- BriarMedia, 80
 - __init__, 81
 - channels, 81
 - DATA_TYPES, 81
 - datetime, 81
 - description, 81
 - fps, 82
 - height, 82
 - IMAGE_FORMATS, 82
 - len, 82
 - metadata, 82
 - source, 82
 - VIDEO_FORMATS, 82
 - width, 83
- BriarProgress, 83
 - __init__, 83
 - desc, 84

- enabled, 84
- leave, 84
- name, 84
- pbar, 84
- position, 84
- prevstep, 85
- tqdm, 85
- update, 84
- BriarTest, 85
 - __init__, 86
 - description, 86
 - run, 86
 - test, 86
 - testdata_folder, 86
- BriarTestResult, 87
 - __init__, 87
 - level, 87
 - name, 87
 - passed, 87
 - reason, 87
- cap
 - Videolocator, 108
- channel
 - BriarClient, 80
- channels
 - BriarMedia, 81
- cli/__init__.py, 111
- cli/annotate.py, 114
- cli/connection.py, 114
- cli/database.py, 115
- cli/detect.py, 115
- cli/enhance.py, 116
- cli/enroll.py, 116
- cli/extract.py, 116
- cli/finalize.py, 117
- cli/media.py, 117
- cli/search.py, 118
- cli/sigset.py, 118
- cli/status.py, 118
- cli/test.py, 119
- cli/track.py, 119
- cli/viz.py, 119
- CLICommands, 88
 - ARGS, 88
 - CMD_SCHEMA, 88
 - CMD_SYNTAX, 88
 - COMMAND, 88
 - DETECT, 89
 - ENROLL, 89
 - EXTRACT, 89
 - MODE, 89
- CMD_SCHEMA
 - CLICommands, 88
- CMD_SYNTAX
 - CLICommands, 88
- collect_files
 - briar.cli.media, 33
- COMMAND
 - CLICommands, 88
- COMMANDS
 - briar.briar_cli, 15
- DATA_TYPES
 - BriarMedia, 81
- DATABASE_COMMANDS
 - briar.briar_cli, 15
- database_create
 - BriarClient, 62
- database_delete
 - briar.cli.database, 20
- database_finalize
 - briar.cli.finalize, 32
- database_info
 - briar.cli.database, 20
- database_insert
 - BriarClient, 62
- database_list
 - briar.cli.database, 20
- database_list_entries
 - briar.cli.database, 21
- database_list_templates
 - BriarClient, 63
- database_merge
 - briar.cli.database, 21
- database_remove_templates
 - BriarClient, 63
- database_rename
 - briar.cli.database, 21
- database_retrieve
 - briar.cli.database, 21
 - BriarClient, 64
- DatabaseTest, 90
 - test, 90
- datetime
 - BriarMedia, 81
- db_no_exist
 - briar.cli.database, 21
- decodeMedia
 - briar.media, 44
- default
 - GrpcDecoder, 97
 - GrpcEncoder, 99
- DEFAULT_ARGS
 - briar.ibriar, 43
- DEFAULT_MAX_ASYNC
 - briar.cli.connection, 19
- DEFAULT_MAX_MESSAGE_SIZE
 - briar, 12
 - briar.cli.connection, 19
- DEFAULT_MAX_SIZE
 - briar.cli.media, 33
- DEFAULT_PORT
 - briar, 12
 - BriarClient, 80
- DEFAULT_SERVE_PORT
 - briar, 12
- desc

- BriarProgress, [84](#)
- description
 - BriarMedia, [81](#)
 - BriarTest, [86](#)
 - DetectTest, [92](#)
 - ExtractTest, [95](#)
- DETECT
 - CLICommands, [89](#)
- detect
 - briar.cli.detect, [25](#)
 - BriarClient, [64](#)
- detect_file_iter
 - briar.cli.detect, [25](#)
- detect_files
 - BriarClient, [65](#)
- detect_files_iter
 - BriarClient, [65](#)
- detect_frames
 - BriarClient, [65](#)
- detect_frames_iter
 - BriarClient, [66](#)
- detect_options2proto
 - briar.cli.detect, [25](#)
- DETECTION_FILE_EXT
 - briar.briar_cli, [15](#)
 - briar.cli.detect, [26](#)
- detection_file_path
 - DetectTest, [92](#)
 - ExtractTest, [96](#)
- detection_output_tests
 - briar.cli.test, [37](#)
- detectParseOptions
 - briar.cli.detect, [25](#)
- DetectTest, [91](#)
 - description, [92](#)
 - detection_file_path, [92](#)
 - output_path, [93](#)
 - test_1_detection_image, [92](#)
 - test_2_detection_image_output, [92](#)
 - test_3_detection_image_withreturn, [92](#)
 - test_4_detection_image_output_withreturn, [92](#)
 - testim_path, [93](#)
- dict_to_proto_obj
 - briar.grpc_json, [41](#)
- DURATION_FILE_EXT
 - briar.timing, [52](#)
- dyn_import
 - briar, [12](#)
- enabled
 - BriarProgress, [84](#)
- end_duration
 - briar.timing, [51](#)
- enhance
 - briar.cli.enhance, [27](#)
 - BriarClient, [66](#)
- enhance_file_iter
 - briar.cli.enhance, [27](#)
- enhance_options2proto
 - briar.cli.enhance, [28](#)
- enhanceParseOptions
 - briar.cli.enhance, [28](#)
- ENROLL
 - CLICommands, [89](#)
- enroll
 - briar.cli.enroll, [29](#)
 - BriarClient, [66](#)
- enroll_file_iter
 - BriarClient, [67](#)
- enroll_files
 - BriarClient, [67](#)
- enroll_frames
 - BriarClient, [68](#)
- enroll_frames2
 - BriarClient, [68](#)
- enroll_frames_iter
 - BriarClient, [70](#)
- enroll_options2proto
 - briar.cli.enroll, [29](#)
- enrollParseOptions
 - briar.cli.enroll, [29](#)
- EnrollTest, [93](#)
 - test, [94](#)
- expandTree
 - briar.sigset.parse, [50](#)
- EXTRACT
 - CLICommands, [89](#)
- extract
 - briar.cli.extract, [30](#)
 - BriarClient, [71](#)
- extract_file_iter
 - BriarClient, [71](#)
- extract_files
 - BriarClient, [72](#)
- extract_frames
 - BriarClient, [72](#)
- extract_frames_iter
 - BriarClient, [73](#)
- extract_options2proto
 - briar.cli.extract, [30](#)
- extraction_output_tests
 - briar.cli.test, [38](#)
- extractParseOptions
 - briar.cli.extract, [31](#)
- ExtractTest, [94](#)
 - description, [95](#)
 - detection_file_path, [96](#)
 - output_path, [96](#)
 - template_file_path, [96](#)
 - test_1_extraction_image, [95](#)
 - test_2_extraction_image_output, [95](#)
 - testim_path, [96](#)
- FACE_COUNT
 - briar.briar_cli, [15](#)
- fdir
 - briar.media.visualize, [45](#)
- filepath

- Imageliterator, 102
 - Videoliterator, 108
- files
 - briar.media.visualize, 45
- finalize
 - BriarClient, 73
- finalizeParseOptions
 - briar.cli.finalize, 32
- fps
 - BriarMedia, 82
 - Imageliterator, 102
 - Videoliterator, 108
- frame
 - Imageliterator, 102
- frame_count
 - Imageliterator, 102
 - Videoliterator, 108
- frame_height
 - Imageliterator, 102
 - Videoliterator, 109
- frame_width
 - Imageliterator, 102
 - Videoliterator, 109
- functions.py, 120
- generate_progress
 - briar.timing, 51
- get_completions
 - BriarCLICompleter, 57
- get_database_names
 - BriarClient, 73
- get_status
 - BriarClient, 73
- grpc_json.py, 120
- GrpcDecoder, 96
 - __init__, 97
 - default, 97
 - options, 98
- GrpcEncoder, 98
 - __init__, 99
 - default, 99
 - options, 100
- hasExtension
 - briar.cli.media, 33
- height
 - BriarMedia, 82
 - Rect, 105
- i
 - Imageliterator, 103
 - Videoliterator, 109
- ibriar.py, 121
- image_cv2proto
 - briar.media_converters, 46
- IMAGE_FORMATS
 - BriarMedia, 82
- image_np2proto
 - briar.media_converters, 47
- image_proto2cv
 - briar.media_converters, 47
- image_proto2np
 - briar.media_converters, 47
- ImageGenerator
 - briar.media, 44
- Imageliterator, 100
 - __init__, 101
 - __iter__, 101
 - __len__, 101
 - __next__, 102
 - filepath, 102
 - fps, 102
 - frame, 102
 - frame_count, 102
 - frame_height, 102
 - frame_width, 102
 - i, 103
 - isOpened, 103
 - length, 103
 - msec, 103
 - pos, 103
 - processed, 103
 - start_frame, 103
 - stop_frame, 103
- incomplete
 - briar.briar_cli, 14
- isOpened
 - Imageliterator, 103
 - Videoliterator, 109
- leave
 - BriarProgress, 84
- len
 - BriarMedia, 82
- length
 - Imageliterator, 103
 - Videoliterator, 109
- level
 - BriarTestResult, 87
- load
 - briar.grpc_json, 41
- load_database
 - BriarClient, 74
- load_media_from_folder
 - briar.briar_media, 17
- load_media_from_image
 - briar.briar_media, 17
- load_media_from_numpy
 - briar.briar_media, 17
- MATCHES_FILE_EXT
 - briar.briar_cli, 15
 - briar.cli.search, 35
- matrix_np2proto
 - briar.media_converters, 48
- matrix_proto2np
 - briar.media_converters, 48
- media/ __init__.py, 112

- media/visualize.py, 121
- media_converters.py, 121
- MediaStream, 104
 - __init__, 104
 - __iter__, 104
 - _media_list, 104
- metadata
 - BriarMedia, 82
- modality_proto2string
 - briar.media_converters, 48
- modality_string2proto
 - briar.media_converters, 48
- modalityDict
 - briar.media_converters, 49
- MODE
 - CLICommands, 89
- msec
 - Imageliterator, 103
 - Videoliterator, 109
- name
 - BriarProgress, 84
 - BriarTestResult, 87
- new_uuid
 - briar.functions, 40
- options
 - BriarClient, 80
 - GrpcDecoder, 98
 - GrpcEncoder, 100
- output_path
 - DetectTest, 93
 - ExtractTest, 96
- parseBriarSigset
 - briar.sigset.parse, 50
- parseDatabaseDeleteOptions
 - briar.cli.database, 22
- parseDatabaseInfoOptions
 - briar.cli.database, 22
- parseDatabaseListEntriesOptions
 - briar.cli.database, 22
- parseDatabaseListOptions
 - briar.cli.database, 22
- parseDatabaseMergeOptions
 - briar.cli.database, 23
- parseDatabaseRenameOptions
 - briar.cli.database, 23
- parseDatabaseRetrieveOptions
 - briar.cli.database, 23
- parseSigsetEnrollOptions
 - briar.cli.sigset, 35
- parseSigsetStatsOptions
 - briar.cli.sigset, 36
- passed
 - BriarTestResult, 87
- pbar
 - BriarProgress, 84
- pos
 - Imageliterator, 103
 - Videoliterator, 109
- position
 - BriarProgress, 84
- prevstep
 - BriarProgress, 85
- print_duration
 - briar.timing, 51
- print_durations
 - briar.timing, 51
- print_verbose
 - BriarClient, 74
- processed
 - Imageliterator, 103
 - Videoliterator, 109
- proto_obj_to_dict
 - briar.grpc_json, 42
- readme-cli.md, 122
- reason
 - BriarTestResult, 87
- Rect, 105
 - __init__, 105
 - height, 105
 - width, 106
 - x, 106
 - y, 106
- retrieve_req_iter
 - BriarClient, 75
- reverseModalityDict
 - briar.media_converters, 50
- run
 - BriarTest, 86
- save
 - briar.grpc_json, 42
- save_detections
 - briar.cli.detect, 26
- save_durations
 - briar.timing, 52
- save_Enhancement
 - briar.cli.enhance, 28
- save_extractions
 - briar.cli.extract, 31
- save_tracklets
 - briar.cli.track, 38
- search
 - briar.cli.search, 34
 - BriarClient, 75
- search_file_iter
 - BriarClient, 76
- search_files
 - BriarClient, 77
- search_options2proto
 - briar.cli.search, 34
- searchParseOptions
 - briar.cli.search, 35
- serve
 - briar, 12

- sigset/___init___py, 112
- sigset/parse.py, 122
- sigset_enroll
 - briar.cli.sigset, 36
- sigset_stats
 - briar.cli.sigset, 36
- source
 - BriarMedia, 82
- start_duration
 - briar.timing, 52
- start_frame
 - ImageIterator, 103
 - VideoIterator, 110
- status
 - briar.cli.status, 36
- statusParseOptions
 - briar.cli.status, 37
- stop_frame
 - ImageIterator, 103
 - VideoIterator, 110
- stub
 - BriarClient, 80
- TEMPLATE_FILE_EXT
 - briar.briar_cli, 16
 - briar.cli.extract, 31
- template_file_path
 - ExtractTest, 96
- test
 - BriarTest, 86
 - DatabaseTest, 90
 - EnrollTest, 94
- test_1_detection_image
 - DetectTest, 92
- test_1_extraction_image
 - ExtractTest, 95
- test_2_detection_image_output
 - DetectTest, 92
- test_2_extraction_image_output
 - ExtractTest, 95
- test_3_detection_image_withreturn
 - DetectTest, 92
- test_4_detection_image_output_withreturn
 - DetectTest, 92
- testdata_folder
 - BriarTest, 86
- testim_path
 - DetectTest, 93
 - ExtractTest, 96
- timeElapsed
 - briar.timing, 52
- timestamp
 - briar.timing, 52
- timing/___init___py, 112
- tqdm
 - BriarProgress, 85
- track
 - briar.cli.track, 38
 - BriarClient, 78
- track_file_iter
 - BriarClient, 78
- track_files
 - BriarClient, 78
- tracking_options2proto
 - briar.cli.detect, 26
- TRACKLET_FILE_EXT
 - briar.cli.track, 39
- update
 - BriarProgress, 84
- vector_np2proto
 - briar.media_converters, 49
- vector_proto2np
 - briar.media_converters, 49
- verify
 - BriarClient, 79
- VIDEO_FORMATS
 - BriarMedia, 82
- VideoIterator, 106
 - ___init___, 107
 - ___iter___, 108
 - ___len___, 108
 - ___next___, 108
 - cap, 108
 - filepath, 108
 - fps, 108
 - frame_count, 108
 - frame_height, 109
 - frame_width, 109
 - i, 109
 - isOpened, 109
 - length, 109
 - msec, 109
 - pos, 109
 - processed, 109
 - start_frame, 110
 - stop_frame, 110
- visualize_detection
 - briar.media.visualize, 45
- visualize_matches
 - briar.media.visualize, 45
- visualize_track
 - briar.media.visualize, 45
- viz
 - briar.cli.viz, 39
- vizParseOptions
 - briar.cli.viz, 39
- width
 - BriarMedia, 83
 - Rect, 106
- x
 - Rect, 106
- y
 - Rect, 106