ConcurrentDataSharer

Generated by Doxygen 1.8.11

Contents

| | REA | DIVIE | | | • |
|---|-------|---------|------------|-------------------------------------|----|
| 2 | Hiera | archica | l Index | | 3 |
| | 2.1 | Class I | Hierarchy | | 3 |
| 3 | Clas | s Index | | | 5 |
| | 3.1 | Class I | List | | 5 |
| 4 | Clas | s Docu | mentation | 1 | 7 |
| | 4.1 | Blockir | ngQueue< | T > Class Template Reference | 7 |
| | | 4.1.1 | Detailed | Description | 8 |
| | | 4.1.2 | Construc | ctor & Destructor Documentation | 8 |
| | | | 4.1.2.1 | BlockingQueue() | 8 |
| | | | 4.1.2.2 | BlockingQueue(std::size_t capacity) | 8 |
| | | 4.1.3 | Member | Function Documentation | 9 |
| | | | 4.1.3.1 | Back() | 9 |
| | | | 4.1.3.2 | Empty() | 9 |
| | | | 4.1.3.3 | Front() | 9 |
| | | | 4.1.3.4 | Put(const T &task) | 9 |
| | | | 4.1.3.5 | SetCapacity(const size_t capacity) | 9 |
| | | | 4.1.3.6 | Size() | 10 |
| | | | 4.1.3.7 | Take() | 10 |
| | 4.2 | clientD | ata Class | Reference | 10 |
| | 4.3 | Concu | rrentDataS | Sharer Class Reference | 10 |
| | | 4.3.1 | Detailed | Description | 11 |

iv CONTENTS

| | 4.3.2 | Construc | etor & Destructor Documentation | 12 |
|-------|--------|-----------|---|--------------|
| | | 4.3.2.1 | ConcurrentDataSharer(std::string const &groupname, std::string const &multicastad 255.0.1"", std::string const &listenadress=""0.0.0.0"", const short multicast-port=30001) | ress=""239.< |
| | | 4.3.2.2 | ~ConcurrentDataSharer() | 12 |
| | 4.3.3 | Member | Function Documentation | 12 |
| | | 4.3.3.1 | get(std::string const &name) | 12 |
| | | 4.3.3.2 | get(std::string const &client, std::string const &name) | 12 |
| | | 4.3.3.3 | getClients() | 12 |
| | | 4.3.3.4 | getClientVariables(std::string const &client) | 13 |
| | | 4.3.3.5 | getMyName() | 13 |
| | | 4.3.3.6 | registerCallback(std::string const &name, CallbackSig func) | 13 |
| | | 4.3.3.7 | registerNewClientCallback(CallbackSig func) | 13 |
| | | 4.3.3.8 | set(std::string const &name, T data) | 13 |
| 4.4 | DataBa | aseElemer | nt Class Reference | 14 |
| 4.5 | Queue | ElementBa | ase Class Reference | 14 |
| 4.6 | Queue | ElementC | allback Class Reference | 15 |
| 4.7 | Queue | ElementG | et Class Reference | 16 |
| 4.8 | Queue | ElementM | ultiSend Class Reference | 17 |
| 4.9 | Queue | ElementSe | et Class Reference | 18 |
| 4.10 | Queue | ElementT(| CPSend Class Reference | 19 |
| Index | | | | 21 |

Chapter 1

README

ConcurrentDataSharer

2 README

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

| ckingQueue <t></t> | 7 |
|----------------------------------|----|
| ckingQueue< QueueElementBase * > | 7 |
| ntData | 10 |
| ncurrentDataSharer | 10 |
| aBaseElement | 14 |
| eueElementBase | 14 |
| QueueElementCallback | 15 |
| QueueElementGet | 16 |
| QueueElementMultiSend | 17 |
| QueueElementSet | 18 |
| QueueElementTCPSend | 19 |

4 Hierarchical Index

Chapter 3

Class Index

3.1 Class List

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

| BlockingQueue < T > | |
|--|----|
| _TCPSendQueue = new BlockingQueue <queueelementbase*>(255); _TCPSendQueue-</queueelementbase*> | |
| >put(new QueueElementBase()); | 7 |
| clientData | 10 |
| ConcurrentDataSharer | |
| ConcurrentDataSharer* sharer = new ConcurrentDataSharer("test"); sharer->set <int>("data",43);</int> | |
| sharer->set <int>("data1",42);</int> | 10 |
| DataBaseElement DataBaseElement | 14 |
| QueueElementBase | 14 |
| QueueElementCallback | 15 |
| QueueElementGet | 16 |
| QueueElementMultiSend | 17 |
| QueueElementSet | 18 |
| QueueElementTCPSend | 19 |

6 Class Index

Chapter 4

Class Documentation

4.1 BlockingQueue < T > Class Template Reference

```
_TCPSendQueue = new BlockingQueue<QueueElementBase*>(255); _TCPSendQueue->put(new Queue 
ElementBase());
#include <BlockingQueue.h>
```

Public Member Functions

```
• BlockingQueue ()
```

Constructor.

• BlockingQueue (std::size_t capacity)

Constructor.

void Put (const T &task)

put an element in the queue, blocks if full

• T Take ()

take an element from the queue, blocks if empty

T Front ()

returns the front element(been longest in the queue), does not pop, blocks if empty

• T Back ()

returns the back element(been shortest in the queue), does not pop, blocks if empty

• size_t Size ()

returns the number of elements in the queue

• bool Empty ()

returns true if the queue is empty

void SetCapacity (const size_t capacity)

sets the capacity of the queue

4.1.1 Detailed Description

```
\label{template} \begin{split} \text{template} &< \text{typename T} > \\ \text{class BlockingQueue} &< \text{T} > \end{split}
```

 $_TCPSendQueue = new BlockingQueue < QueueElementBase*>(255); <math>_TCPSendQueue->put(new Queue \leftarrow ElementBase());$

This is a blocking thread safe FIFO queue

Author

Martin Soderen

Version

0.1

Date

2017/04/147 00:00:00

Contact: martin.soderen@gmail.com

Created on: Fri Apr 14 00:00:00 2017

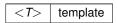
ld

4.1.2 Constructor & Destructor Documentation

4.1.2.1 template<typename T> BlockingQueue<T>::BlockingQueue() [inline]

Constructor.

Parameters



4.1.2.2 template<typename T> BlockingQueue< T>::BlockingQueue(std::size_t capacity) [inline]

Constructor.

Parameters

| < <i>T</i> > | template |
|--------------|--------------------------------|
| capacity | how many elements you can fill |

```
4.1.3
       Member Function Documentation
4.1.3.1 template < typename T > T BlockingQueue < T >::Back ( )
returns the back element(been shortest in the queue), does not pop, blocks if empty
Returns
     element
4.1.3.2 template<typename T > bool BlockingQueue< T >::Empty ( )
returns true if the queue is empty
Returns
     bool if empty
4.1.3.3 template<typename T > T BlockingQueue< T >::Front ( )
returns the front element(been longest in the queue), does not pop, blocks if empty
Returns
     element
4.1.3.4 template<typename T> void BlockingQueue< T>::Put (const T & task)
put an element in the queue, blocks if full
Parameters
 task
        element to put in queue
4.1.3.5 template<typename T > void BlockingQueue< T >::SetCapacity ( const size_t capacity )
sets the capacity of the queue
Parameters
 capacity
            the new capacity of the queue
```

```
4.1.3.6 template < typename T > size_t BlockingQueue < T >::Size ( )
returns the number of elements in the queue

Returns
    size of queue

4.1.3.7 template < typename T > T BlockingQueue < T >::Take ( )
take an element from the queue, blocks if empty

Returns
```

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

- /home/martin/repositories/ConcurrentDataSharer/include/BlockingQueue.h
- /home/martin/repositories/ConcurrentDataSharer/src/BlockingQueue.cpp

4.2 clientData Class Reference

Public Member Functions

element

- clientData (std::string name, std::vector< std::string > IPV4, std::vector< std::string > IPV6)
- std::vector< std::string > getIPV4 ()
- std::vector< std::string > getIPV6 ()
- std::string getName ()

Friends

· class boost::serialization::access

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

 $\bullet \ \ / home/martin/repositories/ConcurrentDataSharer/include/structures.h$

4.3 ConcurrentDataSharer Class Reference

```
\label{local_concurrent_data_sharer} \mbox{ConcurrentDataSharer("test"); sharer->set<int>("data",43); sharer->set<int>("data1",42);.}
```

#include <concurrentdatasharer.h>

Public Member Functions

 ConcurrentDataSharer (std::string const &groupname, std::string const &multicastadress="239.255.0.1", std::string const &listenadress="0.0.0.0", const short multicastport=30001)

Constructor.

∼ConcurrentDataSharer ()

Destructor.

template<typename T >

void set (std::string const &name, T data)

Set a local variable, creates it if it does not exists.

• template<typename T >

T get (std::string const &name)

Get a local variable.

• template<typename T >

T get (std::string const &client, std::string const &name)

Get a variable from another client.

std::vector< std::string > getClients ()

to get all connected clients

void registerNewClientCallback (CallbackSig func)

connects a callback when a new clients connects

• void registerCallback (std::string const &name, CallbackSig func)

connects a callback if a local variable changes

std::string getMyName ()

return the name of this client

std::vector< std::string > getClientVariables (std::string const &client)

get a list with the names of a clients variables

4.3.1 Detailed Description

This is the main class for the ConcurrentDataSharer

Note

This is the shit

Author

Martin Soderen

Version

0.1

Date

2017/04/147 00:00:00

Contact: martin.soderen@gmail.com

Created on: Fri Apr 14 00:00:00 2017

ld

4.3.2 Constructor & Destructor Documentation

4.3.2.1 ConcurrentDataSharer::ConcurrentDataSharer (std::string const & groupname, std::string const & multicastadress = "239.255.0.1", std::string const & listenadress = "0.0.0.0", const short multicastport = 30001)

Constructor.

Parameters

| groupname | the name for the ConcurrentDatasharer group to share data with |
|-----------------|--|
| multicastadress | the adress for UDP broadcasting, default is 239.255.0.1 |
| listenadress | the adress for UDP broadcasting listening, default is 0.0.0.0 |
| multicastport | the port for UDP broadcasting, default is 30001 |

4.3.2.2 ConcurrentDataSharer:: ∼ConcurrentDataSharer ()

Destructor.

Destructs objects

- 4.3.3 Member Function Documentation
- 4.3.3.1 template < typename T > T Concurrent Data Sharer::get (std::string const & name) [inline]

Get a local variable.

Parameters

| name | the name of the variable |
|------|--------------------------|

4.3.3.2 template<typename T > T ConcurrentDataSharer::get (std::string const & client, std::string const & name) [inline]

Get a variable from another client.

Parameters

| client | the name of the client |
|--------|--------------------------|
| name | the name of the variable |

 ${\it 4.3.3.3} \quad {\it std::} {\it vector} < {\it std::} {\it string} > {\it ConcurrentDataSharer::} {\it getClients} \; (\ \)$

to get all connected clients

Returns

std::vector<std::string> with the names of all the clients

4.3.3.4 std::vector < std::string > ConcurrentDataSharer::getClientVariables (std::string const & client)

get a list with the names of a clients variables

Parameters

| client | the name of the client |
|--------|------------------------|
|--------|------------------------|

Returns

a list with the names of a clients variables

4.3.3.5 std::string ConcurrentDataSharer::getMyName() [inline]

return the name of this client

Returns

the name of this client

4.3.3.6 void ConcurrentDataSharer::registerCallback (std::string const & name, CallbackSig func) [inline]

connects a callback if a local variable changes

Parameters

| name | the name of the variable |
|------|-------------------------------------|
| func | the function to connect with change |

 $\textbf{4.3.3.7} \quad \textbf{void ConcurrentDataSharer::} \textbf{registerNewClientCallback (CallbackSig} \textit{ func } \textbf{)} \quad \texttt{[inline]}$

connects a callback when a new clients connects

Parameters

func the function to connect

4.3.3.8 template<typename T > void ConcurrentDataSharer::set (std::string const & name, T data) [inline]

Set a local variable, creates it if it does not exists.

Parameters

| name | the name of the variable |
|------|--------------------------|
| data | the data |

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

- /home/martin/repositories/ConcurrentDataSharer/include/concurrentdatasharer.h
- /home/martin/repositories/ConcurrentDataSharer/src/concurrentdatasharer.cpp

4.4 DataBaseElement Class Reference

Public Member Functions

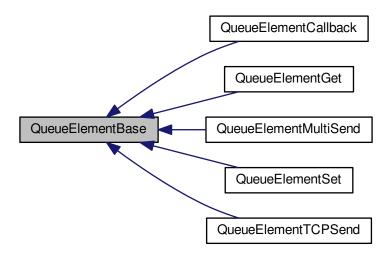
- DataBaseElement (std::string const &, std::string const &)
- DataBaseElement (QueueElementSet *)
- std::string getData ()
- · void setData (std::string const &data)
- void setCallback (CallbackSig func)
- void runCallback ()

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

- · /home/martin/repositories/ConcurrentDataSharer/include/structures.h
- /home/martin/repositories/ConcurrentDataSharer/src/structures.cpp

4.5 QueueElementBase Class Reference

Inheritance diagram for QueueElementBase:



Public Member Functions

- std::string getName ()
- std::string getData ()

Protected Attributes

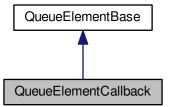
- std::string _name
- std::string _data

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

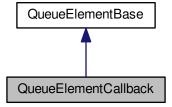
- /home/martin/repositories/ConcurrentDataSharer/include/structures.h
- /home/martin/repositories/ConcurrentDataSharer/src/structures.cpp

4.6 QueueElementCallback Class Reference

Inheritance diagram for QueueElementCallback:



Collaboration diagram for QueueElementCallback:



Public Member Functions

- QueueElementCallback (std::string const &name, CallbackSig func)
- CallbackSig getCallback ()

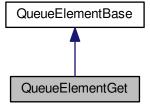
Additional Inherited Members

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

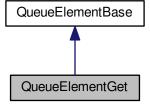
· /home/martin/repositories/ConcurrentDataSharer/include/structures.h

4.7 QueueElementGet Class Reference

Inheritance diagram for QueueElementGet:



Collaboration diagram for QueueElementGet:



Public Member Functions

- QueueElementGet (std::string const &)
- std::string getData ()
- void setData (std::string const &)

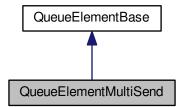
Additional Inherited Members

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

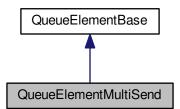
- /home/martin/repositories/ConcurrentDataSharer/include/structures.h
- · /home/martin/repositories/ConcurrentDataSharer/src/structures.cpp

4.8 QueueElementMultiSend Class Reference

Inheritance diagram for QueueElementMultiSend:



Collaboration diagram for QueueElementMultiSend:



Public Member Functions

- QueueElementMultiSend (std::string const &, std::string const &, MultiSend)
- MultiSend getPurpose ()

Friends

· class boost::serialization::access

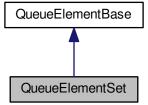
Additional Inherited Members

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

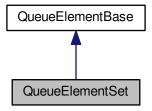
- · /home/martin/repositories/ConcurrentDataSharer/include/structures.h
- /home/martin/repositories/ConcurrentDataSharer/src/structures.cpp

4.9 QueueElementSet Class Reference

Inheritance diagram for QueueElementSet:



Collaboration diagram for QueueElementSet:



Public Member Functions

• QueueElementSet (std::string const &, std::string const &)

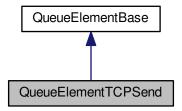
Additional Inherited Members

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

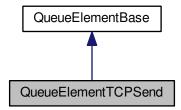
- /home/martin/repositories/ConcurrentDataSharer/include/structures.h
- /home/martin/repositories/ConcurrentDataSharer/src/structures.cpp

4.10 QueueElementTCPSend Class Reference

Inheritance diagram for QueueElementTCPSend:



Collaboration diagram for QueueElementTCPSend:



Public Member Functions

- QueueElementTCPSend (std::string const &name, std::string const &data, TCPSend purpose, bool respons)
- void setTag (std::string const &tag)
- std::string getTag ()
- bool getResponsRequired ()
- TCPSend getPurpose ()
- void setRequestor (std::string const &requestor)
- std::string getRequestor ()
- void setData (std::string const &data)
- std::string getData ()
- std::string getDataNoneBlocking ()

Friends

· class boost::serialization::access

Additional Inherited Members

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

• /home/martin/repositories/ConcurrentDataSharer/include/structures.h

Index

| ~ConcurrentDataSharer ConcurrentDataSharer, 12 |
|---|
| Back BlockingQueue, 9 BlockingQueue Back, 9 BlockingQueue, 8 Empty, 9 Front, 9 Put, 9 SetCapacity, 9 Size, 9 Take, 10 BlockingQueue < T >, 7 |
| clientData, 10 ConcurrentDataSharer, 10 ~ConcurrentDataSharer, 12 ConcurrentDataSharer, 12 get, 12 getClientVariables, 13 getClients, 12 getMyName, 13 registerCallback, 13 registerNewClientCallback, 13 set, 13 |
| DataBaseElement, 14 |
| Empty BlockingQueue, 9 |
| Front BlockingQueue, 9 |
| get ConcurrentDataSharer, 12 getClientVariables ConcurrentDataSharer, 13 getClients ConcurrentDataSharer, 12 getMyName ConcurrentDataSharer, 13 |
| Put BlockingQueue, 9 |
| QueueElementBase, 14 QueueElementCallback, 15 |

QueueElementGet, 16

```
QueueElementMultiSend, 17
QueueElementSet, 18
QueueElementTCPSend, 19
registerCallback
    ConcurrentDataSharer, 13
register New Client Callback\\
    ConcurrentDataSharer, 13
set
    ConcurrentDataSharer, 13
SetCapacity
    BlockingQueue, 9
Size
    BlockingQueue, 9
Take
    BlockingQueue, 10
```