IMIE

Generated by Doxygen 1.8.13

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

Class Documentation

2.1 imie::mgmt::management_client Class Reference

Public Member Functions

- bool connect (std::string &host_or_ip_str, std::string &port_str)

  Connect to the imie management server via grpc.
- void disconnect ()

  Disconnect from imie management server.
- bool ping ()

  Send a ping to the server to verify connection is healthy.
- bool send_reset (std::string module_name, int64_t wgid)

  Request a module to be reset.
- bool set_module_state (std::string module_name, bool enabled, int64_t wgid, std::string sub_module=std↩ ::string())

  Enable or disable a module.
- void get_module_list (messages::mgmt_ext::AllModulesStatus ∗all_modules_status)

  Get a list of all modules, and for each module if it's been registered with the management, and if it has been enabled.
- void list (std::string item_name, std::string module, imie::messages::mgmt_ext::ListResponse ∗list_response, int64_t wgid)

  List all objects of type 'item_name' on 'module' in the workgroup specified by 'wgid'.
- void list (std::string item_name, std::string module, uint32_t id, imie::messages::mgmt_ext::ListResponse ∗list_response, int64_t wgid)

  list the object with id 'id' of type 'item_name' on 'module' in the workgroup specified by 'wgid'.
- void register_listener_callback (CallbackFunc callback)

  Register a callback function to be called for each message we're subscribed to.
- bool subscribe (std::string topic, bool subscribe, int64_t wgid)

  Subscribe to receive status messages. Each message will call the callback registered by 'register_listener_callback()'.
- int64_t add_workgroup (std::string &host_or_ip_str, uint32_t port, std::string wgname)

  Adds a workgroup.
- bool add_source (const imie::messages::types::Source &source, int64_t wgid)

  Add a source.
- bool remove_source (uint32_t source_id, int64_t wgid)
- bool add_flow (imie::messages::types::Flow &flow, int64_t wgid)

  Add a flow.
- bool remove_flow (uint32_t flow_id, int64_t wgid)

Remove a flow.
- bool add_config (imie::messages::mgmt::AddConfig &add_config, int64_t wgid)

  Add a config.
- bool remove_config (uint32_t cfg_id, std::string module_name, int64_t wgid)

  Remove a config.
- bool command (const messages::types::Command &command, int64_t wgid)

  Send command to an imie module.
- bool set_log_level (std::string module_name, imie::common::eLogLevel log_lvl, bool new_state)

  Set the log level of a module.
- bool start_source (uint32_t source_id, int64_t wgid)

  Start a source.
- bool load_yaml (std::string filename, int64_t wgid)

  load a .yaml file, adding all the sources, flows and configs in it.
- int64_t get_workgroup_id (std::string name)

  Get the workgroup id.
- bool set_workgroup_name (int64_t wgid, std::string name)

  Give a workgroup an alias.
- bool push (const std::string &filename, std::string module_name, int64_t wgid)

  copy files from imie mgmt server to module.
- bool pull (std::string module_name, int64_t wgid)

  copy files from imie mgmt server to module.

2.1.1 Member Function Documentation

2.1.1.1 add_config()

```
bool management_client::add_config (
            imie::messages::mgmt::AddConfig & add_config,
            int64_t wgid )
```

Add a config.

Parameters

add_config	The config to add
wgid	the workgroup to add the config to

Returns

true on success
false on failure

2.1.1.2 add_flow()

```
bool management_client::add_flow (
             imie::messages::types::Flow & flow,
             int64_t wgid )
```

Add a flow.

Parameters

flow	The flow to add
wgid	The workgroup to add the flow to

Returns

true on success
false on failure

2.1.1.3 add_source()

```
bool management_client::add_source (
             const imie::messages::types::Source & source,
             int64_t wgid )
```

Add a source.

Parameters

source	The source to add
wgid	The workgroup to add the source to

Returns

true on success
false on failure

2.1.1.4 add_workgroup()

```
int64_t management_client::add_workgroup (
             std::string & host_or_ip_str,
             uint32_t port,
             std::string wgname )
```

Adds a workgroup.

Parameters

host_or_ip_str	The hostname or IP adress of the workgroup
port	The port used by the workgroup
wgname	An alias that can be used to refer to the workgroup in the future

Returns

int64_t An ID identifying the workgroup for future reference. Can be used interchangibly to the wgname.

2.1.1.5 command()

```
bool management_client::command (
            const messages::types::Command & command,
            int64_t wgid )
```

Send command to an imie module.

Parameters

command	Message containing the target module id, and the command. Possible commands: pnpt::eCommands::START, pnpt::eCommands::STOP
wgid	The workgroup to execute the command on

Returns

true on success
false on failure

2.1.1.6 connect()

```
bool management_client::connect (
            std::string & host_or_ip_str,
            std::string & port_str )
```

Connect to the imie management server via grpc.

Parameters

host_or_ip_str	The hostname or the ip address of the server
port_str	The port the server is listening on

Returns

true connection successful
false connection failed

2.1.1.7 get_module_list()

```
void management_client::get_module_list (
            messages::mgmt_ext::AllModulesStatus * all_modules_status )
```

Get a list of all modules, and for each module if it's been registered with the management, and if it has been enabled.

Parameters

all_modules_status	Output parameter - will contain the list after execution.

2.1.1.8 get_workgroup_id()

```
int64_t management_client::get_workgroup_id (
            std::string name )
```

Get the workgroup id.

Parameters

name	The name of the workgroup.

Returns

int64_t The workgroup ID.

2.1.1.9 list() [1/2]

```
void management_client::list (
            std::string item_name,
            std::string module,
            imie::messages::mgmt_ext::ListResponse * list_response,
            int64_t wgid )
```

List all objects of type 'item_name' on 'module' in the workgroup specified by 'wgid'.

Parameters

item_name	A string specifying what item to request. Can be: config, source, flow or workgroup.
module	The module to be probed. Use the string "all" to probe all modules.
list_response	Output parameter - will contain the list of requested items.
wgid	The workgroup to be probed. Use -1 to probe all workgroups.

2.1.1.10 list() [2/2]

```
void management_client::list (
            std::string item_name,
            std::string module,
            uint32_t id,
            imie::messages::mgmt_ext::ListResponse * list_response,
            int64_t wgid )
```

list the object with id 'id' of type 'item_name' on 'module' in the workgroup specified by 'wgid'.

Parameters

item_name	A string specifying what item to request. Can be: config, source, flow or workgroup.
module	The module to be probed. Use the string "all" to probe all modules.
id	The id of the object to show
list_response	Output parameter - will contain the requested item, if it exists
wgid	The workgroup to be probed. Use -1 to probe all workgroups.

2.1.1.11 load_yaml()

```
bool management_client::load_yaml (
            std::string filename,
            int64_t wgid )
```

load a .yaml file, adding all the sources, flows and configs in it.

Parameters

filename	The path to the yaml file
wgid	The workgroup to load the yaml file on

Returns

true on success
false on failure

2.1.1.12 ping()

```
bool management_client::ping ( )
```

Send a ping to the server to verify connection is healthy.

Returns

true Received response to the ping
false Sending of ping failed or received improper response.

2.1.1.13 pull()

```
bool management_client::pull (
             std::string module_name,
             int64_t wgid )
```

copy files from imie mgmt server to module.

Parameters

filename	the file to copy
module_name	the name of the module
wgid	The workgroup the module is on

Returns

true on success
false on failure

2.1.1.14 push()

```
bool management_client::push (
             const std::string & filename,
             std::string module_name,
             int64_t wgid )
```

copy files from imie mgmt server to module.

Parameters

filename	the file to copy
module_name	the name of the module
wgid	The workgroup the module is on

Returns

true on success
false on failure

2.1.1.15 register_listener_callback()

```
void management_client::register_listener_callback (
            CallbackFunc callback )
```

Register a callback function to be called for each message we're subscribed to.

Parameters

callback	the callback function.

2.1.1.16 remove_config()

```
bool management_client::remove_config (
            uint32_t cfg_id,
            std::string module_name,
            int64_t wgid )
```

Remove a config.

Parameters

cfg_id	The ID of the config to remove
module_name	The module on which the config is to be removed
wgid	The workgroup on which the config will be removed

Returns

true on success
false on failure

2.1.1.17 remove_flow()

```
bool management_client::remove_flow (
            uint32_t flow_id,
            int64_t wgid )
```

Remove a flow.

Parameters

flow←↩ _id	The ID of the flow to remove
wgid	The workgroup to remove the flow from

Returns

true on success
false on failure

2.1.1.18 remove_source()

```
bool management_client::remove_source (
            uint32_t source_id,
            int64_t wgid )
```

Parameters

source←↩ _id	The ID of the source to remove
wgid	The workgroup to add the source to

Returns

true on success
false on failure

2.1.1.19 send_reset()

```
bool management_client::send_reset (
            std::string module_name,
            int64_t wgid )
```

Request a module to be reset.

Parameters

module_name	The module to be reset. Send the string "all" to reset all modules.
wgid	The ID of the workgroup that on which the module will be reset. Send -1 as wgid to reset the module on all workgroups

Returns

true on success
false on failure

2.1.1.20 set_log_level()

```
bool management_client::set_log_level (
            std::string module_name,
            imie::common::eLogLevel log_lvl,
            bool new_state )
```

Set the log level of a module.

Parameters

module_name	The name of the module to set the loglevel on. Use the string "all" for all modules
log_lvl	an enum specifying what log levels should be written to logs
new_state	a boolean specifying if the messages on this log level should be written or not

Returns

true on success
false on failure

2.1.1.21 set_module_state()

```
bool management_client::set_module_state (
            std::string module_name,
            bool enabled,
            int64_t wgid,
            std::string sub_module = std::string() )
```

Enable or disable a module.

Parameters

module_name	The name of the module. Send the string "all" to set all modules.
enabled	a boolean stating if the module is to be enabled or disabled
wgid	The ID of the workgroup on which to operate, send -1 to operate on all workgroups
sub_module	The submodule on which to operate. Leave empty if not necessary

Returns

true on success
false on failure

2.1.1.22 set_workgroup_name()

```
bool management_client::set_workgroup_name (
            int64_t wgid,
            std::string name )
```

Give a workgroup an alias.

Parameters

wgid	The workgroup ID
name	A name by which the workgroup will be accessible.

Returns

true on success
false on failure

2.1.1.23 start_source()

```
bool management_client::start_source (
            uint32_t source_id,
            int64_t wgid )
```

Start a source.

Parameters

source←↩ _id	The ID of the source to start
wgid	The workgroup to start the source on

Returns

true on success
false on failure

2.1.1.24 subscribe()

```
bool management_client::subscribe (
            std::string topic,
            bool subscribe,
            int64_t wgid )
```

Subscribe to receive status messages. Each message will call the callback registered by 'register_listener_↩ callback()'.

Parameters

topic	A string specifying what statistics we're subscribing to. Possible topics: mstream, mdecode, inference, tcp_sender
subscribe	A boolean specifying if we want to subscribe or unsubscribe
wgid	The ID of the workgroup we want to subscribe to. Use -1 to subscribe to all workgroups.

Returns

true for success
false for failure

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

- oss/lib/mgmt/include/imie_mgmt_lib.h
- oss/lib/mgmt/src/imie_mgmt_lib.cpp

2.2 imie::mstream::streaming_client Class Reference

Public Member Functions

- bool connect (std::string &host_or_ip_str, std::string port_str="")

  Connect to the msl server.
- void disconnect ()

  Disconnect from the server.
- bool is_connected ()

  Check the connection to the msl server.
- uint32_t get_client_id ()

  Get the client id, alloted on connection.
- bool infer (const uint32_t flow_id, const imie::messages::enums::FrameFormat frame_format, const imie↩ ::messages::types::FramesData &frames_data, imie::messages::msl::InferResponse &response)

  Run the inference for a particular flow.
- bool subscribe (uint32_t flow_id, uint32_t stage_id, bool subscribe)

  Subscribe to inference results of a particular flow. For each result the callback registered by register_listener_↩ callback() will be run.
- void register_listener_callback (CallbackFunc callback)

  Register a callback to be run on every message we're subscribed to.
- bool start_stream_file (const std::string &filename, uint32_t flow_id, float max_mbps=0)

  Start streaming a video file cyclicly.
- bool stop_stream_file (uint32_t flow_id)

  Stop streaming file.

Static Public Attributes

- static const size_t **max_message_size** = 128 $*$ 1024 $*$ 1024
- static const int32_t **default_port** = 50055

2.2.1 Member Function Documentation

2.2.1.1 connect()

```
bool imie::mstream::streaming_client::connect (
            std::string & host_or_ip_str,
            std::string port_str = "" )
```

Connect to the msl server.

Parameters

host_or_ip_str	hostname of ip address of the server
port_str	the port the server is listening on

Returns

true on success
false on failure

2.2.1.2 get_client_id()

```
uint32_t imie::mstream::streaming_client::get_client_id ( )
```

Get the client id, alloted on connection.

Returns

uint32_t a unique ID of this client

2.2.1.3 infer()

```
bool imie::mstream::streaming_client::infer (
            const uint32_t flow_id,
            const imie::messages::enums::FrameFormat frame_format,
            const imie::messages::types::FramesData & frames_data,
            imie::messages::msl::InferResponse & response )
```

Run the inference for a particular flow.

Parameters

flow_id	The flow to run the inference on
frame_format	Enum specifying format of the frames
frames_data	Width/height and payload
response	The response of the inference, reporting if the inference was successful

Returns

true on success
false on failure

2.2.1.4 is_connected()

```
bool imie::mstream::streaming_client::is_connected ( )
```

Check the connection to the msl server.

Returns

true if there's a connection
false there

2.2.1.5 register_listener_callback()

```
void imie::mstream::streaming_client::register_listener_callback (
             CallbackFunc callback )
```

Register a callback to be run on every message we're subscribed to.

Parameters

callback	the callback.

2.2.1.6 start_stream_file()

```
bool imie::mstream::streaming_client::start_stream_file (
             const std::string & filename,
             uint32_t flow_id,
             float max_mbps = 0 )
```

Start streaming a video file cyclicly.

Parameters

filename	The file to stream
flow_id	A unique id
max_rate	Maximum rate for streaming to be in, 0 for unlimited

2.2.1.7 stop_stream_file()

```
bool imie::mstream::streaming_client::stop_stream_file (
             uint32_t flow_id )
```

Stop streaming file.

Parameters

flow←↩ _id	an identifyer of the stream

2.2.1.8 subscribe()

```
bool imie::mstream::streaming_client::subscribe (
             uint32_t flow_id,
             uint32_t stage_id,
             bool subscribe )
```

Subscribe to inference results of a particular flow. For each result the callback registered by register_listener_←↩ callback() will be run.

Parameters

flow_id	The ID of the flow
stage_id	The ID of the stage we want to get results from
subscribe	A boolean specifying if we want to subscribe or unsubscribe

Returns

true on success
false on failure

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

- oss/lib/msl/include/imie_msl_streaming_lib.h
- oss/lib/msl/src/imie_msl_streaming_lib.cpp

Index