

## 2015 Test beam Run Control

Generated by Doxygen 1.8.9.1

Wed Apr 15 2015 15:09:20



# Contents

<b>1</b>	<b>Hierarchical Index</b>	<b>1</b>
1.1	Class Hierarchy	1
<b>2</b>	<b>Data Structure Index</b>	<b>3</b>
2.1	Data Structures	3
<b>3</b>	<b>Data Structure Documentation</b>	<b>5</b>
3.1	Exception Class Reference	5
3.1.1	Detailed Description	5
3.1.2	Constructor & Destructor Documentation	5
3.1.2.1	Exception	5
3.1.2.2	Exception	5
3.1.2.3	~Exception	6
3.1.3	Member Function Documentation	6
3.1.3.1	Description	6
3.1.3.2	Dump	6
3.1.3.3	ErrorNumber	7
3.1.3.4	From	7
3.1.3.5	Type	7
3.1.3.6	TypeString	8
3.2	FPGAHandler Class Reference	8
3.2.1	Constructor & Destructor Documentation	9
3.2.1.1	FPGAHandler	9
3.2.1.2	~FPGAHandler	9
3.3	HTTPMessage Class Reference	9
3.3.1	Constructor & Destructor Documentation	10
3.3.1.1	HTTPMessage	11
3.3.1.2	HTTPMessage	11
3.3.2	Member Function Documentation	11
3.3.2.1	Decode	11
3.3.2.2	Dump	12
3.3.2.3	Encode	12

3.3.2.4	GetKey	12
3.4	Listener Class Reference	12
3.4.1	Detailed Description	13
3.4.2	Constructor & Destructor Documentation	13
3.4.2.1	Listener	13
3.4.2.2	Listener	13
3.4.2.3	~Listener	13
3.4.3	Member Function Documentation	13
3.4.3.1	Connect	13
3.4.3.2	Disconnect	13
3.4.3.3	Receive	14
3.4.3.4	Send	14
3.5	ListenerInfo Struct Reference	14
3.5.1	Field Documentation	14
3.5.1.1	name	14
3.5.1.2	type	14
3.6	Message Class Reference	14
3.6.1	Detailed Description	15
3.6.2	Constructor & Destructor Documentation	15
3.6.2.1	Message	15
3.6.2.2	Message	15
3.6.2.3	Message	15
3.6.2.4	~Message	15
3.6.3	Member Function Documentation	15
3.6.3.1	Dump	15
3.6.3.2	GetKey	15
3.6.3.3	GetString	15
3.6.3.4	IsFromWeb	16
3.6.4	Field Documentation	16
3.6.4.1	fString	16
3.7	Messenger Class Reference	16
3.7.1	Detailed Description	17
3.7.2	Constructor & Destructor Documentation	17
3.7.2.1	Messenger	17
3.7.2.2	Messenger	17
3.7.2.3	~Messenger	17
3.7.3	Member Function Documentation	17
3.7.3.1	Broadcast	17
3.7.3.2	Connect	17
3.7.3.3	Disconnect	17

3.7.3.4	<a href="#">DisconnectClient</a>	17
3.7.3.5	<a href="#">ProcessMessage</a>	18
3.7.3.6	<a href="#">Receive</a>	18
3.7.3.7	<a href="#">Send</a>	18
3.8	<a href="#">Socket Class Reference</a>	18
3.8.1	<a href="#">Detailed Description</a>	20
3.8.2	<a href="#">Constructor &amp; Destructor Documentation</a>	20
3.8.2.1	<a href="#">Socket</a>	20
3.8.2.2	<a href="#">Socket</a>	20
3.8.2.3	<a href="#">~Socket</a>	20
3.8.3	<a href="#">Member Function Documentation</a>	20
3.8.3.1	<a href="#">AcceptConnections</a>	20
3.8.3.2	<a href="#">Bind</a>	20
3.8.3.3	<a href="#">DumpConnected</a>	20
3.8.3.4	<a href="#">FetchMessage</a>	20
3.8.3.5	<a href="#">GetPort</a>	21
3.8.3.6	<a href="#">GetSocketId</a>	21
3.8.3.7	<a href="#">IsWebSocket</a>	21
3.8.3.8	<a href="#">Listen</a>	21
3.8.3.9	<a href="#">PrepareConnection</a>	21
3.8.3.10	<a href="#">SelectConnections</a>	21
3.8.3.11	<a href="#">SendMessage</a>	21
3.8.3.12	<a href="#">SetPort</a>	21
3.8.3.13	<a href="#">SetSocketId</a>	21
3.8.3.14	<a href="#">Start</a>	21
3.8.3.15	<a href="#">Stop</a>	21
3.8.4	<a href="#">Field Documentation</a>	21
3.8.4.1	<a href="#">fBuffer</a>	21
3.8.4.2	<a href="#">fMaster</a>	21
3.8.4.3	<a href="#">fPort</a>	22
3.8.4.4	<a href="#">fReadFds</a>	22
3.8.4.5	<a href="#">fSocketsConnected</a>	22
3.9	<a href="#">SocketMessage Class Reference</a>	22
3.9.1	<a href="#">Detailed Description</a>	23
3.9.2	<a href="#">Constructor &amp; Destructor Documentation</a>	23
3.9.2.1	<a href="#">SocketMessage</a>	23
3.9.2.2	<a href="#">SocketMessage</a>	23
3.9.2.3	<a href="#">SocketMessage</a>	23
3.9.2.4	<a href="#">SocketMessage</a>	23
3.9.2.5	<a href="#">SocketMessage</a>	24

3.9.2.6	SocketMessage	24
3.9.2.7	SocketMessage	24
3.9.2.8	SocketMessage	24
3.9.2.9	SocketMessage	25
3.9.2.10	SocketMessage	25
3.9.2.11	SocketMessage	25
3.9.2.12	~SocketMessage	25
3.9.3	Member Function Documentation	25
3.9.3.1	Dump	25
3.9.3.2	GetIntValue	25
3.9.3.3	GetKey	26
3.9.3.4	GetString	26
3.9.3.5	GetValue	26
3.9.3.6	GetVectorValue	26
3.9.3.7	SetKeyValue	26
3.9.3.8	SetKeyValue	27
3.9.3.9	SetKeyValue	27
3.9.3.10	SetKeyValue	27
3.9.3.11	SetKeyValue	28
<b>Index</b>		<b>29</b>

# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

Exception . . . . .	5
ListenerInfo . . . . .	14
Message . . . . .	14
HTTPMessage . . . . .	9
SocketMessage . . . . .	22
Socket . . . . .	18
Listener . . . . .	12
FPGAHandler . . . . .	8
Messenger . . . . .	16





## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">Exception</a>	
A simple exception handler . . . . .	5
<a href="#">FPGAHandler</a> . . . . .	8
<a href="#">HTTPMessage</a> . . . . .	9
<a href="#">Listener</a> . . . . .	12
<a href="#">ListenerInfo</a> . . . . .	14
<a href="#">Message</a>	
Base message type . . . . .	14
<a href="#">Messenger</a> . . . . .	16
<a href="#">Socket</a> . . . . .	18
<a href="#">SocketMessage</a>	
Socket-passed message type . . . . .	22



## Chapter 3

# Data Structure Documentation

### 3.1 Exception Class Reference

A simple exception handler.

```
#include <Exception.h>
```

#### Public Member Functions

- [Exception](#) (const char \*from, std::string desc, ExceptionType type=Undefined, const int id=0)
- [Exception](#) (const char \*from, const char \*desc, ExceptionType type=Undefined, const int id=0)
- [~Exception](#) ()
- std::string [From](#) () const
- int [ErrorNumber](#) () const
- std::string [Description](#) () const
- ExceptionType [Type](#) () const
- std::string [TypeString](#) () const
- void [Dump](#) (std::ostream &os=std::cerr) const

#### 3.1.1 Detailed Description

A simple exception handler.

Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

Date

24 Mar 2015

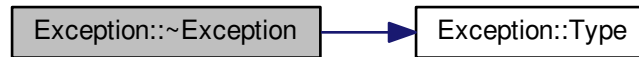
#### 3.1.2 Constructor & Destructor Documentation

**3.1.2.1** `Exception::Exception ( const char * from, std::string desc, ExceptionType type = Undefined, const int id = 0 )`  
[inline]

**3.1.2.2** `Exception::Exception ( const char * from, const char * desc, ExceptionType type = Undefined, const int id = 0 )`  
[inline]

### 3.1.2.3 `Exception::~~Exception ( ) [inline]`

Here is the call graph for this function:



## 3.1.3 Member Function Documentation

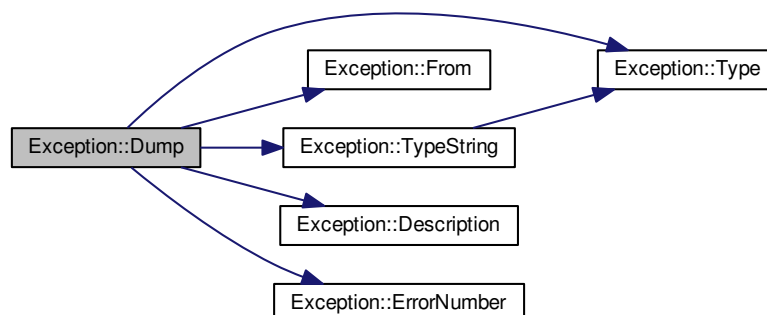
### 3.1.3.1 `std::string Exception::Description ( ) const [inline]`

Here is the caller graph for this function:



### 3.1.3.2 `void Exception::Dump ( std::ostream & os = std::cerr ) const [inline]`

Here is the call graph for this function:



### 3.1.3.3 `int Exception::ErrorNumber ( ) const [inline]`

Here is the caller graph for this function:



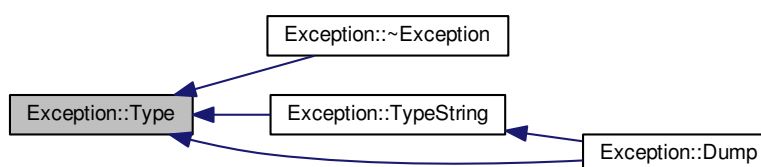
### 3.1.3.4 `std::string Exception::From ( ) const [inline]`

Here is the caller graph for this function:



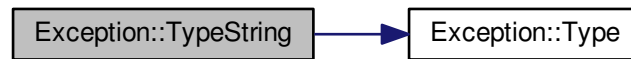
### 3.1.3.5 `ExceptionType Exception::Type ( ) const [inline]`

Here is the caller graph for this function:



### 3.1.3.6 `std::string Exception::TypeString ( ) const` `[inline]`

Here is the call graph for this function:



Here is the caller graph for this function:



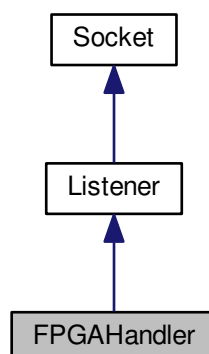
The documentation for this class was generated from the following file:

- `include/Exception.h`

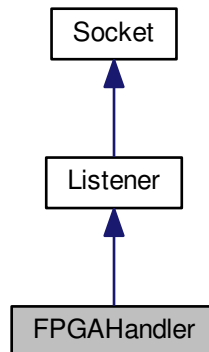
## 3.2 FPGAHandler Class Reference

```
#include <FPGAHandler.h>
```

Inheritance diagram for FPGAHandler:



Collaboration diagram for FPGAHandler:



## Public Member Functions

- [FPGAHandler](#) (const char \*dev)
- virtual [~FPGAHandler](#) ()

## Additional Inherited Members

### 3.2.1 Constructor & Destructor Documentation

3.2.1.1 `FPGAHandler::FPGAHandler ( const char * dev )`

3.2.1.2 `virtual FPGAHandler::~~FPGAHandler ( ) [virtual]`

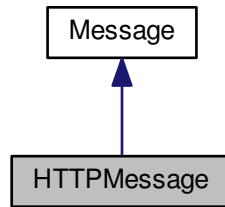
The documentation for this class was generated from the following file:

- include/FPGAHandler.h

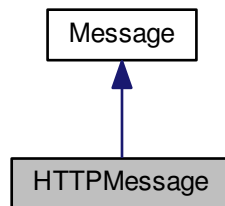
## 3.3 HTTPMessage Class Reference

```
#include <HTTPMessage.h>
```

Inheritance diagram for HTTPMessage:



Collaboration diagram for HTTPMessage:



## Public Member Functions

- [HTTPMessage](#) (WebSocket \*ws, [Message](#) m, MessageAction a)
- [HTTPMessage](#) (WebSocket \*ws, const char \*msg, MessageAction a)
- void [Decode](#) ()
- void [Encode](#) ()
- MessageKey [GetKey](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

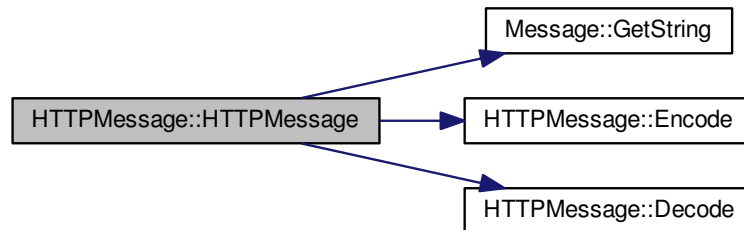
## Additional Inherited Members

### 3.3.1 Constructor & Destructor Documentation



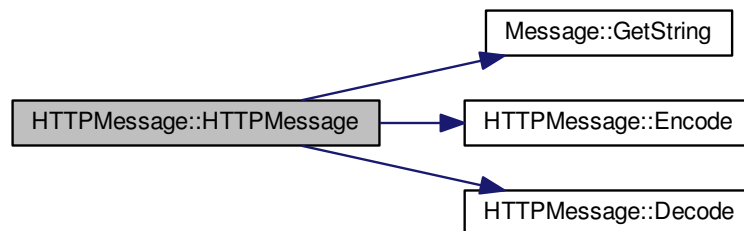
#### 3.3.1.1 HTTPMessage::HTTPMessage ( WebSocket \* *ws*, Message *m*, MessageAction *a* ) [inline]

Here is the call graph for this function:



#### 3.3.1.2 HTTPMessage::HTTPMessage ( WebSocket \* *ws*, const char \* *msg*, MessageAction *a* ) [inline]

Here is the call graph for this function:



### 3.3.2 Member Function Documentation

#### 3.3.2.1 void HTTPMessage::Decode ( ) [inline]

Here is the caller graph for this function:



3.3.2.2 `void HTTPMessage::Dump ( std::ostream & os = std::cout ) const` `[inline]`

3.3.2.3 `void HTTPMessage::Encode ( )` `[inline]`

Here is the caller graph for this function:



3.3.2.4 `MessageKey HTTPMessage::GetKey ( ) const` `[inline]`

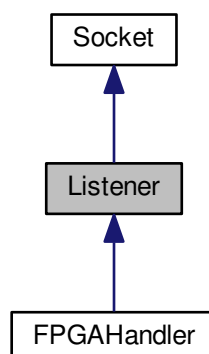
The documentation for this class was generated from the following file:

- `include/HTTPMessage.h`

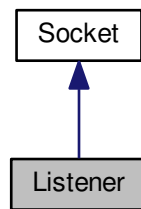
## 3.4 Listener Class Reference

```
#include <Listener.h>
```

Inheritance diagram for Listener:



Collaboration diagram for Listener:



### Public Member Functions

- [Listener](#) ()
- [Listener](#) (int port)
- [~Listener](#) ()
- bool [Connect](#) ()
- void [Disconnect](#) ()
- void [Send](#) (const [Message](#) &m) const
- void [Receive](#) () const

### Additional Inherited Members

#### 3.4.1 Detailed Description

Listener/client object used by the server to send/receive commands from the messenger/broadcaster.

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

24 Mar 2015

#### 3.4.2 Constructor & Destructor Documentation

3.4.2.1 `Listener::Listener ( )` `[inline]`

3.4.2.2 `Listener::Listener ( int port )`

3.4.2.3 `Listener::~~Listener ( )`

#### 3.4.3 Member Function Documentation

3.4.3.1 `bool Listener::Connect ( )`

3.4.3.2 `void Listener::Disconnect ( )`

3.4.3.3 void Listener::Receive ( ) const

3.4.3.4 void Listener::Send ( const Message & m ) const

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

- include/Listener.h

## 3.5 ListenerInfo Struct Reference

```
#include <Messenger.h>
```

### Data Fields

- std::string [name](#)
- int [type](#)

#### 3.5.1 Field Documentation

3.5.1.1 std::string ListenerInfo::name

3.5.1.2 int ListenerInfo::type

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

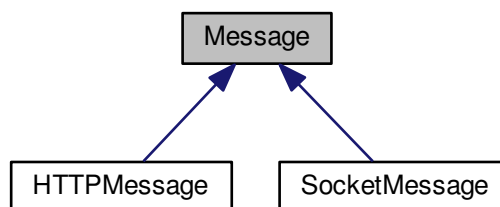
- include/Messenger.h

## 3.6 Message Class Reference

Base message type.

```
#include <Message.h>
```

Inheritance diagram for Message:



### Public Member Functions

- [Message](#) ()

- [Message](#) (const char \*msg)
- [Message](#) (std::string msg)
- [~Message](#) ()
- MessageKey [GetKey](#) () const
- std::string [GetString](#) () const
- bool [IsFromWeb](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

## Protected Attributes

- std::string [fString](#)

### 3.6.1 Detailed Description

Base message type.

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

6 Apr 2015

### 3.6.2 Constructor & Destructor Documentation

3.6.2.1 `Message::Message ( )` [inline]

3.6.2.2 `Message::Message ( const char * msg )` [inline]

3.6.2.3 `Message::Message ( std::string msg )` [inline]

3.6.2.4 `Message::~Message ( )` [inline]

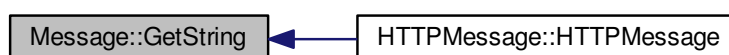
### 3.6.3 Member Function Documentation

3.6.3.1 `void Message::Dump ( std::ostream & os = std::cout ) const` [inline]

3.6.3.2 `MessageKey Message::GetKey ( ) const` [inline]

3.6.3.3 `std::string Message::GetString ( ) const` [inline]

Here is the caller graph for this function:



3.6.3.4 `bool Message::IsFromWeb ( ) const [inline]`

### 3.6.4 Field Documentation

3.6.4.1 `std::string Message::fString [protected]`

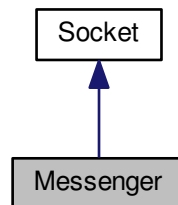
The documentation for this class was generated from the following file:

- `include/Message.h`

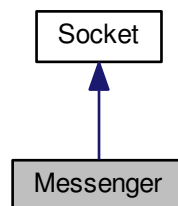
## 3.7 Messenger Class Reference

```
#include <Messenger.h>
```

Inheritance diagram for Messenger:



Collaboration diagram for Messenger:



### Public Member Functions

- [Messenger \( \)](#)
- [Messenger \(int port\)](#)
- [~Messenger \( \)](#)
- `bool` [Connect \( \)](#)
- `void` [Disconnect \( \)](#)

- void [DisconnectClient](#) (int sid, MessageKey key, bool force=false)  
*Disconnect a client.*
- void [Send](#) (const [Message](#) &m, int sid) const  
*Send any type of message to any client.*
- MessageKey [Receive](#) ()  
*Handle a message reception from a client.*
- void [ProcessMessage](#) ([SocketMessage](#) m, int sid)  
*Process a message received from the socket.*
- void [Broadcast](#) (const [Message](#) &m) const  
*Emit a message to all clients connected through the socket.*

## Additional Inherited Members

### 3.7.1 Detailed Description

Messenger/broadcaster object used by the server to send/receive commands from the clients/listeners.

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

23 Mar 2015

### 3.7.2 Constructor & Destructor Documentation

#### 3.7.2.1 Messenger::Messenger ( )

#### 3.7.2.2 Messenger::Messenger ( int port )

#### 3.7.2.3 Messenger::~~Messenger ( )

### 3.7.3 Member Function Documentation

#### 3.7.3.1 void Messenger::Broadcast ( const Message & m ) const

Emit a message to all clients connected through the socket.

#### Parameters

in	<i>m</i>	<a href="#">Message</a> to transmit
----	----------	-------------------------------------

#### 3.7.3.2 bool Messenger::Connect ( )

#### 3.7.3.3 void Messenger::Disconnect ( )

#### 3.7.3.4 void Messenger::DisconnectClient ( int sid, MessageKey key, bool force = false )

Disconnect a client.

Ask to a client to disconnect from this socket

**Parameters**

in	<i>sid</i>	Unique identifier of the client to disconnect
in	<i>key</i>	Key to the message to transmit for disconnection
in	<i>force</i>	Do we need to force the client out of this socket ?

**3.7.3.5 void Messenger::ProcessMessage ( SocketMessage m, int sid )**

Process a message received from the socket.

**Parameters**

in	<i>Unique</i>	identifier of the client sending the message
----	---------------	--

**3.7.3.6 MessageKey Messenger::Receive ( )**

Handle a message reception from a client.

**Returns**

The key to the message received if successfully parsed

**3.7.3.7 void Messenger::Send ( const Message & m, int sid ) const [inline]**

Send any type of message to any client.

**Parameters**

in	<i>m</i>	<a href="#">Message</a> to transmit
in	<i>sid</i>	Unique identifier of the client on this socket

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

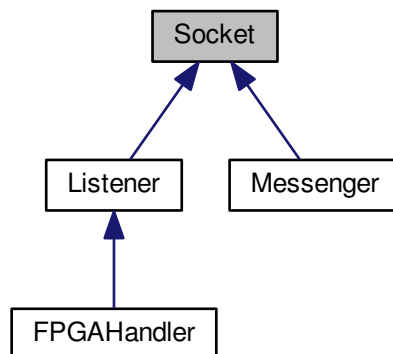
- include/Messenger.h

**3.8 Socket Class Reference**

```
#include <Socket.h>
```



Inheritance diagram for Socket:



### Public Member Functions

- [Socket](#) ()
- [Socket](#) (int port)
- virtual [~Socket](#) ()
- void [SetPort](#) (int port)
- int [GetPort](#) () const  
*Retrieve the port used for this socket.*
- void [AcceptConnections](#) ([Socket](#) &socket)  
*Accept connection from a client.*
- void [SelectConnections](#) ()
- void [SetSocketId](#) (int sid)
- int [GetSocketId](#) () const
- bool [IsWebSocket](#) (int sid) const
- void [DumpConnected](#) () const

### Protected Member Functions

- bool [Start](#) ()  
*Start the socket.*
- void [Stop](#) ()  
*Terminates the socket and all attached communications.*
- void [Bind](#) ()  
*Bind a name to a socket.*
- void [PrepareConnection](#) ()
- void [Listen](#) (int maxconn)  
*Listen to incoming messages.*
- void [SendMessage](#) ([Message](#) message, int id=-1) const  
*Send a message on a socket.*
- [Message](#) [FetchMessage](#) (int id=-1) const  
*Receive a message from a socket.*

## Protected Attributes

- int `fPort`
- char `fBuffer` [MAX\_WORD\_LENGTH]
- SocketCollection `fSocketsConnected`
- fd\_set `fMaster`  
Master file descriptor list.
- fd\_set `fReadFds`  
Temp file descriptor list for select()

### 3.8.1 Detailed Description

General object providing all useful method to connect/bind/send/receive information through system sockets.

#### Author

Laurent Forthomme `laurent.forthomme@cern.ch`

#### Date

23 Mar 2015

### 3.8.2 Constructor & Destructor Documentation

3.8.2.1 `Socket::Socket ( )` [inline]

3.8.2.2 `Socket::Socket ( int port )`

3.8.2.3 `virtual Socket::~~Socket ( )` [virtual]

### 3.8.3 Member Function Documentation

3.8.3.1 `void Socket::AcceptConnections ( Socket & socket )`

Accept connection from a client.

Set the socket to accept connections any client transmitting through the socket

#### Parameters

<code>in, out</code>	<code>socket</code>	Master/client object to enable on the socket
----------------------	---------------------	--

3.8.3.2 `void Socket::Bind ( )` [protected]

Bind a name to a socket.

#### Returns

Success of the operation

3.8.3.3 `void Socket::DumpConnected ( )` const

3.8.3.4 `Message Socket::FetchMessage ( int id = -1 )` const [protected]

Receive a message from a socket.

## Returns

Received message as a `std::string`

**3.8.3.5** `int Socket::GetPort ( ) const` `[inline]`

Retrieve the port used for this socket.

**3.8.3.6** `int Socket::GetSocketId ( ) const` `[inline]`

**3.8.3.7** `bool Socket::IsWebSocket ( int sid ) const` `[inline]`

**3.8.3.8** `void Socket::Listen ( int maxconn )` `[protected]`

Listen to incoming messages.

Set the socket to listen to any message coming from outside

**3.8.3.9** `void Socket::PrepareConnection ( )` `[protected]`

**3.8.3.10** `void Socket::SelectConnections ( )`

Register all open file descriptors to read their communication through the socket

**3.8.3.11** `void Socket::SendMessage ( Message message, int id = -1 ) const` `[protected]`

Send a message on a socket.

**3.8.3.12** `void Socket::SetPort ( int port )` `[inline]`

**3.8.3.13** `void Socket::SetSocketId ( int sid )` `[inline]`

**3.8.3.14** `bool Socket::Start ( )` `[protected]`

Start the socket.

Launch all mandatory operations to set the socket to be used

## Returns

Success of the operation

**3.8.3.15** `void Socket::Stop ( )` `[protected]`

Terminates the socket and all attached communications.

## 3.8.4 Field Documentation

**3.8.4.1** `char Socket::fBuffer[MAX_WORD_LENGTH]` `[protected]`

**3.8.4.2** `fd_set Socket::fMaster` `[protected]`

Master file descriptor list.

3.8.4.3 `int Socket::fPort` `[protected]`

3.8.4.4 `fd_set Socket::fReadFds` `[protected]`

Temp file descriptor list for select()

3.8.4.5 `SocketCollection Socket::fSocketsConnected` `[protected]`

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

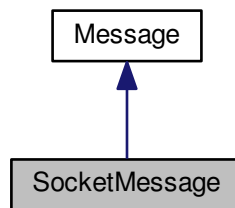
- `include/Socket.h`

## 3.9 SocketMessage Class Reference

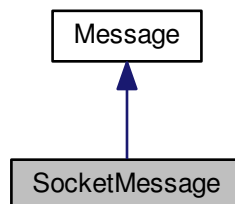
Socket-passed message type.

```
#include <SocketMessage.h>
```

Inheritance diagram for SocketMessage:



Collaboration diagram for SocketMessage:



### Public Member Functions

- [SocketMessage](#) ()

- [SocketMessage](#) ([Message](#) msg)
- [SocketMessage](#) (const char \*msg\_s)
- [SocketMessage](#) (std::string msg\_s)
- [SocketMessage](#) (MessageKey key)
- [SocketMessage](#) (MessageKey key, const char \*value)
- [SocketMessage](#) (MessageKey key, std::string value)
- [SocketMessage](#) (MessageKey key, const int value)
- [SocketMessage](#) (MessageKey key, const float value)
- [SocketMessage](#) (MessageKey key, const double value)
- [SocketMessage](#) (MessageMap msg\_m)
- [~SocketMessage](#) ()
- void [SetKeyValue](#) (MessageKey key, std::string value)  
*Send a string-valued message.*
- void [SetKeyValue](#) (MessageKey key, const char \*value)
- void [SetKeyValue](#) (MessageKey key, int int\_value)  
*Send an integer-valued message.*
- void [SetKeyValue](#) (MessageKey key, float float\_value)  
*Send an float-valued message.*
- void [SetKeyValue](#) (MessageKey key, double double\_value)  
*Send an double-valued message.*
- std::string [GetString](#) () const
- MessageKey [GetKey](#) () const
- std::string [GetValue](#) () const
- int [GetIntValue](#) () const
- VectorValue [GetVectorValue](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

## Additional Inherited Members

### 3.9.1 Detailed Description

Socket-passed message type.

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

26 Mar 2015

### 3.9.2 Constructor & Destructor Documentation

3.9.2.1 [SocketMessage::SocketMessage \( \)](#) `[inline]`

3.9.2.2 [SocketMessage::SocketMessage \( Message msg \)](#) `[inline]`

3.9.2.3 [SocketMessage::SocketMessage \( const char \\* msg\\_s \)](#) `[inline]`

3.9.2.4 [SocketMessage::SocketMessage \( std::string msg\\_s \)](#) `[inline]`

### 3.9.2.5 SocketMessage::SocketMessage ( MessageKey *key* ) [inline]

Here is the call graph for this function:



### 3.9.2.6 SocketMessage::SocketMessage ( MessageKey *key*, const char \* *value* ) [inline]

Here is the call graph for this function:



### 3.9.2.7 SocketMessage::SocketMessage ( MessageKey *key*, std::string *value* ) [inline]

Here is the call graph for this function:



### 3.9.2.8 SocketMessage::SocketMessage ( MessageKey *key*, const int *value* ) [inline]

Here is the call graph for this function:



### 3.9.2.9 SocketMessage::SocketMessage ( MessageKey *key*, const float *value* ) [inline]

Here is the call graph for this function:



### 3.9.2.10 SocketMessage::SocketMessage ( MessageKey *key*, const double *value* ) [inline]

Here is the call graph for this function:



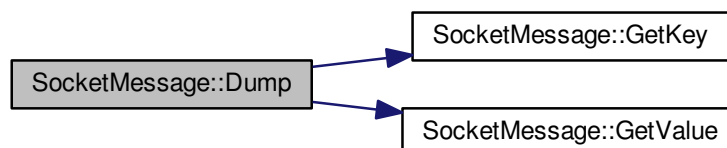
### 3.9.2.11 SocketMessage::SocketMessage ( MessageMap *msg\_m* ) [inline]

### 3.9.2.12 SocketMessage::~~SocketMessage ( ) [inline]

## 3.9.3 Member Function Documentation

### 3.9.3.1 void SocketMessage::Dump ( std::ostream & *os* = std::cout ) const [inline]

Here is the call graph for this function:



### 3.9.3.2 int SocketMessage::GetIntValue ( ) const [inline]

### 3.9.3.3 MessageKey SocketMessage::GetKey ( ) const [inline]

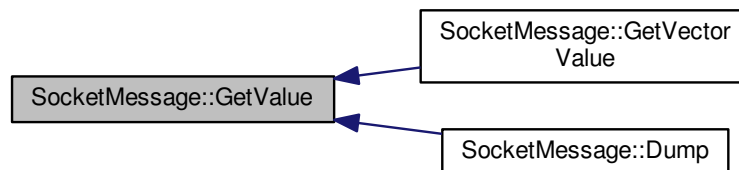
Here is the caller graph for this function:



### 3.9.3.4 std::string SocketMessage::GetString ( ) const [inline]

### 3.9.3.5 std::string SocketMessage::GetValue ( ) const [inline]

Here is the caller graph for this function:



### 3.9.3.6 VectorValue SocketMessage::GetVectorValue ( ) const [inline]

Here is the call graph for this function:

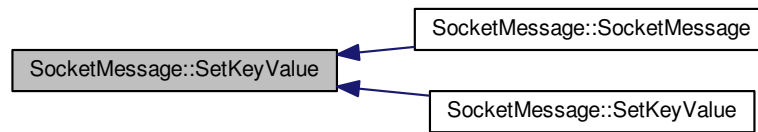


### 3.9.3.7 void SocketMessage::SetKeyValue ( MessageKey key, std::string value ) [inline]

Send a string-valued message.



Here is the caller graph for this function:



**3.9.3.8** `void SocketMessage::SetKeyValue ( MessageKey key, const char * value ) [inline]`

Here is the call graph for this function:



**3.9.3.9** `void SocketMessage::SetKeyValue ( MessageKey key, int int_value ) [inline]`

Send an integer-valued message.

Here is the call graph for this function:



**3.9.3.10** `void SocketMessage::SetKeyValue ( MessageKey key, float float_value ) [inline]`

Send an float-valued message.

Here is the call graph for this function:



3.9.3.11 `void SocketMessage::SetKeyValue ( MessageKey key, double double_value )` `[inline]`

Send an double-valued message.

Here is the call graph for this function:



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

- `include/SocketMessage.h`

# Index

- ~Exception
  - Exception, [5](#)
- ~FPGAHandler
  - FPGAHandler, [9](#)
- ~Listener
  - Listener, [13](#)
- ~Message
  - Message, [15](#)
- ~Messenger
  - Messenger, [17](#)
- ~Socket
  - Socket, [20](#)
- ~SocketMessage
  - SocketMessage, [25](#)
- AcceptConnections
  - Socket, [20](#)
- Bind
  - Socket, [20](#)
- Broadcast
  - Messenger, [17](#)
- Connect
  - Listener, [13](#)
  - Messenger, [17](#)
- Decode
  - HTTPMessage, [11](#)
- Description
  - Exception, [6](#)
- Disconnect
  - Listener, [13](#)
  - Messenger, [17](#)
- DisconnectClient
  - Messenger, [17](#)
- Dump
  - Exception, [6](#)
  - HTTPMessage, [11](#)
  - Message, [15](#)
  - SocketMessage, [25](#)
- DumpConnected
  - Socket, [20](#)
- Encode
  - HTTPMessage, [12](#)
- ErrorNumber
  - Exception, [6](#)
- Exception, [5](#)
  - ~Exception, [5](#)
  - Description, [6](#)
  - Dump, [6](#)
  - ErrorNumber, [6](#)
  - Exception, [5](#)
  - From, [7](#)
  - Type, [7](#)
  - TypeString, [7](#)
- fBuffer
  - Socket, [21](#)
- fMaster
  - Socket, [21](#)
- FPGAHandler, [8](#)
  - ~FPGAHandler, [9](#)
  - FPGAHandler, [9](#)
- fPort
  - Socket, [21](#)
- fReadFds
  - Socket, [22](#)
- fSocketsConnected
  - Socket, [22](#)
- fString
  - Message, [16](#)
- FetchMessage
  - Socket, [20](#)
- From
  - Exception, [7](#)
- GetIntValue
  - SocketMessage, [25](#)
- GetKey
  - HTTPMessage, [12](#)
  - Message, [15](#)
  - SocketMessage, [25](#)
- GetPort
  - Socket, [21](#)
- GetSocketId
  - Socket, [21](#)
- GetString
  - Message, [15](#)
  - SocketMessage, [26](#)
- GetValue
  - SocketMessage, [26](#)
- GetVectorValue
  - SocketMessage, [26](#)
- HTTPMessage, [9](#)
  - Decode, [11](#)
  - Dump, [11](#)
  - Encode, [12](#)
  - GetKey, [12](#)

- HTTPMessage, 10, 11
- IsFromWeb
  - Message, 15
- IsWebSocket
  - Socket, 21
- Listen
  - Socket, 21
- Listener, 12
  - ~Listener, 13
  - Connect, 13
  - Disconnect, 13
  - Listener, 13
  - Receive, 13
  - Send, 14
- ListenerInfo, 14
  - name, 14
  - type, 14
- Message, 14
  - ~Message, 15
  - Dump, 15
  - fString, 16
  - GetKey, 15
  - GetString, 15
  - IsFromWeb, 15
  - Message, 15
- Messenger, 16
  - ~Messenger, 17
  - Broadcast, 17
  - Connect, 17
  - Disconnect, 17
  - DisconnectClient, 17
  - Messenger, 17
  - ProcessMessage, 18
  - Receive, 18
  - Send, 18
- name
  - ListenerInfo, 14
- PrepareConnection
  - Socket, 21
- ProcessMessage
  - Messenger, 18
- Receive
  - Listener, 13
  - Messenger, 18
- SelectConnections
  - Socket, 21
- Send
  - Listener, 14
  - Messenger, 18
- SendMessage
  - Socket, 21
- SetKeyValue
  - SocketMessage, 26–28
- SetPort
  - Socket, 21
- SetSocketId
  - Socket, 21
- Socket, 18
  - ~Socket, 20
  - AcceptConnections, 20
  - Bind, 20
  - DumpConnected, 20
  - fBuffer, 21
  - fMaster, 21
  - fPort, 21
  - fReadFds, 22
  - fSocketsConnected, 22
  - FetchMessage, 20
  - GetPort, 21
  - GetSocketId, 21
  - IsWebSocket, 21
  - Listen, 21
  - PrepareConnection, 21
  - SelectConnections, 21
  - SendMessage, 21
  - SetPort, 21
  - SetSocketId, 21
  - Socket, 20
  - Start, 21
  - Stop, 21
- SocketMessage, 22
  - ~SocketMessage, 25
  - Dump, 25
  - GetIntValue, 25
  - GetKey, 25
  - GetString, 26
  - GetValue, 26
  - GetVectorValue, 26
  - SetKeyValue, 26–28
  - SocketMessage, 23–25
- Start
  - Socket, 21
- Stop
  - Socket, 21
- Type
  - Exception, 7
- type
  - ListenerInfo, 14
- TypeString
  - Exception, 7