

## 2015 Test beam Run Control

Generated by Doxygen 1.8.9.1

Mon Apr 20 2015 19:22:21



# 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	Client Class Reference	5
3.1.1	Detailed Description	6
3.1.2	Constructor & Destructor Documentation	6
3.1.2.1	Client	6
3.1.2.2	Client	6
3.1.2.3	~Client	6
3.1.3	Member Function Documentation	6
3.1.3.1	Connect	6
3.1.3.2	Disconnect	6
3.1.3.3	GetType	6
3.1.3.4	ParseMessage	6
3.1.3.5	Receive	6
3.1.3.6	Send	7
3.2	Exception Class Reference	7
3.2.1	Detailed Description	7
3.2.2	Constructor & Destructor Documentation	7
3.2.2.1	Exception	7
3.2.2.2	Exception	8
3.2.2.3	~Exception	8
3.2.3	Member Function Documentation	8
3.2.3.1	Description	8
3.2.3.2	Dump	8
3.2.3.3	ErrorNumber	9
3.2.3.4	From	9
3.2.3.5	Type	9

3.2.3.6	TypeString . . . . .	10
3.3	file_header_t Struct Reference . . . . .	10
3.3.1	Field Documentation . . . . .	10
3.3.1.1	magic . . . . .	10
3.3.1.2	run_id . . . . .	10
3.3.1.3	spill_id . . . . .	10
3.4	FPGAHandler Class Reference . . . . .	11
3.4.1	Detailed Description . . . . .	12
3.4.2	Constructor & Destructor Documentation . . . . .	12
3.4.2.1	FPGAHandler . . . . .	12
3.4.2.2	~FPGAHandler . . . . .	12
3.4.3	Member Function Documentation . . . . .	12
3.4.3.1	GetFilename . . . . .	12
3.4.3.2	GetType . . . . .	12
3.4.3.3	OpenFile . . . . .	12
3.4.3.4	ReadBuffer . . . . .	12
3.4.3.5	ReadConfiguration . . . . .	12
3.4.3.6	SendConfiguration . . . . .	12
3.5	HTTPMessage Class Reference . . . . .	12
3.5.1	Constructor & Destructor Documentation . . . . .	13
3.5.1.1	HTTPMessage . . . . .	14
3.5.1.2	HTTPMessage . . . . .	14
3.5.2	Member Function Documentation . . . . .	14
3.5.2.1	Decode . . . . .	14
3.5.2.2	Dump . . . . .	15
3.5.2.3	Encode . . . . .	15
3.5.2.4	GetKey . . . . .	15
3.6	ListenerInfo Struct Reference . . . . .	15
3.6.1	Field Documentation . . . . .	15
3.6.1.1	name . . . . .	15
3.6.1.2	type . . . . .	15
3.7	Message Class Reference . . . . .	15
3.7.1	Detailed Description . . . . .	16
3.7.2	Constructor & Destructor Documentation . . . . .	16
3.7.2.1	Message . . . . .	16
3.7.2.2	Message . . . . .	16
3.7.2.3	Message . . . . .	16
3.7.2.4	~Message . . . . .	16
3.7.3	Member Function Documentation . . . . .	17
3.7.3.1	Dump . . . . .	17

3.7.3.2	GetKey	17
3.7.3.3	GetString	17
3.7.3.4	IsFromWeb	17
3.7.4	Field Documentation	17
3.7.4.1	fString	17
3.8	Messenger Class Reference	17
3.8.1	Detailed Description	18
3.8.2	Constructor & Destructor Documentation	18
3.8.2.1	Messenger	18
3.8.2.2	Messenger	18
3.8.2.3	~Messenger	19
3.8.3	Member Function Documentation	19
3.8.3.1	Broadcast	19
3.8.3.2	Connect	19
3.8.3.3	Disconnect	19
3.8.3.4	Receive	19
3.8.3.5	Send	19
3.9	Socket Class Reference	19
3.9.1	Detailed Description	21
3.9.2	Constructor & Destructor Documentation	21
3.9.2.1	Socket	21
3.9.2.2	Socket	21
3.9.2.3	~Socket	21
3.9.3	Member Function Documentation	21
3.9.3.1	AcceptConnections	21
3.9.3.2	Bind	21
3.9.3.3	DumpConnected	21
3.9.3.4	FetchMessage	21
3.9.3.5	GetPort	22
3.9.3.6	GetSocketId	22
3.9.3.7	GetSocketType	22
3.9.3.8	IsWebSocket	22
3.9.3.9	Listen	22
3.9.3.10	PrepareConnection	22
3.9.3.11	SelectConnections	22
3.9.3.12	SendMessage	23
3.9.3.13	SetPort	23
3.9.3.14	SetSocketId	23
3.9.3.15	Start	23
3.9.3.16	Stop	23

3.9.4	Field Documentation	23
3.9.4.1	fBuffer	23
3.9.4.2	fMaster	23
3.9.4.3	fPort	23
3.9.4.4	fReadFds	23
3.9.4.5	fSocketsConnected	23
3.10	SocketMessage Class Reference	24
3.10.1	Detailed Description	25
3.10.2	Constructor & Destructor Documentation	25
3.10.2.1	SocketMessage	25
3.10.2.2	SocketMessage	25
3.10.2.3	SocketMessage	25
3.10.2.4	SocketMessage	25
3.10.2.5	SocketMessage	25
3.10.2.6	SocketMessage	26
3.10.2.7	SocketMessage	26
3.10.2.8	SocketMessage	26
3.10.2.9	SocketMessage	26
3.10.2.10	SocketMessage	27
3.10.2.11	SocketMessage	27
3.10.2.12	~SocketMessage	27
3.10.3	Member Function Documentation	27
3.10.3.1	Dump	27
3.10.3.2	GetIntValue	27
3.10.3.3	GetKey	27
3.10.3.4	GetString	28
3.10.3.5	GetValue	28
3.10.3.6	GetVectorValue	28
3.10.3.7	SetKeyValue	28
3.10.3.8	SetKeyValue	29
3.10.3.9	SetKeyValue	29
3.10.3.10	SetKeyValue	29
3.10.3.11	SetKeyValue	29
3.11	TDCConfiguration Class Reference	30
3.11.1	Detailed Description	31
3.11.2	Member Enumeration Documentation	31
3.11.2.1	DeadTime	31
3.11.2.2	EdgeResolution	31
3.11.2.3	WidthResolution	31
3.11.3	Constructor & Destructor Documentation	32

3.11.3.1	TDCCConfiguration	32
3.11.3.2	~TDCCConfiguration	32
3.11.4	Member Function Documentation	32
3.11.4.1	Dump	32
3.11.4.2	GetChannelOffset	32
3.11.4.3	GetDeadTime	32
3.11.4.4	GetEdgeResolution	32
3.11.4.5	GetEdgesPairing	32
3.11.4.6	GetLeadingMode	32
3.11.4.7	GetTrailingMode	32
3.11.4.8	GetTriggerMatchingMode	32
3.11.4.9	GetWidthResolution	32
3.11.4.10	SetAllChannelsOffset	32
3.11.4.11	SetChannelOffset	33
3.11.4.12	SetDeadTime	33
3.11.4.13	SetEdgeResolution	33
3.11.4.14	SetEdgesPairing	33
3.11.4.15	SetLeadingMode	33
3.11.4.16	SetTrailingMode	33
3.11.4.17	SetTriggerMatchingMode	33
3.11.4.18	SetWidthResolution	33
<b>Index</b>		<b>35</b>





# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

Exception . . . . .	7
file_header_t . . . . .	10
ListenerInfo . . . . .	15
Message . . . . .	15
HTTPMessage . . . . .	12
SocketMessage . . . . .	24
Socket . . . . .	19
Client . . . . .	5
FPGAHandler . . . . .	11
Messenger . . . . .	17
TDCConfiguration . . . . .	30



## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">Client</a> . . . . .	5
<a href="#">Exception</a>	
A simple exception handler . . . . .	7
<a href="#">file_header_t</a> . . . . .	10
<a href="#">FPGAHandler</a> . . . . .	11
<a href="#">HTTPMessage</a> . . . . .	12
<a href="#">ListenerInfo</a> . . . . .	15
<a href="#">Message</a>	
Base message type . . . . .	15
<a href="#">Messenger</a> . . . . .	17
<a href="#">Socket</a> . . . . .	19
<a href="#">SocketMessage</a>	
Socket-passed message type . . . . .	24
<a href="#">TDCCConfiguration</a> . . . . .	30



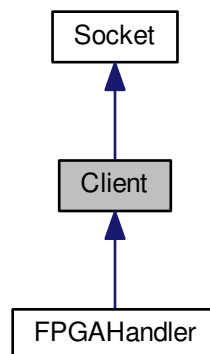
## Chapter 3

# Data Structure Documentation

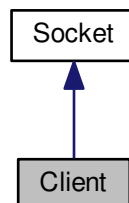
### 3.1 Client Class Reference

```
#include <Client.h>
```

Inheritance diagram for Client:



Collaboration diagram for Client:



## Public Member Functions

- [Client](#) ()
- [Client](#) (int port)
- [~Client](#) ()
- bool [Connect](#) ()
- void [Disconnect](#) ()
- void [Send](#) (const [Message](#) &m) const
- void [Receive](#) ()
- virtual void [ParseMessage](#) (const [SocketMessage](#) &m)
- virtual SocketType [GetType](#) () const

## Additional Inherited Members

### 3.1.1 Detailed Description

[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

### 3.1.2 Constructor & Destructor Documentation

3.1.2.1 [Client::Client](#) ( ) `[inline]`

3.1.2.2 [Client::Client](#) ( int *port* )

3.1.2.3 [Client::~~Client](#) ( )

### 3.1.3 Member Function Documentation

3.1.3.1 bool [Client::Connect](#) ( )

3.1.3.2 void [Client::Disconnect](#) ( )

3.1.3.3 virtual SocketType [Client::GetType](#) ( ) const `[inline],[virtual]`

Reimplemented in [FPGAHandler](#).

3.1.3.4 virtual void [Client::ParseMessage](#) ( const [SocketMessage](#) & *m* ) `[inline],[virtual]`

3.1.3.5 void [Client::Receive](#) ( )

3.1.3.6 `void Client::Send ( const Message & m ) const` `[inline]`

Here is the call graph for this function:



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

- `include/Client.h`

## 3.2 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

### 3.2.1 Detailed Description

A simple exception handler.

#### Author

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

#### Date

24 Mar 2015

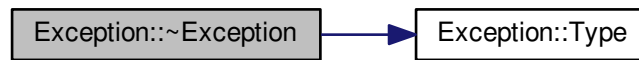
### 3.2.2 Constructor & Destructor Documentation

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

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

**3.2.2.3** `Exception::~~Exception ( )` `[inline]`

Here is the call graph for this function:



### 3.2.3 Member Function Documentation

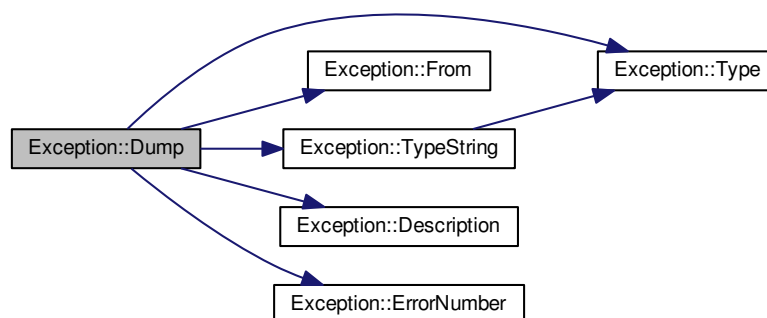
**3.2.3.1** `std::string Exception::Description ( ) const` `[inline]`

Here is the caller graph for this function:



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

Here is the call graph for this function:





3.2.3.3 `int Exception::ErrorNumber ( ) const [inline]`

Here is the caller graph for this function:



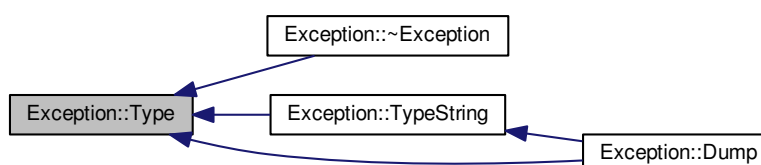
3.2.3.4 `std::string Exception::From ( ) const [inline]`

Here is the caller graph for this function:



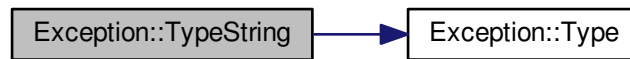
3.2.3.5 `ExceptionType Exception::Type ( ) const [inline]`

Here is the caller graph for this function:



### 3.2.3.6 `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`

## 3.3 `file_header_t` Struct Reference

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

### Data Fields

- `uint32_t magic`
- `uint32_t run_id`
- `uint32_t spill_id`

### 3.3.1 Field Documentation

3.3.1.1 `uint32_t file_header_t::magic`

3.3.1.2 `uint32_t file_header_t::run_id`

3.3.1.3 `uint32_t file_header_t::spill_id`

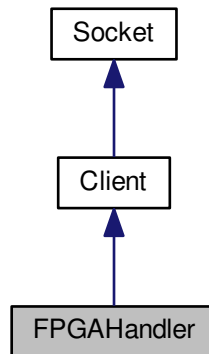
The documentation for this struct was generated from the following file:

- `include/FPGAHandler.h`

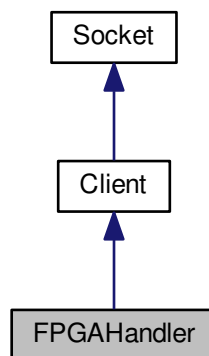
## 3.4 FPGAHandler Class Reference

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

Inheritance diagram for FPGAHandler:



Collaboration diagram for FPGAHandler:



### Public Member Functions

- [FPGAHandler](#) (int port, const char \*dev)
- virtual [~FPGAHandler](#) ()
- void [OpenFile](#) ()
- std::string [GetFilename](#) () const
- void [SendConfiguration](#) (const [TDCCConfiguration](#) &c)
- [TDCCConfiguration](#) [ReadConfiguration](#) ()
- void [ReadBuffer](#) ()
- SocketType [GetType](#) () const

## Additional Inherited Members

### 3.4.1 Detailed Description

#### Author

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

#### Date

14 Apr 2015

### 3.4.2 Constructor & Destructor Documentation

3.4.2.1 `FPGAHandler::FPGAHandler ( int port, const char * dev )`

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

### 3.4.3 Member Function Documentation

3.4.3.1 `std::string FPGAHandler::GetFilename ( )` `const` `[inline]`

3.4.3.2 `SocketType FPGAHandler::GetType ( )` `const` `[inline]`, `[virtual]`

Reimplemented from [Client](#).

3.4.3.3 `void FPGAHandler::OpenFile ( )`

3.4.3.4 `void FPGAHandler::ReadBuffer ( )`

3.4.3.5 `TDCConfiguration FPGAHandler::ReadConfiguration ( )`

3.4.3.6 `void FPGAHandler::SendConfiguration ( const TDCConfiguration & c )`

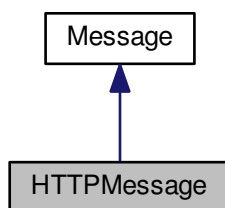
The documentation for this class was generated from the following file:

- `include/FPGAHandler.h`

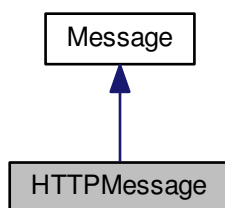
## 3.5 HTTPMessage Class Reference

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

Inheritance diagram for HTTPMessage:



Collaboration diagram for HTTPMessage:



## Public Member Functions

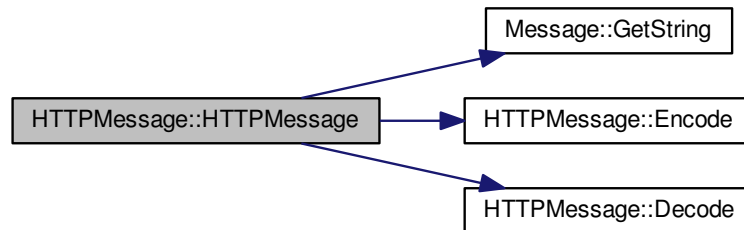
- [HTTPMessage](#) (WebSocket \*ws, [Message](#) m, MessageAction a)
- [HTTPMessage](#) (WebSocket \*ws, const char \*msg, MessageAction a)
- void [Decode](#) ()
- void [Encode](#) ()
- MessageKey [GetKey](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

## Additional Inherited Members

### 3.5.1 Constructor & Destructor Documentation

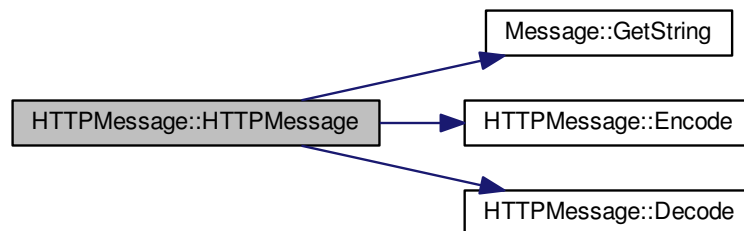
### 3.5.1.1 HTTPMessage::HTTPMessage ( WebSocket \* *ws*, Message *m*, MessageAction *a* ) [inline]

Here is the call graph for this function:



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

Here is the call graph for this function:



## 3.5.2 Member Function Documentation

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

Here is the caller graph for this function:



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

3.5.2.3 void HTTPMessage::Encode ( ) [inline]

Here is the caller graph for this function:



3.5.2.4 MessageKey HTTPMessage::GetKey ( ) const [inline]

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

- include/HTTPMessage.h

## 3.6 ListenerInfo Struct Reference

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

### Data Fields

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

### 3.6.1 Field Documentation

3.6.1.1 std::string ListenerInfo::name

3.6.1.2 SocketType ListenerInfo::type

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

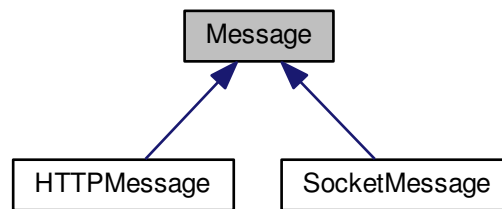
- include/Messenger.h

## 3.7 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`

### 3.7.1 Detailed Description

Base message type.

Author

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

Date

6 Apr 2015

### 3.7.2 Constructor & Destructor Documentation

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

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

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

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



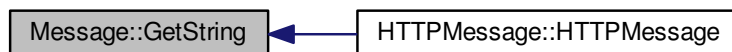
### 3.7.3 Member Function Documentation

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

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

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

Here is the caller graph for this function:



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

### 3.7.4 Field Documentation

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

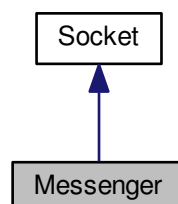
The documentation for this class was generated from the following file:

- `include/Message.h`

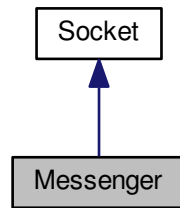
## 3.8 Messenger Class Reference

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

Inheritance diagram for Messenger:



Collaboration diagram for Messenger:



## Public Member Functions

- [Messenger](#) ()
- [Messenger](#) (int port)
- [~Messenger](#) ()
- bool [Connect](#) ()  
*Connect the master.*
- void [Disconnect](#) ()  
*Remove the master.*
- 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 [Broadcast](#) (const [Message](#) &m) const  
*Emit a message to all clients connected through the socket.*

## Additional Inherited Members

### 3.8.1 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

### 3.8.2 Constructor & Destructor Documentation

#### 3.8.2.1 Messenger::Messenger ( )

#### 3.8.2.2 Messenger::Messenger ( int port )

## 3.8.2.3 Messenger::~Messenger ( )

## 3.8.3 Member Function Documentation

3.8.3.1 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
----	----------	-------------------------------------

## 3.8.3.2 bool Messenger::Connect ( )

Connect the master.

Connect this master to the socket for clients to be able to bind.

## 3.8.3.3 void Messenger::Disconnect ( )

Remove the master.

Remove this master from the socket, thus disconnecting automatically the clients connected.

## 3.8.3.4 MessageKey Messenger::Receive ( )

Handle a message reception from a client.

## Returns

The key to the message received if successfully parsed

3.8.3.5 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

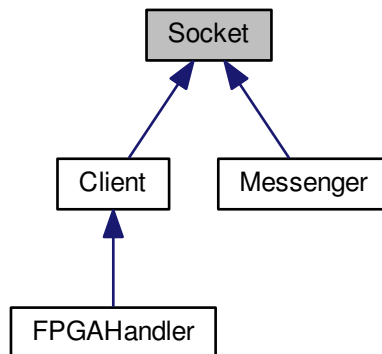
The documentation for this class was generated from the following file:

- include/Messenger.h

## 3.9 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
- SocketType [GetSocketType](#) (int sid) 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()

### 3.9.1 Detailed Description

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

#### Author

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

#### Date

23 Mar 2015

### 3.9.2 Constructor & Destructor Documentation

**3.9.2.1** `Socket::Socket ( )` [inline]

**3.9.2.2** `Socket::Socket ( int port )`

**3.9.2.3** `virtual Socket::~~Socket ( )` [virtual]

### 3.9.3 Member Function Documentation

**3.9.3.1** `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
----------------------	---------------------	--

**3.9.3.2** `void Socket::Bind ( )` [protected]

Bind a name to a socket.

#### Returns

Success of the operation

**3.9.3.3** `void Socket::DumpConnected ( )` const

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

Receive a message from a socket.

**Returns**

Received message as a `std::string`

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

Retrieve the port used for this socket.

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

**3.9.3.7** `SocketType Socket::GetSocketType ( int sid ) const` `[inline]`

Here is the caller graph for this function:



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

Here is the call graph for this function:



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

Listen to incoming messages.

Set the socket to listen to any message coming from outside

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

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

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

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

Send a message on a socket.

Here is the caller graph for this function:



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

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

3.9.3.15 `bool Socket::Start ( )` [protected]

Start the socket.

Launch all mandatory operations to set the socket to be used

Returns

Success of the operation

3.9.3.16 `void Socket::Stop ( )` [protected]

Terminates the socket and all attached communications.

## 3.9.4 Field Documentation

3.9.4.1 `char Socket::fBuffer[MAX_WORD_LENGTH]` [protected]

3.9.4.2 `fd_set Socket::fMaster` [protected]

Master file descriptor list.

3.9.4.3 `int Socket::fPort` [protected]

3.9.4.4 `fd_set Socket::fReadFds` [protected]

Temp file descriptor list for select()

3.9.4.5 `SocketCollection Socket::fSocketsConnected` [protected]

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

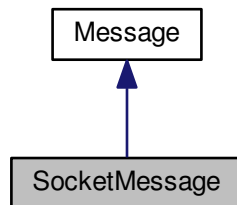
- include/Socket.h

### 3.10 SocketMessage Class Reference

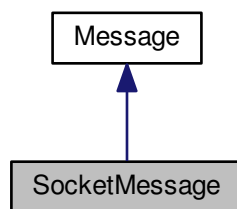
Socket-passed message type.

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

Inheritance diagram for SocketMessage:



Collaboration diagram for SocketMessage:



#### Public Member Functions

- [SocketMessage](#) ()
- [SocketMessage](#) (const [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

### 3.10.1 Detailed Description

Socket-passed message type.

#### Author

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

#### Date

26 Mar 2015

### 3.10.2 Constructor & Destructor Documentation

3.10.2.1 [SocketMessage::SocketMessage](#) ( ) [\[inline\]](#)

3.10.2.2 [SocketMessage::SocketMessage](#) ( const Message & msg ) [\[inline\]](#)

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

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

3.10.2.5 [SocketMessage::SocketMessage](#) ( MessageKey key ) [\[inline\]](#)

Here is the call graph for this function:



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

Here is the call graph for this function:



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

Here is the call graph for this function:



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

Here is the call graph for this function:



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

Here is the call graph for this function:



3.10.2.10 `SocketMessage::SocketMessage ( MessageKey key, const double value )` `[inline]`

Here is the call graph for this function:



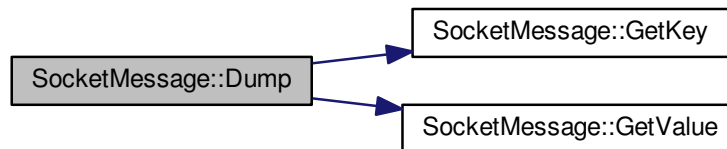
3.10.2.11 `SocketMessage::SocketMessage ( MessageMap msg_m )` `[inline]`

3.10.2.12 `SocketMessage::~~SocketMessage ( )` `[inline]`

### 3.10.3 Member Function Documentation

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

Here is the call graph for this function:



3.10.3.2 `int SocketMessage::GetIntValue ( ) const` `[inline]`

3.10.3.3 `MessageKey SocketMessage::GetKey ( ) const` `[inline]`

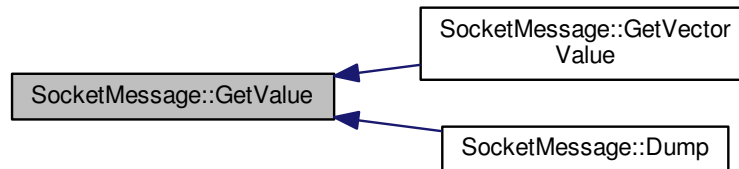
Here is the caller graph for this function:



3.10.3.4 `std::string SocketMessage::GetString ( ) const` `[inline]`

3.10.3.5 `std::string SocketMessage::GetValue ( ) const` `[inline]`

Here is the caller graph for this function:



3.10.3.6 `VectorValue SocketMessage::GetVectorValue ( ) const` `[inline]`

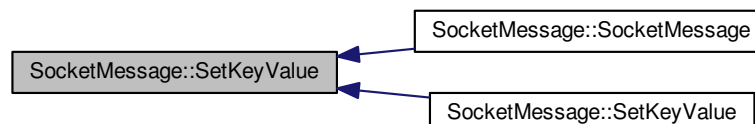
Here is the call graph for this function:



3.10.3.7 `void SocketMessage::SetKeyValue ( MessageKey key, std::string value )` `[inline]`

Send a string-valued message.

Here is the caller graph for this function:



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

Here is the call graph for this function:



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

Send an integer-valued message.

Here is the call graph for this function:



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

Send an float-valued message.

Here is the call graph for this function:



3.10.3.11 `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

### 3.11 TDCConfiguration Class Reference

```
#include <TDCConfiguration.h>
```

#### Public Types

- enum [EdgeResolution](#) {  
[E\\_100PS](#) =0, [E\\_200PS](#), [E\\_400PS](#), [E\\_800PS](#),  
[E\\_1600PS](#), [E\\_3120PS](#), [E\\_6250PS](#), [E\\_12500PS](#) }
- enum [DeadTime](#) { [DT\\_5NS](#) =0, [DT\\_10NS](#), [DT\\_30NS](#), [DT\\_100NS](#) }
- enum [WidthResolution](#) {  
[W\\_100PS](#) =0, [W\\_200PS](#), [W\\_400PS](#), [W\\_800PS](#),  
[W\\_1p6NS](#), [W\\_3p2NS](#), [W\\_6p25NS](#), [W\\_12p5NS](#),  
[W\\_25NS](#), [W\\_50NS](#), [W\\_100NS](#), [W\\_200NS](#),  
[W\\_400NS](#), [W\\_800NS](#) }

#### Public Member Functions

- [TDCConfiguration](#) ()
- virtual [~TDCConfiguration](#) ()
- void [SetEdgeResolution](#) (const [EdgeResolution](#) r)
- [EdgeResolution](#) [GetEdgeResolution](#) () const
- void [SetChannelOffset](#) (int channel, uint16\_t offset)
- uint16\_t [GetChannelOffset](#) (int channel)
- void [SetAllChannelsOffset](#) (short offset)
- void [SetWidthResolution](#) (const [WidthResolution](#) r)
- [WidthResolution](#) [GetWidthResolution](#) () const
- void [SetDeadTime](#) (const [DeadTime](#) dt)
- [DeadTime](#) [GetDeadTime](#) () const
- void [SetLeadingMode](#) (const bool lead=true)  
*Enable the detection of leading edges.*
- bool [GetLeadingMode](#) () const  
*Extract the status for the detection of leading edges.*
- void [SetTrailingMode](#) (const bool trail=true)  
*Enable/disable the detection of trailing edges.*
- bool [GetTrailingMode](#) () const  
*Extract the status for the detection of trailing edges.*

- void [SetTriggerMatchingMode](#) (const bool trig=true)
- bool [GetTriggerMatchingMode](#) () const
- void [SetEdgesPairing](#) (const bool pair=true)
- bool [GetEdgesPairing](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

### 3.11.1 Detailed Description

Object handling the configuration word provided by/to the HPTDC chip

Author

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

Date

16 Apr 2015

### 3.11.2 Member Enumeration Documentation

#### 3.11.2.1 enum TDCCConfiguration::DeadTime

Enumerator

***DT\_5NS***  
***DT\_10NS***  
***DT\_30NS***  
***DT\_100NS***

#### 3.11.2.2 enum TDCCConfiguration::EdgeResolution

Enumerator

***E\_100PS***  
***E\_200PS***  
***E\_400PS***  
***E\_800PS***  
***E\_1600PS***  
***E\_3120PS***  
***E\_6250PS***  
***E\_12500PS***

#### 3.11.2.3 enum TDCCConfiguration::WidthResolution

Enumerator

***W\_100PS***  
***W\_200PS***  
***W\_400PS***  
***W\_800PS***  
***W\_1p6NS***

***W\_3p2NS***  
***W\_6p25NS***  
***W\_12p5NS***  
***W\_25NS***  
***W\_50NS***  
***W\_100NS***  
***W\_200NS***  
***W\_400NS***  
***W\_800NS***

### 3.11.3 Constructor & Destructor Documentation

3.11.3.1 `TDCConfiguration::TDCConfiguration ( )`

3.11.3.2 `virtual TDCConfiguration::~~TDCConfiguration ( )` `[inline]`, `[virtual]`

### 3.11.4 Member Function Documentation

3.11.4.1 `void TDCConfiguration::Dump ( std::ostream & os = std::cout ) const`

3.11.4.2 `uint16_t TDCConfiguration::GetChannelOffset ( int channel )`

3.11.4.3 `DeadTime TDCConfiguration::GetDeadTime ( ) const` `[inline]`

3.11.4.4 `EdgeResolution TDCConfiguration::GetEdgeResolution ( ) const` `[inline]`

3.11.4.5 `bool TDCConfiguration::GetEdgesPairing ( ) const` `[inline]`

3.11.4.6 `bool TDCConfiguration::GetLeadingMode ( ) const` `[inline]`

Extract the status for the detection of leading edges.

3.11.4.7 `bool TDCConfiguration::GetTrailingMode ( ) const` `[inline]`

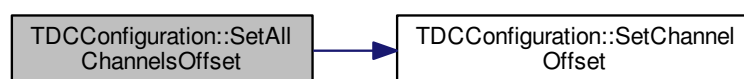
Extract the status for the detection of trailing edges.

3.11.4.8 `bool TDCConfiguration::GetTriggerMatchingMode ( ) const` `[inline]`

3.11.4.9 `WidthResolution TDCConfiguration::GetWidthResolution ( ) const` `[inline]`

3.11.4.10 `void TDCConfiguration::SetAllChannelsOffset ( short offset )` `[inline]`

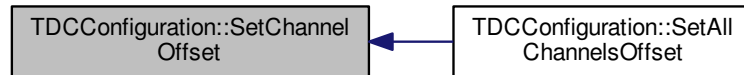
Here is the call graph for this function:





3.11.4.11 void TDCCConfiguration::SetChannelOffset ( int *channel*, uint16\_t *offset* )

Here is the caller graph for this function:



3.11.4.12 void TDCCConfiguration::SetDeadTime ( const DeadTime *dt* ) [inline]

3.11.4.13 void TDCCConfiguration::SetEdgeResolution ( const EdgeResolution *r* ) [inline]

3.11.4.14 void TDCCConfiguration::SetEdgesPairing ( const bool *pair* = true ) [inline]

3.11.4.15 void TDCCConfiguration::SetLeadingMode ( const bool *lead* = true ) [inline]

Enable the detection of leading edges.

3.11.4.16 void TDCCConfiguration::SetTrailingMode ( const bool *trail* = true ) [inline]

Enable/disable the detection of trailing edges.

3.11.4.17 void TDCCConfiguration::SetTriggerMatchingMode ( const bool *trig* = true ) [inline]

3.11.4.18 void TDCCConfiguration::SetWidthResolution ( const WidthResolution *r* ) [inline]

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

- include/TDCCConfiguration.h



# Index

- ~Client
  - Client, [6](#)
- ~Exception
  - Exception, [8](#)
- ~FPGAHandler
  - FPGAHandler, [12](#)
- ~Message
  - Message, [16](#)
- ~Messenger
  - Messenger, [18](#)
- ~Socket
  - Socket, [21](#)
- ~SocketMessage
  - SocketMessage, [27](#)
- ~TDCConfiguration
  - TDCConfiguration, [32](#)
- AcceptConnections
  - Socket, [21](#)
- Bind
  - Socket, [21](#)
- Broadcast
  - Messenger, [19](#)
- Client, [5](#)
  - ~Client, [6](#)
  - Client, [6](#)
  - Connect, [6](#)
  - Disconnect, [6](#)
  - GetType, [6](#)
  - ParseMessage, [6](#)
  - Receive, [6](#)
  - Send, [6](#)
- Connect
  - Client, [6](#)
  - Messenger, [19](#)
- DT\_100NS
  - TDCConfiguration, [31](#)
- DT\_10NS
  - TDCConfiguration, [31](#)
- DT\_30NS
  - TDCConfiguration, [31](#)
- DT\_5NS
  - TDCConfiguration, [31](#)
- DeadTime
  - TDCConfiguration, [31](#)
- Decode
  - HTTPMessage, [14](#)
- Description
  - Exception, [8](#)
- Disconnect
  - Client, [6](#)
  - Messenger, [19](#)
- Dump
  - Exception, [8](#)
  - HTTPMessage, [14](#)
  - Message, [17](#)
  - SocketMessage, [27](#)
  - TDCConfiguration, [32](#)
- DumpConnected
  - Socket, [21](#)
- E\_100PS
  - TDCConfiguration, [31](#)
- E\_12500PS
  - TDCConfiguration, [31](#)
- E\_1600PS
  - TDCConfiguration, [31](#)
- E\_200PS
  - TDCConfiguration, [31](#)
- E\_3120PS
  - TDCConfiguration, [31](#)
- E\_400PS
  - TDCConfiguration, [31](#)
- E\_6250PS
  - TDCConfiguration, [31](#)
- E\_800PS
  - TDCConfiguration, [31](#)
- EdgeResolution
  - TDCConfiguration, [31](#)
- Encode
  - HTTPMessage, [15](#)
- ErrorNumber
  - Exception, [8](#)
- Exception, [7](#)
  - ~Exception, [8](#)
  - Description, [8](#)
  - Dump, [8](#)
  - ErrorNumber, [8](#)
  - Exception, [7](#)
  - From, [9](#)
  - Type, [9](#)
  - TypeString, [9](#)
- fBuffer
  - Socket, [23](#)
- fMaster
  - Socket, [23](#)

- FPGAHandler, 11
  - ~FPGAHandler, 12
  - FPGAHandler, 12
  - GetFilename, 12
  - GetType, 12
  - OpenFile, 12
  - ReadBuffer, 12
  - ReadConfiguration, 12
  - SendConfiguration, 12
- fPort
  - Socket, 23
- fReadFds
  - Socket, 23
- fSocketsConnected
  - Socket, 23
- fString
  - Message, 17
- FetchMessage
  - Socket, 21
- file\_header\_t, 10
  - magic, 10
  - run\_id, 10
  - spill\_id, 10
- From
  - Exception, 9
- GetChannelOffset
  - TDCConfiguration, 32
- GetDeadTime
  - TDCConfiguration, 32
- GetEdgeResolution
  - TDCConfiguration, 32
- GetEdgesPairing
  - TDCConfiguration, 32
- GetFilename
  - FPGAHandler, 12
- GetIntValue
  - SocketMessage, 27
- GetKey
  - HTTPMessage, 15
  - Message, 17
  - SocketMessage, 27
- GetLeadingMode
  - TDCConfiguration, 32
- GetPort
  - Socket, 22
- GetSocketId
  - Socket, 22
- GetSocketType
  - Socket, 22
- GetString
  - Message, 17
  - SocketMessage, 27
- GetTrailingMode
  - TDCConfiguration, 32
- GetTriggerMatchingMode
  - TDCConfiguration, 32
- GetType
  - Client, 6
- FPGAHandler, 12
  - GetValue
    - SocketMessage, 28
  - GetVectorValue
    - SocketMessage, 28
  - GetWidthResolution
    - TDCConfiguration, 32
- HTTPMessage, 12
  - Decode, 14
  - Dump, 14
  - Encode, 15
  - GetKey, 15
  - HTTPMessage, 13, 14
- IsFromWeb
  - Message, 17
- IsWebSocket
  - Socket, 22
- Listen
  - Socket, 22
- ListenerInfo, 15
  - name, 15
  - type, 15
- magic
  - file\_header\_t, 10
- Message, 15
  - ~Message, 16
  - Dump, 17
  - fString, 17
  - GetKey, 17
  - GetString, 17
  - IsFromWeb, 17
  - Message, 16
- Messenger, 17
  - ~Messenger, 18
  - Broadcast, 19
  - Connect, 19
  - Disconnect, 19
  - Messenger, 18
  - Receive, 19
  - Send, 19
- name
  - ListenerInfo, 15
- OpenFile
  - FPGAHandler, 12
- ParseMessage
  - Client, 6
- PrepareConnection
  - Socket, 22
- ReadBuffer
  - FPGAHandler, 12
- ReadConfiguration
  - FPGAHandler, 12

- Receive
  - Client, [6](#)
  - Messenger, [19](#)
- run\_id
  - file\_header\_t, [10](#)
- SelectConnections
  - Socket, [22](#)
- Send
  - Client, [6](#)
  - Messenger, [19](#)
- SendConfiguration
  - FPGAHandler, [12](#)
- SendMessage
  - Socket, [22](#)
- SetAllChannelsOffset
  - TDCConfiguration, [32](#)
- SetChannelOffset
  - TDCConfiguration, [32](#)
- SetDeadTime
  - TDCConfiguration, [33](#)
- SetEdgeResolution
  - TDCConfiguration, [33](#)
- SetEdgesPairing
  - TDCConfiguration, [33](#)
- SetKeyValue
  - SocketMessage, [28](#), [29](#)
- SetLeadingMode
  - TDCConfiguration, [33](#)
- SetPort
  - Socket, [23](#)
- SetSocketId
  - Socket, [23](#)
- SetTrailingMode
  - TDCConfiguration, [33](#)
- SetTriggerMatchingMode
  - TDCConfiguration, [33](#)
- SetWidthResolution
  - TDCConfiguration, [33](#)
- Socket, [19](#)
  - ~Socket, [21](#)
  - AcceptConnections, [21](#)
  - Bind, [21](#)
  - DumpConnected, [21](#)
  - fBuffer, [23](#)
  - fMaster, [23](#)
  - fPort, [23](#)
  - fReadFds, [23](#)
  - fSocketsConnected, [23](#)
  - FetchMessage, [21](#)
  - GetPort, [22](#)
  - GetSocketId, [22](#)
  - GetSocketType, [22](#)
  - IsWebSocket, [22](#)
  - Listen, [22](#)
  - PrepareConnection, [22](#)
  - SelectConnections, [22](#)
  - SendMessage, [22](#)
  - SetPort, [23](#)
  - SetSocketId, [23](#)
  - Socket, [21](#)
  - Start, [23](#)
  - Stop, [23](#)
- SocketMessage, [24](#)
  - ~SocketMessage, [27](#)
  - Dump, [27](#)
  - GetIntValue, [27](#)
  - GetKey, [27](#)
  - GetString, [27](#)
  - GetValue, [28](#)
  - GetVectorValue, [28](#)
  - SetKeyValue, [28](#), [29](#)
  - SocketMessage, [25–27](#)
- spill\_id
  - file\_header\_t, [10](#)
- Start
  - Socket, [23](#)
- Stop
  - Socket, [23](#)
- TDCConfiguration, [30](#)
  - ~TDCConfiguration, [32](#)
  - DT\_100NS, [31](#)
  - DT\_10NS, [31](#)
  - DT\_30NS, [31](#)
  - DT\_5NS, [31](#)
  - DeadTime, [31](#)
  - Dump, [32](#)
  - E\_100PS, [31](#)
  - E\_12500PS, [31](#)
  - E\_1600PS, [31](#)
  - E\_200PS, [31](#)
  - E\_3120PS, [31](#)
  - E\_400PS, [31](#)
  - E\_6250PS, [31](#)
  - E\_800PS, [31](#)
  - EdgeResolution, [31](#)
  - GetChannelOffset, [32](#)
  - GetDeadTime, [32](#)
  - GetEdgeResolution, [32](#)
  - GetEdgesPairing, [32](#)
  - GetLeadingMode, [32](#)
  - GetTrailingMode, [32](#)
  - GetTriggerMatchingMode, [32](#)
  - GetWidthResolution, [32](#)
  - SetAllChannelsOffset, [32](#)
  - SetChannelOffset, [32](#)
  - SetDeadTime, [33](#)
  - SetEdgeResolution, [33](#)
  - SetEdgesPairing, [33](#)
  - SetLeadingMode, [33](#)
  - SetTrailingMode, [33](#)
  - SetTriggerMatchingMode, [33](#)
  - SetWidthResolution, [33](#)
  - TDCConfiguration, [32](#)
  - W\_100NS, [32](#)
  - W\_100PS, [31](#)
  - W\_12p5NS, [32](#)

- W\_1p6NS, [31](#)
- W\_200NS, [32](#)
- W\_200PS, [31](#)
- W\_25NS, [32](#)
- W\_3p2NS, [31](#)
- W\_400NS, [32](#)
- W\_400PS, [31](#)
- W\_50NS, [32](#)
- W\_6p25NS, [32](#)
- W\_800NS, [32](#)
- W\_800PS, [31](#)
- WidthResolution, [31](#)
- Type
  - Exception, [9](#)
- type
  - ListenerInfo, [15](#)
- TypeString
  - Exception, [9](#)
- W\_100NS
  - TDCConfiguration, [32](#)
- W\_100PS
  - TDCConfiguration, [31](#)
- W\_12p5NS
  - TDCConfiguration, [32](#)
- W\_1p6NS
  - TDCConfiguration, [31](#)
- W\_200NS
  - TDCConfiguration, [32](#)
- W\_200PS
  - TDCConfiguration, [31](#)
- W\_25NS
  - TDCConfiguration, [32](#)
- W\_3p2NS
  - TDCConfiguration, [31](#)
- W\_400NS
  - TDCConfiguration, [32](#)
- W\_400PS
  - TDCConfiguration, [31](#)
- W\_50NS
  - TDCConfiguration, [32](#)
- W\_6p25NS
  - TDCConfiguration, [32](#)
- W\_800NS
  - TDCConfiguration, [32](#)
- W\_800PS
  - TDCConfiguration, [31](#)
- WidthResolution
  - TDCConfiguration, [31](#)