MVZMQNetwork

Generated by Doxygen 1.8.16

1 Mantis Vision: MVZMQNetwork 1

2 Protocol Description 3

3 Release Notes 7

4 Data Structure Index 9

4.1 Data Structures . . . 9

5 File Index 11

5.1 File List . . . 11

6 Data Structure Documentation 13

6.1 MVZMQNetwork::Receiver Class Reference . . . 13

6.1.1 Detailed Description . . . 13

6.1.2 Constructor & Destructor Documentation . . . 14

6.1.2.1 Receiver() . . . 14

6.1.2.2 ~Receiver() . . . 14

6.1.3 Member Function Documentation . . . 14

6.1.3.1 GetProfile() . . . 14

6.1.3.2 GetStreamIDs() . . . 14

6.1.3.3 GetStreamInfo() . . . 15

6.1.3.4 ReceiveAtom() . . . 15

6.1.3.5 Running() . . . 15

6.1.3.6 Start() . . . 15

6.2 MVZMQNetwork::Transmitter Class Reference . . . 16

6.2.1 Detailed Description . . . 17

6.2.2 Constructor & Destructor Documentation . . . 17

6.2.2.1 Transmitter() . . . 17

6.2.2.2 ~Transmitter() . . . 17

6.2.3 Member Function Documentation . . . 17

6.2.3.1 GetDroppedAtomsCount() . . . 17

6.2.3.2 Running() . . . 18

6.2.3.3 SendAtom() . . . 18

6.2.3.4 Start() . . . 18

7 File Documentation 21

7.1 public/MVZMQNetwork/MessageType.h File Reference . . . 21

7.1.1 Enumeration Type Documentation . . . 21

7.1.1.1 MessageType . . . 21

7.2 public/MVZMQNetwork/MVZMQNetworkVersion.h File Reference . . . 22

7.3 public/MVZMQNetwork/Protocol.h File Reference . . . 22

Index 23

Chapter 1

Mantis Vision: MVZMQNetwork

An API for establishing network communication in the environment of Mvx2 framework, based on ZeroMQ library

Description

MVZMQNetwork is a library project, which provides basic network-data-receiving and network-data-transmitting classes usable in Mvx2 framework environment. Plugins of Mvx2 may use features of this library in order to avoid implementing complicated networking procedures on their own.

The implementation of the library includes a rather sophisticated handshake protocol, which ensures that communicating peers have enough knowledge about each other. The protocol is being evolved whenever necessary, but a big effort is made every time to secure compatibility with older versions of the protocol as much as possible.

The library is internally based on facilities of ZeroMQ library (http://zeromq.org/), which takes care of lower-level data transmission between peers in a form of messages.

Table of Contents

- Release Notes
- Protocol Description

Chapter 2

Protocol Description

A description of the handshake protocol - its versions, with all the details regarding messages content and information related to ensuring compatibility with older versions.

Version #0

(no longer supported)

Commands sockets

3-message handshake:

- profile
- stream info
- num streams

Data sockets

ZMQ_PUB - ZMQ_SUB model

Handshake protocol

Receiver	Transmitter
1. request profile	
	2. response profile
3. request stream info	
	4. response stream info
5. request num streams	
	6. response num streams

Version #1

Commands sockets

1-message handshake :

- stream description (profile, stream info, num streams)

Data sockets

ZMQ_PUB - ZMQ_SUB model

Handshake protocol

Receiver	Transmitter
1. request stream description [15 B - mvx header] [1 B - message type '1']	
	2. response stream description [15 B - mvx header] [1 B - message type '2'] [1 B - data profiles count N] [N $*$ 3 $*$ 16 B - data profile GUID triplets] [4 + {var} B - serialized stream info] [1 B - num streams]

Version #2

Commands sockets

2-message handshake:

- protocol version
- stream description (profile, stream info, num streams)

Data sockets

ZMQ_PUB - ZMQ_SUB model

Handshake protocol

Receiver	Transmitter
1. request transmitter version, send receiver version [15 B - mvx header] [1 B - message type '1'] [2 B - receiver version]	
	2. [if receiver version == 1] **response stream description** [15 B - mvx header] [1 B - message type '2'] [1 B - data profiles count N] [N $*$ 3 $*$ 16 B - data profile GUID triplets] [4 + {var} B - serialized stream info] [1 B - num streams]
	2. [if receiver version == 2] **response transmitter version** [15 B - mvx header] [1 B - message type '6'] [2 B - transmitter version]
3. [if transmitter version == 2] **request stream description** [15 B - mvx header] [1 B - message type '4'] [2 B - receiver version]	
	4. response stream description [15 B - mvx header] [1 B - message type '5'] [1 B - data profiles count N] [N $*$ 3 $*$ 16 B - data profile GUID triplets] [4 + {var} B - serialized stream info] [1 B - num streams]

Version #3

Commands sockets

2-message handshake:

- protocol version
- stream description (profile, stream info, num streams, stream IDs)

Data sockets

ZMQ_PUB - ZMQ_SUB model

Handshake protocol

Receiver	Transmitter
1. request transmitter version, send receiver version [15 B - mvx header] [1 B - message type '1'] [2 B - receiver version]	
	2. [if receiver version == 1] **response stream description** [15 B - mvx header] [1 B - message type '2'] [1 B - data profiles count N] [N ∗ 3 ∗ 16 B - data profile GUID triplets] [4 + {var} B - serialized stream info] [1 B - num streams]
	2. [if receiver version == {2,3}] **response transmitter version** [15 B - mvx header] [1 B - message type '6'] [2 B - transmitter version]
3. [if transmitter version == {2,3}] **request stream description** [15 B - mvx header] [1 B - message type '4'] [2 B - receiver version]	
	4. [if receiver version == 2] **response stream description** [15 B - mvx header] [1 B - message type '5'] [1 B - data profiles count N] [N ∗ 3 ∗ 16 B - data profile GUID triplets] [4 + {var} B - serialized stream info] [1 B - num streams]
	4. [if receiver version == 3] **response stream description** [15 B - mvx header] [1 B - message type '5'] [1 B - data profiles count N] [N ∗ 3 ∗ 16 B - data profile GUID triplets] [4 + {var} B - serialized stream info] [2 B - num streams M] [M ∗ 2 B - stream IDs]
5. [if transmitter version == 2] **generate stream IDs in range (0, N-1)**	

Chapter 3

Release Notes

1.0.0

Initial version with protocol version 3.

Documentation

- **1.0.0_D1** | added 'release notes' section
- **1.0.0_D2** | added API reference documentation

Build support

- **1.0.0_BS1** | added MVZMQNetwork's own MVZMQNetworkConfig.cmake file for cmake support

2.0.0

Module

- **2.0.0_M1** | updated `Mvx2` 3rdparty dependency to version 3.0.0
- **2.0.0_M2** | removed deprecated StreamZMQNetworkReceiver class
- **2.0.0_M3** | `sendDataBufferCapacity` parameter of MVZMQNetwork::Transmitter::Start function is interpreted as buffer capacity 'per peer and per stream' instead of 'per peer' (i.e. number of streams in frames affects buffer capacity as well)
- **2.0.0_M4** | `receiveDataBufferCapacity` parameter of MVZMQNetwork::Receiver::Start function is interpreted as buffer capacity 'per stream' instead of 'per connection' (i.e. number of streams in frames from transmitter affects buffer capacity)

Build support

- **2.0.0_BS1** | size of Android and LuminOS libraries reduced by ~90%
- **2.0.0_BS2** | android API level raised from 19 to 21
- **2.0.0_BS3** | Linux and MacOS binaries do not consist of a versioned library file and a version-neutral symlink file anymore - the library file itself has version-neutral name

3.0.0

Module

- **3.0.0_M1** | updated `Mvx2` 3rdparty dependency to version 4.0.0

3.1.0

Module

- **3.1.0_M1** | added an option to enable experimental IPv6 support in transmitters and receivers
  - the feature may not work on all platforms as expected and may even prevent correct functioning of IPv4 communication
  - introduced `enableIPv6` parameter to MVZMQNetwork::Transmitter::Transmitter constructor with `false` as default value
  - introduced `enableIPv6` parameter to MVZMQNetwork::Receiver::Receiver constructor with `false` as default value

4.0.0

Module

- **4.0.0_M1** | updated `MVCommon` 3rdparty dependency to version 3.0.0
- **4.0.0_M2** | updated `Mvx2` 3rdparty dependency to version 5.0.0
- **4.0.0_M3** | logs generated by the module use custom `MVZMQNetwork` tag instead of the generic (`MVX2`) one

Build support

- **4.0.0_BS1** | CMake minimal required version increased from 3.9 to 3.14
  - updated `MVZMQNetworkConfig.cmake` script and its dependencies

5.0.0

Module

- **5.0.0_M1** | updated `MVCommon` 3rdparty dependency to version 4.0.0
- **5.0.0_M2** | updated `Mvx2` 3rdparty dependency to version 6.0.0

Build support

- **5.0.0_BS1** | from now on the windows libraries are compiled using msvc compiler version 142 (VS 2019)
- **5.0.0_BS2** | upgraded `cmake/toolchains/ios.cmake` toolchain file used for building for iOS platform

Documentation

- **5.0.0_D1** | introduced PDF documentation as an alternative to the HTML one:
  - `doc/MVZMQNetwork.pdf`

5.0.1

Package

- **5.0.1_PA1** | added missing library dependencies to Windows and Linux packages (libzmq.dll and libzmq.so respectively)

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

MVZMQNetwork::Receiver
A data receiver 13
MVZMQNetwork::Transmitter
A data transmitter 16

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

public/MVZMQNetwork/MessageType.h 21
public/MVZMQNetwork/**MVZMQNetworkAPI.h** . . . . . **??**
public/MVZMQNetwork/MVZMQNetworkVersion.h 22
public/MVZMQNetwork/Protocol.h 22
public/MVZMQNetwork/**Receiver.h** . . . . . **??**
public/MVZMQNetwork/**Transmitter.h** . . . . . **??**

Chapter 6

Data Structure Documentation

6.1 MVZMQNetwork::Receiver Class Reference

A data receiver.

`#include <Receiver.h>`

Public Member Functions

- Receiver (bool enableIPv6=false)

  A constructor.
- ~Receiver ()

  A destructor.
- bool Start (std::set< ProtocolVersion > const &unsupportedTransmitterProtocolVersions, MVCommon::↩ String const &commandsAddress, MVCommon::String const &dataAddress, uint32_t receiveDataBuffer↩ Capacity, uint64_t responseReceiveTimeout=3000)

  Starts the receiver's procedure.
- void Stop ()

  Stops the receiver's procedure.
- bool Running () const

  Determines whether the receiver's procedure is running.
- std::shared_ptr< MVX::Profile const > GetProfile () const

  Provides a transmitter's declared profile of a transmitted stream of data.
- std::shared_ptr< MVX::DataTypeStreamInfo const > GetStreamInfo () const

  Provides transmitter's declared properties of a transmitted stream of data.
- const std::vector< MVX::StreamId > & GetStreamIDs () const

  Provides a transmitter's declared collection of identifiers of transmitted MVX streams/atoms.
- std::shared_ptr< MVX::Atom > ReceiveAtom ()

  Receives an atom from transmitter.

6.1.1 Detailed Description

A data receiver.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 Receiver()

```
MVZMQNetwork::Receiver::Receiver (
            bool enableIPv6 = false )
```

A constructor.

Parameters

enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

6.1.2.2 ~Receiver()

```
MVZMQNetwork::Receiver::~Receiver ( )
```

A destructor.

Closes a connection.

6.1.3 Member Function Documentation

6.1.3.1 GetProfile()

```
std::shared_ptr<MVX::Profile const> MVZMQNetwork::Receiver::GetProfile ( ) const
```

Provides a transmitter's declared profile of a transmitted stream of data.

Returns

a declared stream profile or nullptr if connection is not established

6.1.3.2 GetStreamIDs()

```
const std::vector<MVX::StreamId>& MVZMQNetwork::Receiver::GetStreamIDs ( ) const
```

Provides a transmitter's declared collection of identifiers of transmitted MVX streams/atoms.

Returns

declared MVX stream identifiers or an empty collection if connection is not established

6.1.3.3 GetStreamInfo()

```
std::shared_ptr<MVX::DataTypeStreamInfo const> MVZMQNetwork::Receiver::GetStreamInfo ( ) const
```

Provides transmitter's declared properties of a transmitted stream of data.

Returns

declared stream properties or nullptr if connection is not established

6.1.3.4 ReceiveAtom()

```
std::shared_ptr<MVX::Atom> MVZMQNetwork::Receiver::ReceiveAtom ( )
```

Receives an atom from transmitter.

Returns

a received atom or nullptr if connection is not established or there is no atom available

6.1.3.5 Running()

```
bool MVZMQNetwork::Receiver::Running ( ) const
```

Determines whether the receiver's procedure is running.

Returns

true in case the receiver's procedure is running

6.1.3.6 Start()

```
bool MVZMQNetwork::Receiver::Start (
            std::set< ProtocolVersion > const & unsupportedTransmitterProtocolVersions,
            MVCommon::String const & commandsAddress,
            MVCommon::String const & dataAddress,
            uint32_t receiveDataBufferCapacity,
            uint64_t responseReceiveTimeout = 3000 )
```

Starts the receiver's procedure.

Parameters

unsupportedTransmitterProtocolVersions	a collection of transmitter versions this receiver shall not communicate with
commandsAddress	an address in zmq-accepted format of commands socket of a transmitter to connect to (e.g. 'tcp://192.168.0.11:5555', 'tcp://localhost:5555')
dataAddress	an address in zmq-accepted format of data socket of a transmitter to connect to (e.g. 'tcp://192.168.0.11:5556', 'tcp://localhost:5556')
receiveDataBufferCapacity	a capacity of receive data buffer (in messages) per stream (check ZMQ_RCVHWM option of zmq_setsockopt()) (e.g. when the transmitter provides frames with 3 streams, total capacity of the data buffer is 3 times greater than the passed value)
responseReceiveTimeout	an interval available for establishing connection with a transmitter (in ms) before the operation is considered unsuccessful

Returns

true in case the receiver managed to connect to a transmitter and its procedure started successfully

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

- public/MVZMQNetwork/Receiver.h

6.2 MVZMQNetwork::Transmitter Class Reference

A data transmitter.

`#include <Transmitter.h>`

Public Member Functions

- Transmitter (bool enableIPv6=false)

  A constructor.
- ~Transmitter ()

  A destructor.
- bool Start (std::set< ProtocolVersion > const &unsupportedReceiverProtocolVersions, MVCommon::String const &commandsAddress, MVCommon::String const &dataAddress, uint32_t sendDataBufferCapacity, std::shared_ptr< MVX::Profile > spProfile, std::shared_ptr< MVX::DataTypeStreamInfo > spStreamInfo, std::vector< MVX::StreamId > const &streamIDs)

  Starts the transmitter's procedure.
- void Stop ()

  Stops the transmitter's procedure.
- bool Running () const

  Determines whether the transmitter's procedure is running.
- bool SendAtom (std::shared_ptr< MVX::Atom > spAtom)

  Sends an atom to all connected receivers.
- void ResetDroppedAtomsCounter ()

  Resets a counter of dropped atoms.
- uint64_t GetDroppedAtomsCount () const

  Returns a count of dropped atoms.

6.2.1 Detailed Description

A data transmitter.

For monitoring purposes keeps a counter of dropped atoms (i.e. atoms whose transmission failed).

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Transmitter()

```
MVZMQNetwork::Transmitter::Transmitter (
            bool enableIPv6 = false )
```

A constructor.

Parameters

enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

6.2.2.2 ~Transmitter()

```
MVZMQNetwork::Transmitter::~Transmitter ( )
```

A destructor.

Closes any open connections.

6.2.3 Member Function Documentation

6.2.3.1 GetDroppedAtomsCount()

```
uint64_t MVZMQNetwork::Transmitter::GetDroppedAtomsCount ( ) const
```

Returns a count of dropped atoms.

Returns

dropped atoms count

6.2.3.2 Running()

```
bool MVZMQNetwork::Transmitter::Running ( ) const
```

Determines whether the transmitter's procedure is running.

Returns

true in case the transmitter's procedure is running

6.2.3.3 SendAtom()

```
bool MVZMQNetwork::Transmitter::SendAtom (
            std::shared_ptr< MVX::Atom > spAtom )
```

Sends an atom to all connected receivers.

Parameters

spAtom	an atom to send

Returns

true in case the transmitter managed to send the atom (which however does not guarantee that all receivers managed to get it)

6.2.3.4 Start()

```
bool MVZMQNetwork::Transmitter::Start (
            std::set< ProtocolVersion > const & unsupportedReceiverProtocolVersions,
            MVCommon::String const & commandsAddress,
            MVCommon::String const & dataAddress,
            uint32_t sendDataBufferCapacity,
            std::shared_ptr< MVX::Profile > spProfile,
            std::shared_ptr< MVX::DataTypeStreamInfo > spStreamInfo,
            std::vector< MVX::StreamId > const & streamIDs )
```

Starts the transmitter's procedure.

Parameters

unsupportedReceiverProtocolVersions	a collection of receiver versions this transmitter shall not respond to
commandsAddress	an address in zmq-accepted format to bind the transmitter's commands socket with (e.g. 'tcp://∗:5555')
dataAddress	an address in zmq-accepted format to bind the transmitter's data socket with (e.g. 'tcp://∗:5556')

Parameters

sendDataBufferCapacity	a capacity of data buffer (in messages) per peer and per stream (check ZMQ_SNDHWM option of zmq_setsockopt()) (e.g. when the number of streams in frames is 3, total capacity of the data buffer per peer is 3 times greater than the passed value)
spProfile	a declared profile of a to-be transmitted stream of data
spStreamInfo	declared properties of a to-be transmitted stream of data
streamIDs	a declared collection of identifiers of a to-be transmitted MVX streams/atoms

Returns

true in case the transmitter's procedure started successfully

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

- public/MVZMQNetwork/Transmitter.h

Chapter 7

File Documentation

7.1 public/MVZMQNetwork/MessageType.h File Reference

Enumerations

- enum MVZMQNetwork::MessageType {
  MVZMQNetwork::MT_RESPONSE_INVALID_REQUEST = 3, MVZMQNetwork::MT_REQUEST_PROTOCOL_VERSION
  = 1, MVZMQNetwork::MT_REQUEST_STREAM_DESCRIPTION = 4, MVZMQNetwork::MT_RESPONSE_STREAM_DESCRIP
  = 2,
  MVZMQNetwork::MT_RESPONSE_STREAM_DESCRIPTION = 5, MVZMQNetwork::MT_RESPONSE_PROTOCOL_VERSIO
  = 6 }

  An enumeration with message types - each having unique integer value.

7.1.1 Enumeration Type Documentation

7.1.1.1 MessageType

`enum MVZMQNetwork::MessageType`

An enumeration with message types - each having unique integer value.

Enumerator

MT_RESPONSE_INVALID_REQUEST	An 'invalid-request response' message type. Value introduced in protocol version 1.
MT_REQUEST_PROTOCOL_VERSION	A 'request protocol version' message type. Value introduced in protocol version 1 as original MT_REQUEST_STREAM_DESCRIPTION. This value must always be the initial message sent by receiver when establishing connection with a transmitter.
MT_REQUEST_STREAM_DESCRIPTION	A 'request stream description' message type. Value introduced in protocol version 2.
MT_RESPONSE_STREAM_DESCRIPTION_V1	A 'response stream description' message type for receiver version 1. Value introduced in protocol version 1 as original MT_RESPONSE_STREAM_DESCRIPTION.
MT_RESPONSE_STREAM_DESCRIPTION	A 'response stream description' message type for receiver version $>$ 1. Value introduced in protocol version 2.
MT_RESPONSE_PROTOCOL_VERSION	A 'response protocol version' message type. Value introduced in protocol version 2.

7.2 public/MVZMQNetwork/MVZMQNetworkVersion.h File Reference

```
#include <MVCommon/utils/VersionInfo.h>
```

Macros

- #define MVZMQNETWORK_VERSION_MAJOR 5

  Current value of the most-significant MVZMQNetwork version component.
- #define MVZMQNETWORK_VERSION_MINOR 0

  Current value of the medium-significant MVZMQNetwork version component.
- #define MVZMQNETWORK_VERSION_PATCH 2

  Current value of the least-significant MVZMQNetwork version component.

Variables

- const MVCommon::VersionInfo MVZMQNetwork::MVZMQNETWORK_VERSION = { 5 , 0 , 2 }

  Current version of MVZMQNetwork.
- const MVCommon::VersionInfo MVZMQNetwork::MVX_VERSION

  A version of Mvx2 framework that MVZMQNetwork was compiled with.

7.3 public/MVZMQNetwork/Protocol.h File Reference

```
#include <cstdint>
```

Macros

- #define PROTOCOL_VERSION_1 1

  A protocol version 1 value.
- #define PROTOCOL_VERSION_2 2

  A protocol version 2 value.
- #define PROTOCOL_VERSION_3 3

  A protocol version 3 value.
- #define CURRENT_PROTOCOL_VERSION PROTOCOL_VERSION_3

  Current protocol version value.

Typedefs

- using MVZMQNetwork::ProtocolVersion = uint16_t

  A type of protocol versions.

Index

~Receiver
  MVZMQNetwork::Receiver, 14
~Transmitter
  MVZMQNetwork::Transmitter, 17

GetDroppedAtomsCount
  MVZMQNetwork::Transmitter, 17
GetProfile
  MVZMQNetwork::Receiver, 14
GetStreamIDs
  MVZMQNetwork::Receiver, 14
GetStreamInfo
  MVZMQNetwork::Receiver, 14

MessageType
  MessageType.h, 21
MessageType.h
  MessageType, 21
  MT_REQUEST_PROTOCOL_VERSION, 21
  MT_REQUEST_STREAM_DESCRIPTION, 21
  MT_RESPONSE_INVALID_REQUEST, 21
  MT_RESPONSE_PROTOCOL_VERSION, 21
  MT_RESPONSE_STREAM_DESCRIPTION, 21
  MT_RESPONSE_STREAM_DESCRIPTION_V1, 21
MT_REQUEST_PROTOCOL_VERSION
  MessageType.h, 21
MT_REQUEST_STREAM_DESCRIPTION
  MessageType.h, 21
MT_RESPONSE_INVALID_REQUEST
  MessageType.h, 21
MT_RESPONSE_PROTOCOL_VERSION
  MessageType.h, 21
MT_RESPONSE_STREAM_DESCRIPTION
  MessageType.h, 21
MT_RESPONSE_STREAM_DESCRIPTION_V1
  MessageType.h, 21
MVZMQNetwork::Receiver, 13
  ~Receiver, 14
  GetProfile, 14
  GetStreamIDs, 14
  GetStreamInfo, 14
  ReceiveAtom, 15
  Receiver, 14
  Running, 15
  Start, 15
MVZMQNetwork::Transmitter, 16
  ~Transmitter, 17
  GetDroppedAtomsCount, 17
  Running, 17
  SendAtom, 18
  Start, 18
  Transmitter, 17

public/MVZMQNetwork/MessageType.h, 21
public/MVZMQNetwork/MVZMQNetworkVersion.h, 22
public/MVZMQNetwork/Protocol.h, 22

ReceiveAtom
  MVZMQNetwork::Receiver, 15
Receiver
  MVZMQNetwork::Receiver, 14
Running
  MVZMQNetwork::Receiver, 15
  MVZMQNetwork::Transmitter, 17

SendAtom
  MVZMQNetwork::Transmitter, 18
Start
  MVZMQNetwork::Receiver, 15
  MVZMQNetwork::Transmitter, 18

Transmitter
  MVZMQNetwork::Transmitter, 17