



Delta3D Version 2.4.0

dtNetGM::

Reference Manual

Contents

1	Main Page	1
2	Directory Documentation	3
2.1	inc/dtNetGM/ Directory Reference	3
2.2	src/dtNetGM/ Directory Reference	4
2.3	inc/ Directory Reference	5
2.4	src/ Directory Reference	6
3	Namespace Documentation	7
3.1	dtCore Namespace Reference	7
3.2	dtGame Namespace Reference	8
3.3	dtNetGM Namespace Reference	9
3.3.1	Detailed Description	9
3.3.2	Function Documentation	9
3.3.2.1	IMPLEMENT_ENUM	9
3.3.2.2	IMPLEMENT_ENUM	9
3.3.2.3	IMPLEMENT_MANAGEMENT_LAYER	9
4	Class Documentation	11
4.1	ClientConnectionListener Class Reference	11
4.1.1	Detailed Description	11
4.1.2	Member Typedef Documentation	12
4.1.2.1	sptr	12
4.1.2.2	wptr	12
4.1.3	Constructor & Destructor Documentation	12
4.1.3.1	~ClientConnectionListener	12
4.1.3.2	ClientConnectionListener	12
4.1.4	Member Function Documentation	12
4.1.4.1	Create	12
4.1.4.2	onConnect	12
4.1.4.3	onConnectFailure	12
4.1.4.4	onDisconnect	12
4.1.4.5	onError	12
4.1.4.6	onExit	12
4.1.4.7	onFailure	12
4.1.4.8	onNewConn	12
4.1.4.9	onReceive	12
4.2	ClientNetworkComponent Class Reference	13

4.2.1	Constructor & Destructor Documentation	14
4.2.1.1	ClientNetworkComponent	14
4.2.1.2	~ClientNetworkComponent	14
4.2.2	Member Function Documentation	14
4.2.2.1	GetMachineInfo	14
4.2.2.2	GetServer	14
4.2.2.3	IsConnectedClient	14
4.2.2.4	OnDisconnect	14
4.2.2.5	ProcessInfoClientConnected	14
4.2.2.6	ProcessNetClientNotifyDisconnect	14
4.2.2.7	ProcessNetServerAcceptConnection	15
4.2.2.8	ProcessNetServerRejectConnection	15
4.2.2.9	ProcessNetServerRejectMessage	15
4.2.2.10	SendRequestConnectionMessage	15
4.2.2.11	SetupClient	15
4.2.3	Member Data Documentation	15
4.2.3.1	DEFAULT_NAME	15
4.2.3.2	mConnectedClients	15
4.3	DataStreamPacket Class Reference	16
4.3.1	Detailed Description	16
4.3.2	Member Typedef Documentation	17
4.3.2.1	sptr	17
4.3.2.2	wptr	17
4.3.3	Constructor & Destructor Documentation	17
4.3.3.1	DataStreamPacket	17
4.3.3.2	DataStreamPacket	17
4.3.3.3	DataStreamPacket	17
4.3.3.4	~DataStreamPacket	17
4.3.4	Member Function Documentation	17
4.3.4.1	GetDataStreamId	17
4.3.4.2	GetDataStreamSize	17
4.3.4.3	GetIndex	17
4.3.4.4	GetPacketCount	17
4.3.4.5	GetPayloadBuffer	17
4.3.4.6	GetPayloadSize	17
4.3.4.7	getSize	17
4.3.4.8	readPacket	17
4.3.4.9	SetDataStreamId	17
4.3.4.10	SetDataStreamSize	17
4.3.4.11	SetIndex	17
4.3.4.12	SetPacketCount	17

4.3.4.13	SetPayloadSize	17
4.3.4.14	writePacket	17
4.3.5	Member Data Documentation	18
4.3.5.1	ID	18
4.3.5.2	MAX_PAYLOAD	18
4.4	DestinationType Class Reference	19
4.4.1	Detailed Description	19
4.4.2	Member Data Documentation	19
4.4.2.1	ALL_CLIENTS	19
4.4.2.2	ALL_NOT_CLIENTS	19
4.4.2.3	DESTINATION	19
4.5	MachineInfoMessage Class Reference	20
4.5.1	Detailed Description	21
4.5.2	Constructor & Destructor Documentation	21
4.5.2.1	MachineInfoMessage	21
4.5.2.2	~MachineInfoMessage	21
4.5.3	Member Function Documentation	21
4.5.3.1	GetHostName	21
4.5.3.2	GetIPAddress	21
4.5.3.3	GetMachineInfo	21
4.5.3.4	GetMachineInfoName	21
4.5.3.5	GetPing	21
4.5.3.6	GetTimeStamp	21
4.5.3.7	GetUniqueld	21
4.5.3.8	SetHostName	21
4.5.3.9	SetIPAddress	21
4.5.3.10	SetMachineInfo	21
4.5.3.11	SetMachineInfoName	21
4.5.3.12	SetPing	22
4.5.3.13	SetTimeStamp	22
4.5.3.14	SetUniqueld	22
4.6	MessageActionCode Class Reference	23
4.6.1	Member Data Documentation	23
4.6.1.1	DROP	23
4.6.1.2	REJECT	23
4.6.1.3	SEND	23
4.6.1.4	WAIT	23
4.7	MessagePacket Class Reference	24
4.7.1	Detailed Description	25
4.7.2	Member Typedef Documentation	25
4.7.2.1	sptr	25

4.7.2.2	wptr	25
4.7.3	Constructor & Destructor Documentation	25
4.7.3.1	MessagePacket	25
4.7.3.2	MessagePacket	25
4.7.3.3	MessagePacket	25
4.7.3.4	~MessagePacket	25
4.7.4	Member Function Documentation	25
4.7.4.1	BuildFromMessage	25
4.7.4.2	FillMessage	25
4.7.4.3	GetDestinatonId	25
4.7.4.4	GetMessageld	25
4.7.4.5	GetMessageParameters	26
4.7.4.6	getSize	26
4.7.4.7	GetSourceId	26
4.7.4.8	OverrideDestination	26
4.7.4.9	readPacket	26
4.7.4.10	writePacket	26
4.7.5	Member Data Documentation	26
4.7.5.1	ID	26
4.8	NetworkBridge Class Reference	27
4.8.1	Detailed Description	28
4.8.2	Constructor & Destructor Documentation	28
4.8.2.1	NetworkBridge	28
4.8.2.2	~NetworkBridge	28
4.8.3	Member Function Documentation	28
4.8.3.1	Disconnect	28
4.8.3.2	GetHostDescription	28
4.8.3.3	GetMachineInfo	28
4.8.3.4	IsConnectedClient	28
4.8.3.5	IsNetworkConnected	28
4.8.3.6	OnConnect	28
4.8.3.7	OnConnectFailure	28
4.8.3.8	OnDisconnect	28
4.8.3.9	OnError	28
4.8.3.10	OnExit	29
4.8.3.11	OnFailure	29
4.8.3.12	OnNewConnection	29
4.8.3.13	OnReceive	29
4.8.3.14	OnTimeout	29
4.8.3.15	SendDataStream	29
4.8.3.16	SetClientConnected	29

4.8.3.17	SetMachineInfo	29
4.9	NetworkComponent Class Reference	30
4.9.1	Detailed Description	32
4.9.2	Constructor & Destructor Documentation	32
4.9.2.1	NetworkComponent	32
4.9.2.2	~NetworkComponent	32
4.9.3	Member Function Documentation	32
4.9.3.1	AddConnection	32
4.9.3.2	CreateDataStream	33
4.9.3.3	CreateMessage	33
4.9.3.4	Disconnect	33
4.9.3.5	DispatchNetworkMessage	33
4.9.3.6	GetConnectedClients	33
4.9.3.7	GetConnection	33
4.9.3.8	GetHostName	33
4.9.3.9	GetMachineInfo	33
4.9.3.10	IsGnetInitialized	33
4.9.3.11	IsReliable	33
4.9.3.12	IsServer	33
4.9.3.13	IsShuttingDown	33
4.9.3.14	OnAddedToGM	33
4.9.3.15	OnBeforeSendMessage	34
4.9.3.16	OnConnect	34
4.9.3.17	OnConnectFailure	34
4.9.3.18	OnDisconnect	34
4.9.3.19	OnError	34
4.9.3.20	OnExit	34
4.9.3.21	OnFailure	34
4.9.3.22	OnNewConnection	34
4.9.3.23	OnReceivedDataStream	35
4.9.3.24	OnReceivedNetworkMessage	35
4.9.3.25	OnRemovedFromGM	35
4.9.3.26	OnTimeOut	35
4.9.3.27	ProcessInfoClientConnected	35
4.9.3.28	ProcessMessage	35
4.9.3.29	ProcessNetClientNotifyDisconnect	35
4.9.3.30	ProcessNetClientRequestConnection	35
4.9.3.31	ProcessNetServerAcceptConnection	35
4.9.3.32	ProcessNetServerRejectConnection	36
4.9.3.33	ProcessNetServerRejectMessage	36
4.9.3.34	ProcessTickLocal	36

4.9.3.35	RemoveConnection	36
4.9.3.36	SendNetworkMessage	36
4.9.3.37	SetConnectionParameters	36
4.9.3.38	ShutdownNetwork	36
4.9.4	Member Data Documentation	36
4.9.4.1	mBufferMutex	36
4.9.4.2	mConnections	36
4.9.4.3	mMutex	36
4.9.4.4	mRateIn	36
4.9.4.5	mRateOut	36
4.9.4.6	mReliable	36
4.10	ServerConnectionListener Class Reference	37
4.10.1	Detailed Description	37
4.10.2	Member Typedef Documentation	37
4.10.2.1	sptr	37
4.10.2.2	wptr	37
4.10.3	Constructor & Destructor Documentation	37
4.10.3.1	ServerConnectionListener	37
4.10.3.2	~ServerConnectionListener	38
4.10.4	Member Function Documentation	38
4.10.4.1	Create	38
4.10.4.2	getNewConnectionParams	38
4.10.4.3	onListenFailure	38
4.10.4.4	onListenSuccess	38
4.10.5	Member Data Documentation	38
4.10.5.1	mInRate	38
4.10.5.2	mMutex	38
4.10.5.3	mOutRate	38
4.10.5.4	mReliable	38
4.10.5.5	mServerNetworkcomponent	38
4.11	ServerNetworkComponent Class Reference	39
4.11.1	Constructor & Destructor Documentation	40
4.11.1.1	ServerNetworkComponent	40
4.11.1.2	~ServerNetworkComponent	40
4.11.2	Member Function Documentation	40
4.11.2.1	AcceptClient	40
4.11.2.2	OnDisconnect	40
4.11.2.3	OnListenFailure	40
4.11.2.4	OnListenSuccess	40
4.11.2.5	ProcessNetClientNotifyDisconnect	40
4.11.2.6	ProcessNetClientRequestConnection	40

4.11.2.7	SendConnectedClientMessage	40
4.11.2.8	SendInfoClientConnectedMessage	41
4.11.2.9	SetupServer	41
4.11.3	Member Data Documentation	41
4.11.3.1	DEFAULT_NAME	41
4.11.3.2	mAcceptClients	41
5	File Documentation	43
5.1	clientconnectionlistener.cpp File Reference	43
5.2	clientconnectionlistener.h File Reference	44
5.3	clientnetworkcomponent.cpp File Reference	45
5.4	clientnetworkcomponent.h File Reference	46
5.5	datastreampacket.cpp File Reference	47
5.6	datastreampacket.h File Reference	48
5.7	dtnetgm.h File Reference	49
5.8	export.h File Reference	50
5.8.1	Define Documentation	50
5.8.1.1	DT_NETGM_EXPORT	50
5.9	machineinfomessage.cpp File Reference	51
5.10	machineinfomessage.h File Reference	52
5.11	mainpage.h File Reference	53
5.11.1	Detailed Description	53
5.12	messagepacket.cpp File Reference	54
5.13	messagepacket.h File Reference	55
5.14	networkbridge.cpp File Reference	56
5.15	networkbridge.h File Reference	57
5.16	networkcomponent.cpp File Reference	58
5.17	networkcomponent.h File Reference	59
5.18	serverconnectionlistener.cpp File Reference	60
5.19	serverconnectionlistener.h File Reference	61
5.20	servernetworkcomponent.cpp File Reference	62
5.21	servernetworkcomponent.h File Reference	63

Main Page

Delta3D is an Open Source engine which can be used for games, simulations, or other graphical applications.

The Delta3D framework exists as a number of modules, each sitting in its own library, enclosed within its own namespace. At the very core lies the **dtCore** (p. 7) library. This contains basic, low-level functionality which is mostly required for all 3D applications written in C++.

Around and alongside this sit other supporting libraries, such as dtUtil (containing reusable features which are useful for most applications), dtTerrain (for rendering terrain databases), **dtGame** (p. 8), dtNet, etc.

Extensive online documentation is available from the Delta3D Docs section to help in using Delta3D.

The project's original reference guides generated by Doxygen from the source code may be viewed at the Delta3D API Documentation section.

To download source code, binaries, dependencies and sample datasets visit the Delta3D Downloads page.

For more about dependencies see the Delta3D Dependencies page.

The documentation you are looking at can be downloaded from www.3draum.ch.

Enjoy!

Directory Documentation

2.1 inc/dtNetGM/ Directory Reference

Files

- file `clientconnectionlistener.h`
- file `clientnetworkcomponent.h`
- file `datastreampacket.h`
- file `dtnetgm.h`
- file `export.h`
- file `machineinfomessage.h`
- file `mainpage.h`
- file `messagepacket.h`
- file `networkbridge.h`
- file `networkcomponent.h`
- file `serverconnectionlistener.h`
- file `servernetworkcomponent.h`

2.2 src/dtNetGM/ Directory Reference

Files

- file `clientconnectionlistener.cpp`
- file `clientnetworkcomponent.cpp`
- file `datastreampacket.cpp`
- file `machineinfomessage.cpp`
- file `messagepacket.cpp`
- file `networkbridge.cpp`
- file `networkcomponent.cpp`
- file `serverconnectionlistener.cpp`
- file `servernetworkcomponent.cpp`

2.3 inc/ Directory Reference

Directories

- directory dtNetGM

2.4 src/ Directory Reference

Directories

- directory **dtNetGM**

Namespace Documentation

3.1 dtCore Namespace Reference

3.2 dtGame Namespace Reference

3.3 dtNetGM Namespace Reference

The **dtNetGM** (p. 9) namespace contains networking classes.

Classes

- class **ClientConnectionListener**
Provides the interface to a GNE::Connection.
- class **ClientNetworkComponent**
- class **DataStreamPacket**
*A **DataStreamPacket** (p. 16) contains a dtGame::Message blocks to be sent over a Network of other connected dtGame::GameManager's.*
- class **MachineInfoMessage**
*A **MachineInfoMessage** (p. 20) contains a dtGame::MachineInfo to be used with different messages to send information about a GameManager accross the network.*
- class **MessageActionCode**
- class **MessagePacket**
*A **MessagePacket** (p. 24) contains a dtGame::Message to be sent over a Network of other connected dtGame::GameManager's.*
- class **NetworkBridge**
contains GNE components for communication across a network This class represents a single host on the other side of the network
- class **NetworkComponent**
baseclass GMComponent to communicate as client - server
- class **ServerConnectionListener**
This class is used as an interface to the GNE::Server connection.
- class **ServerNetworkComponent**

Functions

- **IMPLEMENT_ENUM (NetworkComponent::DestinationType)**
- **IMPLEMENT_ENUM (MessageActionCode)**
- **IMPLEMENT_MANAGEMENT_LAYER (NetworkComponent)**

3.3.1 Detailed Description

The **dtNetGM** (p. 9) namespace contains networking classes. **dtNetGM** (p.9) uses the Game Network Engine and HawkNL for backbone functionality.

3.3.2 Function Documentation

3.3.2.1 **dtNetGM::IMPLEMENT_ENUM (NetworkComponent::DestinationType)**

3.3.2.2 **dtNetGM::IMPLEMENT_ENUM (MessageActionCode)**

3.3.2.3 **dtNetGM::IMPLEMENT_MANAGEMENT_LAYER (NetworkComponent)**

Class Documentation

4.1 ClientConnectionListener Class Reference

Provides the interface to a GNE::Connection.

```
#include <inc/dtNetGM/clientconnectionlistener.h>
```

Public Types

- typedef GNE::SmartPtr< **ClientConnectionListener** > **sptr**
- typedef GNE::WeakPtr< **ClientConnectionListener** > **wptr**

Public Member Functions

- virtual ~**ClientConnectionListener** (void)
- virtual void **onConnect** (GNE::SyncConnection &conn)
- virtual void **onConnectFailure** (GNE::Connection &conn, const GNE::Error &error)
- virtual void **onDisconnect** (GNE::Connection &conn)
- virtual void **onError** (GNE::Connection &conn, const GNE::Error &error)
- virtual void **onExit** (GNE::Connection &conn)
- virtual void **onFailure** (GNE::Connection &conn, const GNE::Error &error)
- virtual void **onNewConn** (GNE::SyncConnection &conn)
- virtual void **onReceive** (GNE::Connection &conn)

Static Public Member Functions

- static **sptr Create** (**NetworkBridge** *netwBridge)
*static method used to create a new **ClientConnectionListener** (p. 11)*

Protected Member Functions

- **ClientConnectionListener** (**NetworkBridge** *netwBridge)

4.1.1 Detailed Description

Provides the interface to a GNE::Connection. This class is used internally by the NetMgr and is typically not used directly by the end user. This class contains a reference to an instance of NetMgr and calls it's virtual methods, mimicking the GNE interface.

4.1.2 Member Typedef Documentation

4.1.2.1 `typedef GNE::SmartPtr<ClientConnectionListener> sptr`

4.1.2.2 `typedef GNE::WeakPtr<ClientConnectionListener> wptr`

4.1.3 Constructor & Destructor Documentation

4.1.3.1 `~ClientConnectionListener (void) [virtual]`

4.1.3.2 `ClientConnectionListener (NetworkBridge * netwBridge) [protected]`

4.1.4 Member Function Documentation

4.1.4.1 `static sptr Create (NetworkBridge * netwBridge) [inline, static]`

static method used to create a new `ClientConnectionListener` (p. 11)

4.1.4.2 `void onConnect (GNE::SyncConnection & conn) [virtual]`

4.1.4.3 `void onConnectFailure (GNE::Connection & conn, const GNE::Error & error) [virtual]`

4.1.4.4 `void onDisconnect (GNE::Connection & conn) [virtual]`

4.1.4.5 `void onError (GNE::Connection & conn, const GNE::Error & error) [virtual]`

4.1.4.6 `void onExit (GNE::Connection & conn) [virtual]`

4.1.4.7 `void onFailure (GNE::Connection & conn, const GNE::Error & error) [virtual]`

4.1.4.8 `void onNewConn (GNE::SyncConnection & conn) [virtual]`

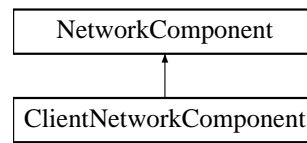
4.1.4.9 `void onReceive (GNE::Connection & conn) [virtual]`

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

- `clientconnectionlistener.h`
- `clientconnectionlistener.cpp`

4.2 ClientNetworkComponent Class Reference

#include <inc/dtNetGM/clientnetworkcomponent.h> Inheritance diagram for ClientNetworkComponent::



Public Member Functions

- **ClientNetworkComponent** (const std::string &gameName, const int gameVersion, const std::string &logFile="")
*Construct a **ClientNetworkComponent** (p. 13) with a game name and version to be used by GNE Calls base class constructor to initialize GNE.*
- const dtGame::MachineInfo * **GetServer** ()
Returns the MachineInfo of the Server, or NULL if we don't have a accepted connection.
- bool **IsConnectedClient** ()
Reveals if a server has accepted our connectionrequest.
- virtual void **OnDisconnect** (NetworkBridge &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a connection disconnects.*
- virtual void **ProcessInfoClientConnected** (const MachineInfoMessage &msg)
Processes a MessageType::INFO_CLIENT_CONNECTED Message.
- virtual void **ProcessNetClientNotifyDisconnect** (const MachineInfoMessage &msg)
Processes a MessageType::NETCLIENT_NOTIFY_DISCONNECT Message and removes the client from the connected clients vector.
- virtual void **ProcessNetServerAcceptConnection** (const MachineInfoMessage &msg)
Processes a MessageType::NETSERVER_ACCEPT_CONNECTION Message.
- virtual void **ProcessNetServerRejectConnection** (const dtGame::NetServerRejectMessage &msg)
Processes a MessageType::NETSERVER_REJECT_CONNECTION Message.
- virtual void **ProcessNetServerRejectMessage** (const dtGame::ServerMessageRejected &msg)
Processes a MessageType::SERVER_REQUEST_REJECTED Message.
- void **SendRequestConnectionMessage** ()
Utility method to put together a request connection message and send it.
- bool **SetupClient** (const std::string &host, const int portNum)
Setup a new network connection to a server.

Static Public Attributes

- static const std::string **DEFAULT_NAME** = "ClientNetworkComponent"

Protected Member Functions

- virtual ~**ClientNetworkComponent** (void)
- virtual const dtGame::MachineInfo * **GetMachineInfo** (const dtCore::Uniqueld &uniqueld)
Retrieves a dtGame::MachineInfo from the stored connections including all other connected clients If no matching connection is found, NULL is returned.*

Protected Attributes

- `std::vector< dtCore::RefPtr< dtGame::MachineInfo > >` **mConnectedClients**

4.2.1 Constructor & Destructor Documentation

4.2.1.1 ClientNetworkComponent (const std::string & *gameName*, const int *gameVersion*, const std::string & *logFile* = "")

Construct a **ClientNetworkComponent** (p. 13) with a game name and version to be used by GNE Calls base class constructor to initialize GNE. Parameters

gameName The game name

gameVersion The game version

logFile The logfile name

4.2.1.2 ~ClientNetworkComponent (void) [protected, virtual]

4.2.2 Member Function Documentation

4.2.2.1 const dtGame::MachineInfo * GetMachineInfo (const dtCore::Uniqueld & *uniqueld*) [protected, virtual]

Retrieves a `dtGame::MachineInfo*` from the stored connections including all other connected clients If no matching connection is found, NULL is returned. Parameters

dtCore::Uniqueld of the MachineInfo

Returns Pointer to the `dtGame::MachineInfo*` or NULL

Reimplemented from **NetworkComponent** (p. 33).

4.2.2.2 const dtGame::MachineInfo * GetServer ()

Returns the MachineInfo of the Server, or NULL if we don't have a accepted connection. Returns MachineInfo

4.2.2.3 bool IsConnectedClient () [inline]

Reveals if a server has accepted our connectionrequest. Returns boolean indicating if the server has accepted our connection request

4.2.2.4 void OnDisconnect (NetworkBridge & *networkBridge*) [virtual]

Function called by a **NetworkBridge** (p. 27) if a connection disconnects. Parameters

The NetworkBridge (p. 27)

Reimplemented from **NetworkComponent** (p. 34).

4.2.2.5 void ProcessInfoClientConnected (const MachineInfoMessage & *msg*) [virtual]

Processes a `MessageType::INFO_CLIENT_CONNECTED` Message. Parameters

msg The message

Reimplemented from **NetworkComponent** (p. 35).

4.2.2.6 void ProcessNetClientNotifyDisconnect (const MachineInfoMessage & *msg*) [virtual]

Processes a `MessageType::NETCLIENT_NOTIFY_DISCONNECT` Message and removes the client from the connected clients vector. Parameters

msg The message

Reimplemented from **NetworkComponent** (p. 35).

4.2.2.7 void ProcessNetServerAcceptConnection (const MachineInfoMessage & msg) [virtual]

Processes a MessageType::NETSERVER_ACCEPT_CONNECTION Message. Parameters

msg The message

Reimplemented from **NetworkComponent** (p. 35).

4.2.2.8 void ProcessNetServerRejectConnection (const dtGame::NetServerRejectMessage & msg) [virtual]

Processes a MessageType::NETSERVER_REJECT_CONNECTION Message. Parameters

msg The message

Reimplemented from **NetworkComponent** (p. 36).

4.2.2.9 void ProcessNetServerRejectMessage (const dtGame::ServerMessageRejected & msg) [virtual]

Processes a MessageType::SERVER_REQUEST_REJECTED Message. Parameters

msg The message

Reimplemented from **NetworkComponent** (p. 36).

4.2.2.10 void SendRequestConnectionMessage ()

Utility method to put together a request connection message and send it. Note - Your client will not be able to send or receive messages from the server until you do this. This has no effect if you are not connected to a server yet.

4.2.2.11 bool SetupClient (const std::string & host, const int portNum)

Setup a new network connection to a server. Call ConnectToServer to establish a connection Parameters

host The hostname to connect to

portNum The port to connect to

Returns boolean indicating a successful connection

4.2.3 Member Data Documentation**4.2.3.1 const std::string DEFAULT_NAME = "ClientNetworkComponent" [static]****4.2.3.2 std::vector< dtCore::RefPtr<dtGame::MachineInfo> > mConnectedClients [protected]**

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

- **clientnetworkcomponent.h**
- **clientnetworkcomponent.cpp**

4.3 DataStreamPacket Class Reference

A **DataStreamPacket** (p. 16) contains a dtGame::Message blocks to be sent over a Network of other connected dtGame::GameManager's.

```
#include <inc/dtNetGM/datastreampacket.h>
```

Public Types

- typedef GNE::SmartPtr< **DataStreamPacket** > **sptr**
- typedef GNE::WeakPtr< **DataStreamPacket** > **wptr**

Public Member Functions

- **DataStreamPacket** (const **DataStreamPacket** &dataStreamPacket)
*Copy constructor, used by the GNE::PacketParser to send a copy of the **DataStreamPacket** (p. 16) across the Network.*
- **DataStreamPacket** (GNE::guint16 streamId, GNE::guint16 dataStreamSize, GNE::guint8 packetId=0)
Constructor.
- **DataStreamPacket** ()
Constructor.
- virtual ~**DataStreamPacket** (void)
Destructor, public for GNE.....
- unsigned int **GetDataStreamId** ()
- unsigned int **GetDataStreamSize** ()
- unsigned int **GetIndex** ()
- unsigned int **GetPacketCount** ()
- GNE::gbyte * **GetPayloadBuffer** ()
- unsigned int **GetPayloadSize** ()
- virtual int **getSize** () const
*Gets the size of the **DataStreamPacket** (p. 16), used by GNE::PacketParser.*
- virtual void **readPacket** (GNE::Buffer &raw)
*Reads a **DataStreamPacket** (p. 16) from a packet stream, used by GNE::PacketParser.*
- void **SetDataStreamId** (GNE::guint16 id)
- void **SetDataStreamSize** (GNE::guint16 size)
- void **SetIndex** (GNE::guint8 index)
- void **SetPacketCount** (GNE::guint8 count)
- void **SetPayloadSize** (GNE::guint16 size)
- virtual void **writePacket** (GNE::Buffer &raw) const
*Writes a **DataStreamPacket** (p. 16) into a packet stream, used by GNE::PacketParser.*

Static Public Attributes

- static const int **ID** = GNE::PacketParser::MIN_USER_ID + 1
*ID used by GNE to identify the **DataStreamPacket** (p. 16).*
- static const int **MAX_PAYLOAD** = 500

4.3.1 Detailed Description

A **DataStreamPacket** (p. 16) contains a dtGame::Message blocks to be sent over a Network of other connected dtGame::GameManager's. See also dtGame::Message

4.3.2 Member Typedef Documentation

4.3.2.1 typedef GNE::SmartPtr<DataStreamPacket> *sptr*

4.3.2.2 typedef GNE::WeakPtr<DataStreamPacket> *wptr*

4.3.3 Constructor & Destructor Documentation

4.3.3.1 DataStreamPacket ()

Constructor.

4.3.3.2 DataStreamPacket (GNE::guint16 *streamId*, GNE::guint16 *dataStreamSize*, GNE::guint8 *packetId* = 0)

Constructor.

4.3.3.3 DataStreamPacket (const DataStreamPacket & *dataStreamPacket*)

Copy constructor, used by the GNE::PacketParser to send a copy of the **DataStreamPacket** (p. 16) across the Network. Parameters

messagePacket the **DataStreamPacket** (p. 16) to be copied.

4.3.3.4 ~DataStreamPacket (void) [virtual]

Destructor, public for GNE.....

4.3.4 Member Function Documentation

4.3.4.1 unsigned int GetDataStreamId () [inline]

4.3.4.2 unsigned int GetDataStreamSize () [inline]

4.3.4.3 unsigned int GetIndex () [inline]

4.3.4.4 unsigned int GetPacketCount () [inline]

4.3.4.5 GNE::gbyte* GetPayloadBuffer () [inline]

4.3.4.6 unsigned int GetPayloadSize () [inline]

4.3.4.7 int getSize () const [virtual]

Gets the size of the **DataStreamPacket** (p. 16), used by GNE::PacketParser. Returns The size

4.3.4.8 void readPacket (GNE::Buffer & *raw*) [virtual]

Reads a **DataStreamPacket** (p. 16) from a packet stream, used by GNE::PacketParser. Parameters

raw The buffer to read the **DataStreamPacket** (p. 16) from

4.3.4.9 void SetDataStreamId (GNE::guint16 *id*) [inline]

4.3.4.10 void SetDataStreamSize (GNE::guint16 *size*) [inline]

4.3.4.11 void SetIndex (GNE::guint8 *index*) [inline]

4.3.4.12 void SetPacketCount (GNE::guint8 *count*) [inline]

4.3.4.13 void SetPayloadSize (GNE::guint16 *size*) [inline]

4.3.4.14 void writePacket (GNE::Buffer & *raw*) const [virtual]

Writes a **DataStreamPacket** (p. 16) into a packet stream, used by GNE::PacketParser. Parameters

raw The buffer to write the **DataStreamPacket** (p. 16) to

4.3.5 Member Data Documentation

4.3.5.1 `const int ID = GNE::PacketParser::MIN_USER_ID + 1` [static]

ID used by GNE to identify the **DataStreamPacket** (p.16). The **DataStreamPacket** (p.16) has an ID of `GNE::PacketParser::MIN_USER_ID`

4.3.5.2 `const int MAX_PAYLOAD = 500` [static]

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

- `datastreampacket.h`
- `datastreampacket.cpp`

4.4 DestinationType Class Reference

enumeration class to address different stored connections

```
#include <inc/dtNetGM/networkcomponent.h>
```

Static Public Attributes

- static const **DestinationType** **ALL_CLIENTS**
- static const **DestinationType** **ALL_NOT_CLIENTS**
- static const **DestinationType** **DESTINATION**

4.4.1 Detailed Description

enumeration class to address different stored connections

4.4.2 Member Data Documentation

4.4.2.1 `const NetworkComponent::DestinationType ALL_CLIENTS` [static]

4.4.2.2 `const NetworkComponent::DestinationType ALL_NOT_CLIENTS` [static]

4.4.2.3 `const NetworkComponent::DestinationType DESTINATION` [static]

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

- `networkcomponent.h`
- `networkcomponent.cpp`

4.5 MachineInfoMessage Class Reference

A **MachineInfoMessage** (p.20) contains a dtGame::MachineInfo to be used with different messages to send information about a GameManager accross the network.

```
#include <inc/dtNetGM/machineinfomessage.h>
```

Public Member Functions

- **MachineInfoMessage** ()
- const std::string & **GetHostName** () const
Gets the HostName from the contained MachineInfo.
- const std::string & **GetIPAddress** () const
Gets the IpAddress from the contained MachineInfo.
- dtCore::RefPtr< dtGame::MachineInfo > **GetMachineInfo** () const
Gets the MachineInfo from the message.
- const std::string & **GetMachineInfoName** () const
Gets the Name from the contained MachineInfo.
- const unsigned int & **GetPing** () const
Gets the Ping from the contained MachineInfo.
- const unsigned long & **GetTimeStamp** () const
Gets the TimeStamp from the contained MachineInfo.
- const std::string & **GetUniqueld** () const
Gets the Uniqueld from the contained MachineInfo as string.
- void **SetHostName** (const std::string &hostname)
Sets the HostName associated with the MachineInfo.
- void **SetIPAddress** (const std::string &ipAddress)
Sets the IpAddress associated with the MachineInfo.
- void **SetMachineInfo** (const dtGame::MachineInfo &machineInfo)
Sets the MachineInfo.
- void **SetMachineInfoName** (const std::string &name)
Sets the name associated with the MachineInfo.
- void **SetPing** (const unsigned int &ping)
Sets the Ping associated with the MachineInfo.
- void **SetTimeStamp** (const unsigned long &timeStamp)
Sets the TimeStamp associated with the MachineInfo.
- void **SetUniqueld** (const std::string &strId)
Sets the Uniqueld associated with the MachineInfo.

Protected Member Functions

- virtual ~**MachineInfoMessage** ()
Destructor.

4.5.1 Detailed Description

A **MachineInfoMessage** (p.20) contains a dtGame::MachineInfo to be used with different messages to send information about a GameManager across the network. See also dtGame::MachineInfo

4.5.2 Constructor & Destructor Documentation

4.5.2.1 MachineInfoMessage () [inline]

4.5.2.2 virtual ~MachineInfoMessage () [inline, protected, virtual]

Destructor.

4.5.3 Member Function Documentation

4.5.3.1 const std::string & GetHostName () const

Gets the HostName from the contained MachineInfo. Returns The hostname

4.5.3.2 const std::string & GetIPAddress () const

Gets the IpAddress from the contained MachineInfo. Returns The ipaddress

4.5.3.3 dtCore::RefPtr< dtGame::MachineInfo > GetMachineInfo () const

Gets the MachineInfo from the message. Returns The machineinfo

4.5.3.4 const std::string & GetMachineInfoName () const

Gets the Name from the contained MachineInfo. Returns The name

4.5.3.5 const unsigned int & GetPing () const

Gets the Ping from the contained MachineInfo. Returns The ping

4.5.3.6 const unsigned long & GetTimeStamp () const

Gets the TimeStamp from the contained MachineInfo. Returns The timestamp

4.5.3.7 const std::string & GetUniqueld () const

Gets the Uniqueld from the contained MachineInfo as string. Returns The uniqueid

4.5.3.8 void SetHostName (const std::string & *hostname*)

Sets the HostName associated with the MachineInfo. Parameters

The new hostname

4.5.3.9 void SetIPAddress (const std::string & *ipAddress*)

Sets the IpAddress associated with the MachineInfo. Parameters

The new ipaddress

4.5.3.10 void SetMachineInfo (const dtGame::MachineInfo & *machineInfo*)

Sets the MachineInfo. Parameters

The new machineinfo

4.5.3.11 void SetMachineInfoName (const std::string & *name*)

Sets the name associated with the MachineInfo. Parameters

The new name

4.5.3.12 void SetPing (const unsigned int & *ping*)

Sets the Ping associated with the MachineInfo. Parameters

The new ping

4.5.3.13 void SetTimeStamp (const unsigned long & *timeStamp*)

Sets the TimeStamp associated with the MachineInfo. Parameters

The new timestamp

4.5.3.14 void SetUniqueld (const std::string & *strId*)

Sets the Uniqueld associated with the MachineInfo. Parameters

The new uniqueid as string

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

- **machineinfomessage.h**
- **machineinfomessage.cpp**

4.6 MessageActionCode Class Reference

```
#include <inc/dtNetGM/networkcomponent.h>
```

Static Public Attributes

- static **MessageActionCode DROP**
- static **MessageActionCode REJECT**
- static **MessageActionCode SEND**
- static **MessageActionCode WAIT**

4.6.1 Member Data Documentation

4.6.1.1 **MessageActionCode DROP** [static]

4.6.1.2 **MessageActionCode REJECT** [static]

4.6.1.3 **MessageActionCode SEND** [static]

4.6.1.4 **MessageActionCode WAIT** [static]

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

- **networkcomponent.h**
- **networkcomponent.cpp**

4.7 MessagePacket Class Reference

A **MessagePacket** (p.24) contains a dtGame::Message to be sent over a Network of other connected dtGame::GameManager's.

```
#include <inc/dtNetGM/messagepacket.h>
```

Public Types

- typedef GNE::SmartPtr< **MessagePacket** > **sptr**
- typedef GNE::WeakPtr< **MessagePacket** > **wptr**

Public Member Functions

- **MessagePacket** (const dtGame::Message &message)
*Construct a **MessagePacket** (p. 24) from a dtGame::Message.*
- **MessagePacket** (const **MessagePacket** &messagePacket)
*Copy constructor, used by the GNE::PacketParser to send a copy of the **MessagePacket** (p. 24) across the Network.*
- **MessagePacket** ()
Constructor.
- virtual ~**MessagePacket** (void)
Destructor, public for GNE.....
- void **BuildFromMessage** (const dtGame::Message &message)
*Builds a **MessagePacket** (p. 24) from a message.*
- void **FillMessage** (dtGame::Message &message) const
Fills the content of the encapsulated Message into a Message, The message should be created using the Message-Factory.
- const dtCore::Uniqueld **GetDestinatonId** () const
Gets the Uniqueld of the Message destination.
- const unsigned short **GetMessageld** () const
*Gets the MessageType::mID of the Message contained in the **MessagePacket** (p. 24) This can be used to recreate the message with the Messagefactory.*
- const std::string **GetMessageParameters** () const
Gets the message parameters as string.
- virtual int **getSize** () const
*Gets the size of the **MessagePacket** (p. 24), used by GNE::PacketParser.*
- const dtCore::Uniqueld **GetSourceId** () const
Gets the Uniqueld of the Message source.
- void **OverrideDestination** (const dtGame::MachineInfo &destination)
Overrides the destination of the contained Message.
- virtual void **readPacket** (GNE::Buffer &raw)
*Reads a **MessagePacket** (p. 24) from a packet stream, used by GNE::PacketParser.*
- virtual void **writePacket** (GNE::Buffer &raw) const
*Writes a **MessagePacket** (p. 24) into a packet stream, used by GNE::PacketParser.*

Static Public Attributes

- static const int **ID** = GNE::PacketParser::MIN_USER_ID
*ID used by GNE to identify the **MessagePacket** (p. 24).*

4.7.1 Detailed Description

A **MessagePacket** (p. 24) contains a dtGame::Message to be sent over a Network of other connected dtGame::GameManager's. The **MessagePacket** (p. 24) uses the dtGame::MachineInfo::Uniqueld to specify its source and destination dtGame::GameManager Currently it is assumed all messagecontent fits into one packet. See also dtGame::Message

4.7.2 Member Typedef Documentation

4.7.2.1 typedef GNE::SmartPtr<MessagePacket> **sptr**

4.7.2.2 typedef GNE::WeakPtr<MessagePacket> **wptr**

4.7.3 Constructor & Destructor Documentation

4.7.3.1 MessagePacket ()

Constructor.

4.7.3.2 MessagePacket (const MessagePacket & *messagePacket*)

Copy constructor, used by the GNE::PacketParser to send a copy of the **MessagePacket** (p. 24) across the Network. Parameters

messagePacket the **MessagePacket** (p. 24) to be copied.

4.7.3.3 MessagePacket (const dtGame::Message & *message*)

Construct a **MessagePacket** (p. 24) from a dtGame::Message. Parameters

message the Message to be contained in the **MessagePacket** (p. 24).

4.7.3.4 ~MessagePacket (void) [virtual]

Destructor, public for GNE.....

4.7.4 Member Function Documentation

4.7.4.1 void BuildFromMessage (const dtGame::Message & *message*)

Builds a **MessagePacket** (p. 24) from a message. Parameters

the message to be encapsulated by the **MessagePacket** (p. 24)

4.7.4.2 void FillMessage (dtGame::Message & *message*) const

Fills the content of the encapsulated Message into a Message, The message should be created using the MessageFactory. Parameters

The message to write the content to.

4.7.4.3 const dtCore::Uniqueld GetDestinatonId () const [inline]

Gets the Uniqueld of the Message destination. Returns The message destination

4.7.4.4 const unsigned short GetMessageld () const [inline]

Gets the MessageType::mID of the Message contained in the **MessagePacket** (p. 24) This can be used to recreate the message with the Messagefactory. Returns The id of the MessageType

4.7.4.5 const std::string GetMessageParameters () const [inline]

Gets the message parameters as string. Returns The message paramters

4.7.4.6 int getSize () const [virtual]

Gets the size of the **MessagePacket** (p. 24), used by GNE::PacketParser. Returns The size

4.7.4.7 const dtCore::Uniqueld GetSourceId () const [inline]

Gets the Uniqueld of the Message source. Returns The message source

4.7.4.8 void OverrideDestination (const dtGame::MachineInfo & destination)

Overrides the destination of the contained Message. Parameters

The new destination

4.7.4.9 void readPacket (GNE::Buffer & raw) [virtual]

Reads a **MessagePacket** (p. 24) from a packet stream, used by GNE::PacketParser. Parameters

raw The buffer to read the **MessagePacket** (p. 24) from

4.7.4.10 void writePacket (GNE::Buffer & raw) const [virtual]

Writes a **MessagePacket** (p. 24) into a packet stream, used by GNE::PacketParser. Parameters

raw The buffer to write the **MessagePacket** (p. 24) to

4.7.5 Member Data Documentation**4.7.5.1 const int ID = GNE::PacketParser::MIN_USER_ID [static]**

ID used by GNE to identify the **MessagePacket** (p.24). The **MessagePacket** (p.24) has an ID of GNE::PacketParser::MIN_USER_ID

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

- **messagepacket.h**
- **messagepacket.cpp**

4.8 NetworkBridge Class Reference

contains GNE components for communication across a network This class represents a single host on the other side of the network

```
#include <inc/dtNetGM/networkbridge.h>
```

Public Member Functions

- **NetworkBridge** (**NetworkComponent** *networkComp)
- virtual **~NetworkBridge** (void)
- void **Disconnect** (int waitTime=-1)
Disconnects the current connection.
- std::string **GetHostDescription** ()
Returns a string describing the host.
- const dtGame::MachineInfo & **GetMachineInfo** () const
Gets the MachineInfo from the host.
- const bool **IsConnectedClient** () const
Returns if the networkbridge is connected as an accepted client to a server.
- const bool **IsNetworkConnected** ()
Returns if the networkbridge is connected.
- void **OnConnect** (GNE::SyncConnection &conn)
Callback function for GNE::ConnectionListener.
- void **OnConnectFailure** (GNE::Connection &conn, const GNE::Error &error)
Callback function for GNE::ConnectionListener.
- void **OnDisconnect** (GNE::Connection &conn)
Callback function for GNE::ConnectionListener.
- void **OnError** (GNE::Connection &conn, const GNE::Error &error)
Callback function for GNE::ConnectionListener.
- void **OnExit** (GNE::Connection &conn)
Callback function for GNE::ConnectionListener.
- void **OnFailure** (GNE::Connection &conn, const GNE::Error &error)
Callback function for GNE::ConnectionListener.
- void **OnNewConnection** (GNE::SyncConnection &conn)
Callback function for GNE::ConnectionListener.
- void **OnReceive** (GNE::Connection &conn)
Callback function for GNE::ConnectionListener.
- void **OnTimeout** (GNE::Connection &conn)
Callback function for GNE::ConnectionListener.
- void **SendDataStream** (dtUtil::DataStream &dataStream)
Sends a DataStream across the network.
- void **SetClientConnected** (bool client=true)

Sets if the networkbridge is an accepted client.

- void **SetMachineInfo** (const dtGame::MachineInfo &machineInfo)
Sets the MachineInfo.

4.8.1 Detailed Description

contains GNE components for communication across a network This class represents a single host on the other side of the network

4.8.2 Constructor & Destructor Documentation

4.8.2.1 NetworkBridge (NetworkComponent * *networkComp*)

4.8.2.2 ~NetworkBridge (void) [virtual]

4.8.3 Member Function Documentation

4.8.3.1 void Disconnect (int *waitTime* = -1)

Disconnects the current connection.

4.8.3.2 std::string GetHostDescription ()

Returns a string describing the host. Returns string describing the host

4.8.3.3 const dtGame::MachineInfo & GetMachineInfo () const

Gets the MachineInfo from the host. Returns The name

4.8.3.4 const bool IsConnectedClient () const

Returns if the networkbridge is connected as an accepted client to a server. Returns is connected client

4.8.3.5 const bool IsNetworkConnected ()

Returns if the networkbridge is connected. Returns network connected

4.8.3.6 void OnConnect (GNE::SyncConnection & *conn*)

Callback function for GNE::ConnectionListener. Parameters

The GNE::SyncConnection

4.8.3.7 void OnConnectFailure (GNE::Connection & *conn*, const GNE::Error & *error*)

Callback function for GNE::ConnectionListener. Parameters

conn The GNE::Connection

error The GNE::Error description

4.8.3.8 void OnDisconnect (GNE::Connection & *conn*)

Callback function for GNE::ConnectionListener. Parameters

The GNE::Connection

4.8.3.9 void OnError (GNE::Connection & *conn*, const GNE::Error & *error*)

Callback function for GNE::ConnectionListener. Parameters

conn The GNE::Connection

error The GNE::Error description

4.8.3.10 void OnExit (GNE::Connection & *conn*)

Callback function for GNE::ConnectionListener. Parameters

The GNE::Connection

4.8.3.11 void OnFailure (GNE::Connection & *conn*, const GNE::Error & *error*)

Callback function for GNE::ConnectionListener. Parameters

conn The GNE::Connection

error The GNE::Error description

4.8.3.12 void OnNewConnection (GNE::SyncConnection & *conn*)

Callback function for GNE::ConnectionListener. Parameters

The GNE::SyncConnection

4.8.3.13 void OnReceive (GNE::Connection & *conn*)

Callback function for GNE::ConnectionListener. Parameters

The GNE::Connection

4.8.3.14 void OnTimeout (GNE::Connection & *conn*)

Callback function for GNE::ConnectionListener. Parameters

The GNE::Connection

4.8.3.15 void SendDataStream (dtUtil::DataStream & *dataStream*)

Sends a DataStream across the network. Parameters

The messagepacket

4.8.3.16 void SetClientConnected (bool *client* = true) [inline]

Sets if the networkbridge is an accepted client. Parameters

The client state

4.8.3.17 void SetMachineInfo (const dtGame::MachineInfo & *machineInfo*)

Sets the MachineInfo. Parameters

The machineinfo

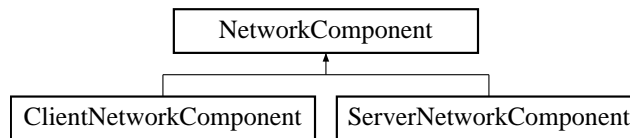
The documentation for this class was generated from the following files:

- **networkbridge.h**
- **networkbridge.cpp**

4.9 NetworkComponent Class Reference

baseclass GMComponent to communicate as client - server

#include <inc/dtNetGM/networkcomponent.h> Inheritance diagram for NetworkComponent::



Classes

- class **DestinationType**
enumeration class to address different stored connections

Public Member Functions

- **NetworkComponent** (const std::string &gameName, const int gameVersion, const std::string &logFile="")
*Construct a **NetworkComponent** (p. 30) with a game name and version to be used by GNE Calls InitializeNetwork to initialize GNE.*
- dtUtil::DataStream **CreateDataStream** (const dtGame::Message &message)
- dtCore::RefPtr< dtGame::Message > **CreateMessage** (dtUtil::DataStream &dataStream, const **NetworkBridge** &networkBridge)
- virtual void **Disconnect** ()
Disconnect from all active network connections.
- virtual void **DispatchNetworkMessage** (const dtGame::Message &message)
Function called by a GameManager to send Messages across a Network.
- void **GetConnectedClients** (std::vector< **NetworkBridge** * > &connectedClients)
*Retrieves a vector containing all **NetworkBridge** (p. 27) which have an accepted client connection.*
- std::string **GetHostName** ()
*Return hostname of the machine this **NetworkComponent** (p. 30) is running on.*
- const bool & **IsGneInitialized** () const
Is GNE already initialized.
- const bool & **IsReliable** () const
Is our GNE connection reliable.
- virtual const bool **IsServer** () const
*Is this **NetworkComponent** (p. 30) a server?*
- bool **IsShuttingDown** ()
Are we shutting down?
- virtual void **OnAddedToGM** ()
Called immediately after a component is added to the GM.
- virtual **MessageActionCode** & **OnBeforeSendMessage** (const dtGame::Message &message, std::string &rejectReason)
This is called when the component is ready to send a message to the game manager so the user.

- virtual void **OnConnect** (**NetworkBridge** &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a new connection is made.*
- virtual void **OnConnectFailure** (**NetworkBridge** &networkBridge, const GNE::Error &error)
*Function called by a **NetworkBridge** (p. 27) if a connection failure occurs.*
- virtual void **OnDisconnect** (**NetworkBridge** &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a connection disconnects.*
- virtual void **OnError** (**NetworkBridge** &networkBridge, const GNE::Error &error)
*Function called by a **NetworkBridge** (p. 27) if an error occurs.*
- virtual void **OnExit** (**NetworkBridge** &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a connection exits.*
- virtual void **OnFailure** (**NetworkBridge** &networkBridge, const GNE::Error &error)
*Function called by a **NetworkBridge** (p. 27) if a failure occurs.*
- virtual void **OnNewConnection** (**NetworkBridge** &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a new connection is received.*
- virtual void **OnReceivedDataStream** (**NetworkBridge** &networkBridge, dtUtil::DataStream &dataStream)
Function called by a Networkbridge to Signal a received DataStream The network component creates the message and checks for destination and / or connection info contained in the message.
- virtual void **OnReceivedNetworkMessage** (const dtGame::Message &message, **NetworkBridge** &networkBridge)
- virtual void **OnRemovedFromGM** ()
Overridden to handle shutdown.
- virtual void **OnTimeOut** (**NetworkBridge** &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a timeout occurs.*
- virtual void **ProcessInfoClientConnected** (const **MachineInfoMessage** &msg)
Processes a MessageType::INFO_CLIENT_CONNECTED Message.
- virtual void **ProcessMessage** (const dtGame::Message &message)
Function called by a GameManager to process Messages.
- virtual void **ProcessNetClientNotifyDisconnect** (const **MachineInfoMessage** &msg)
Processes a MessageType::NETCLIENT_NOTIFY_DISCONNECT Message.
- virtual void **ProcessNetClientRequestConnection** (const **MachineInfoMessage** &msg)
Processes a MessageType::NETCLIENT_REQUEST_CONNECTION Message.
- virtual void **ProcessNetServerAcceptConnection** (const **MachineInfoMessage** &msg)
Processes a MessageType::NETSERVER_ACCEPT_CONNECTION Message.
- virtual void **ProcessNetServerRejectConnection** (const dtGame::NetServerRejectMessage &msg)
Processes a MessageType::NETSERVER_REJECT_CONNECTION Message.
- virtual void **ProcessNetServerRejectMessage** (const dtGame::ServerMessageRejected &msg)
Processes a MessageType::SERVER_REQUEST_REJECTED Message.
- virtual void **ProcessTickLocal** (const dtGame::TickMessage &msg)

Processes a MessageType::TICK_LOCAL Message.

- void **SendNetworkMessage** (const dtGame::Message &message, const **DestinationType** &destination-Type=**DestinationType::DESTINATION**)
- void **SetConnectionParameters** (bool reliable=true, int bandWidthIn=0, int bandWidthOut=0)
Sets the connection Parameters to be used by GNE.
- virtual void **ShutdownNetwork** ()
Shutdown network communications and clear connections plus shutdown the internal networking subsystem.

Protected Member Functions

- virtual ~**NetworkComponent** (void)
- void **AddConnection** (**NetworkBridge** *networkBridge)
*Adds a **NetworkBridge** (p. 27) to the map.*
- **NetworkBridge** * **GetConnection** (const dtGame::MachineInfo &machineInfo)
*Retrieves a **NetworkBridge** (p. 27) from the map If no networkbridge is found, NULL is returned.*
- virtual const dtGame::MachineInfo * **GetMachineInfo** (const dtCore::UniqueId &uniqueId)
Retrieves a dtGame::MachineInfo from the stored connections If no matching connection is found, NULL is returned.*
- void **RemoveConnection** (const dtGame::MachineInfo &machineInfo)
*Removes a **NetworkBridge** (p. 27) from the map.*

Protected Attributes

- OpenThreads::Mutex **mBufferMutex**
- std::vector< **NetworkBridge** * > **mConnections**
- OpenThreads::Mutex **mMutex**
- int **mRateIn**
- int **mRateOut**
- bool **mReliable**

4.9.1 Detailed Description

baseclass GMComponent to communicate as client - server

4.9.2 Constructor & Destructor Documentation

4.9.2.1 **NetworkComponent** (const std::string & *gameName*, const int *gameVersion*, const std::string & *logFile* = "")

Construct a **NetworkComponent** (p. 30) with a game name and version to be used by GNE Calls InitializeNetwork to initialize GNE. Parameters

gameName The game name

gameVersion The game version

logFile The logfile name

4.9.2.2 ~**NetworkComponent** (void) [protected, virtual]

4.9.3 Member Function Documentation

4.9.3.1 void **AddConnection** (**NetworkBridge** * *networkBridge*) [protected]

Adds a **NetworkBridge** (p. 27) to the map. Parameters

The networkbridge

4.9.3.2 dtUtil::DataStream CreateDataStream (const dtGame::Message & message)

4.9.3.3 dtCore::RefPtr< dtGame::Message > CreateMessage (dtUtil::DataStream & dataStream, const NetworkBridge & networkBridge)

4.9.3.4 void Disconnect () [virtual]

Disconnect from all active network connections.

4.9.3.5 void DispatchNetworkMessage (const dtGame::Message & message) [virtual]

Function called by a GameManager to send Messages across a Network. Parameters

The Message to be sent

4.9.3.6 void GetConnectedClients (std::vector< NetworkBridge * > & connectedClients)

Retrieves a vector containing all **NetworkBridge** (p.27) which have an accepted client connection. Parameters

The vector to be filled

4.9.3.7 NetworkBridge * GetConnection (const dtGame::MachineInfo & machineInfo) [protected]

Retrieves a **NetworkBridge** (p.27) from the map If no networkbridge is found, NULL is returned. Parameters

The machineinfo

Returns Pointer to the networkbridge

4.9.3.8 std::string GetHostName ()

Return hostname of the machine this **NetworkComponent** (p.30) is running on.

4.9.3.9 const dtGame::MachineInfo * GetMachineInfo (const dtCore::Uniqueld & uniqueld) [protected, virtual]

Retrieves a dtGame::MachineInfo* from the stored connections If no matching connection is found, NULL is returned. Parameters

dtCore::Uniqueld of the MachineInfo

Returns Pointer to the dtGame::MachineInfo* or NULL

Reimplemented in **ClientNetworkComponent** (p.14).

4.9.3.10 const bool& IsGneInitialized () const [inline]

Is GNE already initialized. Returns GNE initialization

4.9.3.11 const bool& IsReliable () const [inline]

Is our GNE connection reliable. Returns The reliability of the connection

4.9.3.12 virtual const bool IsServer () const [inline, virtual]

Is this **NetworkComponent** (p.30) a server? Returns server true/false

4.9.3.13 bool IsShuttingDown () [inline]

Are we shutting down? Returns ShuttingDown

4.9.3.14 void OnAddedToGM () [virtual]

Called immediately after a component is added to the GM. Used to register 'additional' Network Messages on the GameManager

4.9.3.15 **MessageActionCode & OnBeforeSendMessage (const dtGame::Message & *message*, std::string & *rejectReason*) [virtual]**

This is called when the component is ready to send a message to the game manager so the user. can decide what to do with it. By default, the message is sent unless the gm is waiting for a map change to complete, then it returns wait.

4.9.3.16 **void OnConnect (NetworkBridge & *networkBridge*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if a new connection is made. Parameters

The NetworkBridge (p. 27)

4.9.3.17 **void OnConnectFailure (NetworkBridge & *networkBridge*, const GNE::Error & *error*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if a connection failure occurs. Parameters

networkBridge The **NetworkBridge** (p. 27)

error The GNE::Error description

4.9.3.18 **void OnDisconnect (NetworkBridge & *networkBridge*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if a connection disconnects. Parameters

The NetworkBridge (p. 27)

Reimplemented in **ClientNetworkComponent** (p. 14), and **ServerNetworkComponent** (p. 40).

4.9.3.19 **void OnError (NetworkBridge & *networkBridge*, const GNE::Error & *error*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if an error occurs. Parameters

networkBridge The **NetworkBridge** (p. 27)

error The GNE::Error description

4.9.3.20 **void OnExit (NetworkBridge & *networkBridge*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if a connection exits. Parameters

The NetworkBridge (p. 27)

4.9.3.21 **void OnFailure (NetworkBridge & *networkBridge*, const GNE::Error & *error*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if a failure occurs. Parameters

networkBridge The **NetworkBridge** (p. 27)

error The GNE::Error description

4.9.3.22 **void OnNewConnection (NetworkBridge & *networkBridge*) [virtual]**

Function called by a **NetworkBridge** (p. 27) if a new connection is received. Parameters

The NetworkBridge (p. 27)

4.9.3.23 void OnReceivedDataStream (NetworkBridge & *networkBridge*, dtUtil::DataStream & *dataStream*) [virtual]

Function called by a Networkbridge to Signal a received DataStream The network component creates the message and checks for destination and / or connection info contained in the message. If appropriate, the message is delivered to the GameManager Parameters

networkBridge The **NetworkBridge** (p. 27) which received the MessagePakcet

dataStream The DataStream received

4.9.3.24 void OnReceivedNetworkMessage (const dtGame::Message & *message*, NetworkBridge & *networkBridge*) [virtual]

4.9.3.25 void OnRemovedFromGM () [virtual]

Overridden to handle shutdown.

4.9.3.26 void OnTimeOut (NetworkBridge & *networkBridge*) [virtual]

Function called by a **NetworkBridge** (p. 27) if a timeout occurs. Parameters

networkBridge The **NetworkBridge** (p. 27)

4.9.3.27 virtual void ProcessInfoClientConnected (const MachineInfoMessage & *msg*) [inline, virtual]

Processes a MessageType::INFO_CLIENT_CONNECTED Message. Parameters

msg The message

Reimplemented in **ClientNetworkComponent** (p. 14).

4.9.3.28 void ProcessMessage (const dtGame::Message & *message*) [virtual]

Function called by a GameManager to process Messages. This function forwards the connection related messages to the functions processing these messages. Parameters

The Message to be process

4.9.3.29 virtual void ProcessNetClientNotifyDisconnect (const MachineInfoMessage & *msg*) [inline, virtual]

Processes a MessageType::NETCLIENT_NOTIFY_DISCONNECT Message. Parameters

msg The message

Reimplemented in **ClientNetworkComponent** (p. 14).

4.9.3.30 virtual void ProcessNetClientRequestConnection (const MachineInfoMessage & *msg*) [inline, virtual]

Processes a MessageType::NETCLIENT_REQUEST_CONNECTION Message. Parameters

msg The message

Reimplemented in **ServerNetworkComponent** (p. 40).

4.9.3.31 virtual void ProcessNetServerAcceptConnection (const MachineInfoMessage & *msg*) [inline, virtual]

Processes a MessageType::NETSERVER_ACCEPT_CONNECTION Message. Parameters

msg The message

Reimplemented in **ClientNetworkComponent** (p. 15).

4.9.3.32 virtual void ProcessNetServerRejectConnection (const dtGame::NetServerRejectMessage & *msg*) [inline, virtual]

Processes a MessageType::NETSERVER_REJECT_CONNECTION Message. Parameters

msg The message

Reimplemented in **ClientNetworkComponent** (p. 15).

4.9.3.33 virtual void ProcessNetServerRejectMessage (const dtGame::ServerMessageRejected & *msg*) [inline, virtual]

Processes a MessageType::SERVER_REQUEST_REJECTED Message. Parameters

msg The message

Reimplemented in **ClientNetworkComponent** (p. 15).

4.9.3.34 void ProcessTickLocal (const dtGame::TickMessage & *msg*) [virtual]

Processes a MessageType::TICK_LOCAL Message. Parameters

msg The message

4.9.3.35 void RemoveConnection (const dtGame::MachineInfo & *machineInfo*) [protected]

Removes a **NetworkBridge** (p. 27) from the map. Parameters

The MachineInfo of the connection to be removed

4.9.3.36 void SendNetworkMessage (const dtGame::Message & *message*, const DestinationType & *destinationType* = DestinationType::DESTINATION)

4.9.3.37 void SetConnectionParameters (bool *reliable* = true, int *bandWidthIn* = 0, int *bandWidthOut* = 0)

Sets the connection Parameters to be used by GNE. Parameters

reliable The reliability of the connection

bandWidthIn The incoming bandwidth throttle

bandWidthOut The outgoing bandwidth throttle

4.9.3.38 void ShutdownNetwork () [virtual]

Shutdown network communications and clear connections plus shutdown the internal networking subsystem.

4.9.4 Member Data Documentation

4.9.4.1 OpenThreads::Mutex mBufferMutex [protected]

4.9.4.2 std::vector<NetworkBridge*> mConnections [protected]

4.9.4.3 OpenThreads::Mutex mMutex [protected]

4.9.4.4 int mRateIn [protected]

4.9.4.5 int mRateOut [protected]

4.9.4.6 bool mReliable [protected]

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

- **networkcomponent.h**
- **networkcomponent.cpp**

4.10 ServerConnectionListener Class Reference

This class is used as an interface to the GNE::Server connection.

```
#include <inc/dtNetGM/serverconnectionlistener.h>
```

Public Types

- typedef GNE::SmartPtr< **ServerConnectionListener** > **sptr**
- typedef GNE::WeakPtr< **ServerConnectionListener** > **wptr**

Public Member Functions

- **ServerConnectionListener** (**ServerNetworkComponent** *serverNetworkComp, int inRate, int outRate, bool reliable)
*Construct a **ServerConnectionListener** (p. 37).*
- virtual ~**ServerConnectionListener** (void)
- virtual void **getNewConnectionParams** (GNE::ConnectionParams ¶ms)
- virtual void **onListenFailure** (const GNE::Error &error, const GNE::Address &from, const GNE::ConnectionListener::sptr &listener)
- virtual void **onListenSuccess** (const GNE::ConnectionListener::sptr &listener)

Static Public Member Functions

- static **sptr Create** (**ServerNetworkComponent** *serverNetworkComp, int inRate, int outRate, bool reliable)
*Method used to create a new instance of **ServerConnectionListener** (p. 37).*

Protected Attributes

- int **mInRate**
- GNE::Mutex **mMutex**
- int **mOutRate**
The incoming bandwidth rate.
- bool **mReliable**
The outgoing bandwidth rate.
- dtCore::RefPtr< **ServerNetworkComponent** > **mServerNetworkcomponent**
The reliability of the connection.

4.10.1 Detailed Description

This class is used as an interface to the GNE::Server connection. It is used internally by the **ServerNetworkComponent** (p. 39) and is typically not used directly by the end user. This class takes in a reference to a **ServerNetworkComponent** (p. 39) and calls it's virtual methods to mimic the GNE::ServerConnectionListener's callbacks.

4.10.2 Member Typedef Documentation

4.10.2.1 typedef GNE::SmartPtr<ServerConnectionListener> **sptr**

4.10.2.2 typedef GNE::WeakPtr<ServerConnectionListener> **wptr**

4.10.3 Constructor & Destructor Documentation

4.10.3.1 **ServerConnectionListener** (**ServerNetworkComponent** * *serverNetworkComp*, int *inRate*, int *outRate*, bool *reliable*)

Construct a **ServerConnectionListener** (p. 37). Parameters

serverNetworkComp : instance of a **ServerNetworkComponent** (p. 39)

inRate : the incoming bandwidth throttle

outRate : the outgoing bandwidth throttle

reliable : reliability of the Network connection

4.10.3.2 **~ServerConnectionListener (void)** [virtual]

4.10.4 Member Function Documentation

4.10.4.1 **static *sptr* Create (ServerNetworkComponent * *serverNetworkComp*, int *inRate*, int *outRate*, bool *reliable*)** [inline, static]

Method used to create a new instance of **ServerConnectionListener** (p. 37).

4.10.4.2 **void getNewConnectionParams (GNE::ConnectionParams & *params*)** [virtual]

4.10.4.3 **void onListenFailure (const GNE::Error & *error*, const GNE::Address & *from*, const GNE::ConnectionListener::sptr & *listener*)** [virtual]

4.10.4.4 **void onListenSuccess (const GNE::ConnectionListener::sptr & *listener*)** [virtual]

4.10.5 Member Data Documentation

4.10.5.1 **int *mInRate*** [protected]

4.10.5.2 **GNE::Mutex *mMutex*** [protected]

4.10.5.3 **int *mOutRate*** [protected]

The incoming bandwidth rate.

4.10.5.4 **bool *mReliable*** [protected]

The outgoing bandwidth rate.

4.10.5.5 **dtCore::RefPtr<ServerNetworkComponent> *mServerNetworkcomponent*** [protected]

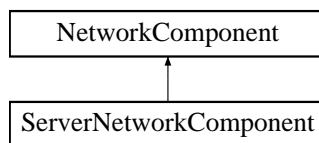
The reliability of the connection.

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

- **serverconnectionlistener.h**
- **serverconnectionlistener.cpp**

4.11 ServerNetworkComponent Class Reference

#include <inc/dtNetGM/servernetworkcomponent.h> Inheritance diagram for ServerNetworkComponent::



Public Member Functions

- **ServerNetworkComponent** (const std::string &gameName, const int gameVersion, const std::string &logFile="")
*Construct a **ServerNetworkComponent** (p. 39) with a game name and version to be used by GNE Calls base class constructor to initialize GNE.*
- virtual void **OnDisconnect** (**NetworkBridge** &networkBridge)
*Function called by a **NetworkBridge** (p. 27) if a connection disconnects.*
- virtual void **OnListenFailure** (const GNE::Error &error, const GNE::Address &from, const GNE::ConnectionListener::sptr &listener)
callback to signal the connection to the socket failed
- virtual void **OnListenSuccess** ()
callback to signal a connection is successful
- virtual void **ProcessNetClientNotifyDisconnect** (const dtGame::Message &msg)
Processes a MessageType::NETCLIENT_NOTIFY_DISCONNECT Message.
- virtual void **ProcessNetClientRequestConnection** (const **MachineInfoMessage** &msg)
Processes a MessageType::NETCLIENT_REQUEST_CONNECTION Message.
- bool **SetupServer** (int portNum)
Start a Server.

Static Public Attributes

- static const dtUtil::RefString **DEFAULT_NAME**

Protected Member Functions

- ~**ServerNetworkComponent** (void)
- virtual bool **AcceptClient** (const dtGame::MachineInfo &machineInfo, std::string &rejectionReason)
Function to accept or deny a client connection request.
- virtual void **SendConnectedClientMessage** (const dtGame::MachineInfo &machineInfo)
Sends a INFO_CLIENT_CONNECTED messages to the new client, informing the new client of existing connections.
- virtual void **SendInfoClientConnectedMessage** (const dtGame::MachineInfo &machineInfo)
Sends a INFO_CLIENT_CONNECTED messages to the already connected clients.

Protected Attributes

- bool **mAcceptClients**

4.11.1 Constructor & Destructor Documentation

4.11.1.1 **ServerNetworkComponent** (const std::string & *gameName*, const int *gameVersion*, const std::string & *logFile* = "")

Construct a **ServerNetworkComponent** (p. 39) with a game name and version to be used by GNE Calls base class constructor to initialize GNE. Parameters

gameName The game name

gameVersion The game version

logFile The logfile name

4.11.1.2 **~ServerNetworkComponent** (void) [protected]

4.11.2 Member Function Documentation

4.11.2.1 **bool AcceptClient** (const dtGame::MachineInfo & *machineInfo*, std::string & *rejectionReason*) [protected, virtual]

Function to accept or deny a client connection request. Parameters

machineInfo The MachineInfo of the new client

rejectionReason Rejection reason which is send to the client if denied

Returns boolean indicating if a client should be accepted

4.11.2.2 **void OnDisconnect** (NetworkBridge & *networkBridge*) [virtual]

Function called by a **NetworkBridge** (p. 27) if a connection disconnects. Parameters

The NetworkBridge (p. 27)

Reimplemented from **NetworkComponent** (p. 34).

4.11.2.3 **void OnListenFailure** (const GNE::Error & *error*, const GNE::Address & *from*, const GNE::ConnectionListener::sptr & *listener*) [virtual]

callback to signal the connection to the socket failed

4.11.2.4 **void OnListenSuccess** () [virtual]

callback to signal a connection is successful

4.11.2.5 **void ProcessNetClientNotifyDisconnect** (const dtGame::Message & *msg*) [virtual]

Processes a MessageType::NETCLIENT_NOTIFY_DISCONNECT Message. Parameters

msg The message

4.11.2.6 **void ProcessNetClientRequestConnection** (const MachineInfoMessage & *msg*) [virtual]

Processes a MessageType::NETCLIENT_REQUEST_CONNECTION Message. Parameters

msg The message

Reimplemented from **NetworkComponent** (p. 35).

4.11.2.7 **void SendConnectedClientMessage** (const dtGame::MachineInfo & *machineInfo*) [protected, virtual]

Sends a INFO_CLIENT_CONNECTED messages to the new client, informing the new client of existing connections. Parameters

machineInfo The MachineInfo of the new client

4.11.2.8 void SendInfoClientConnectedMessage (const dtGame::MachineInfo & *machineInfo*)
[protected, virtual]

Sends a INFO_CLIENT_CONNECTED messages to the already connected clients. Parameters

machineInfo The MachineInfo of the new client

4.11.2.9 bool SetupServer (int *portNum*)

Start a Server. Parameters

The port number used by the server

4.11.3 Member Data Documentation**4.11.3.1 const dtUtil::RefString DEFAULT_NAME [static]****4.11.3.2 bool mAcceptClients [protected]**

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

- **servernetworkcomponent.h**
- **servernetworkcomponent.cpp**

File Documentation

5.1 clientconnectionlistener.cpp File Reference

```
#include <dtNetGM/clientconnectionlistener.h>
#include <dtNetGM/networkbridge.h>
#include <dtNetGM/networkcomponent.h>
#include <dtUtil/log.h>
#include <gnelib/Error.h>
#include <gnelib/Connection.h>
#include <gnelib/SyncConnection.h>
#include <gnelib/PingPacket.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.2 clientconnectionlistener.h File Reference

```
#include <gnelib/ConnectionListener.h>
```

```
#include <dtCore/refptr.h>
```

Classes

- class **ClientConnectionListener**
Provides the interface to a GNE::Connection.

Namespaces

- namespace **dtNetGM**
*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.3 clientnetworkcomponent.cpp File Reference

```
#include <dtNetGM/clientnetworkcomponent.h>
#include <dtNetGM/clientconnectionlistener.h>
#include <dtNetGM/networkbridge.h>
#include <dtNetGM/machineinfomessage.h>
#include <dtNetGM/networkcomponent.h>
#include <dtGame/basemessages.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.4 clientnetworkcomponent.h File Reference

```
#include <dtNetGM/export.h>
```

```
#include <dtNetGM/networkcomponent.h>
```

Classes

- class **ClientNetworkComponent**

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.5 datastreampacket.cpp File Reference

```
#include <dtNetGM/datastreampacket.h>
#include <iostream>
#include <gnelib.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.6 datastreampacket.h File Reference

```
#include <dtNetGM/export.h>
#include <gnelib.h>
#include <dtCore/refptr.h>
#include <dtCore/uniqueid.h>
```

Classes

- class **DataStreamPacket**

A **DataStreamPacket** (p. 16) contains a `dtGame::Message` blocks to be sent over a Network of other connected `dtGame::GameManager`'s.

Namespaces

- namespace **dtGame**
- namespace **dtNetGM**

The **dtNetGM** (p. 9) namespace contains networking classes.

5.7 dtnetgm.h File Reference

```
#include <dtNetGM/machineinfomessage.h>
#include <dtNetGM/networkbridge.h>
#include <dtNetGM/networkcomponent.h>
#include <dtNetGM/clientnetworkcomponent.h>
#include <dtNetGM/servernetworkcomponent.h>
```

Namespaces

- namespace **dtNetGM**
*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.8 export.h File Reference

Defines

- #define **DT_NETGM_EXPORT**

5.8.1 Define Documentation

5.8.1.1 #define **DT_NETGM_EXPORT**

5.9 machineinfomessage.cpp File Reference

```
#include <dtNetGM/machineinfomessage.h>
#include <dtGame/message.h>
#include <dtGame/machineinfo.h>
#include <dtGame/messageparameter.h>
```

Namespaces

- namespace **dtNetGM**
*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.10 machineinfomessage.h File Reference

```
#include <string>
#include <dtNetGM/export.h>
#include <dtCore/refptr.h>
#include <dtGame/message.h>
#include <dtGame/messageparameter.h>
```

Classes

- class **MachineInfoMessage**

A **MachineInfoMessage** (p. 20) contains a `dtGame::MachineInfo` to be used with different messages to send information about a `GameManager` accross the network.

Namespaces

- namespace **dtNetGM**

The **dtNetGM** (p. 9) namespace contains networking classes.

5.11 mainpage.h File Reference

5.11.1 Detailed Description

This file contains Doxygen special commands and text for the **Main Page** (p. ??) and some other minor aspects of this documentation. It is not part of Delta3D.

5.12 messagepacket.cpp File Reference

```
#include <dtNetGM/messagepacket.h>
#include <iostream>
#include <gnelib.h>
#include <dtGame/messagetype.h>
#include <dtGame/message.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.13 messagepacket.h File Reference

```
#include <dtNetGM/export.h>
#include <gnelib.h>
#include <dtCore/refptr.h>
#include <dtCore/uniqueid.h>
```

Classes

- class **MessagePacket**

A **MessagePacket** (p. 24) contains a `dtGame::Message` to be sent over a Network of other connected `dtGame::GameManager`'s.

Namespaces

- namespace **dtGame**
- namespace **dtNetGM**

The **dtNetGM** (p. 9) namespace contains networking classes.

5.14 networkbridge.cpp File Reference

```
#include <queue>
#include <dtNetGM/networkbridge.h>
#include <dtNetGM/networkcomponent.h>
#include <dtNetGM/datastreampacket.h>
#include <dtGame/machineinfo.h>
#include <dtNetGM/machineinfomessage.h>
#include <gnelib.h>
#include <dtUtil/log.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.15 networkbridge.h File Reference

```
#include <dtNetGM/export.h>
#include <gnelib/ConnectionListener.h>
#include <gnelib/Error.h>
#include <dtCore/refptr.h>
#include <dtCore/uniqueid.h>
#include <dtCore/base.h>
#include <dtUtil/datastream.h>
```

Classes

- class **NetworkBridge**

contains GNE components for communication across a network This class represents a single host on the other side of the network

Namespaces

- namespace **dtGame**
- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.16 networkcomponent.cpp File Reference

```
#include <dtNetGM/networkcomponent.h>
#include <dtNetGM/datastreampacket.h>
#include <dtNetGM/machineinfomessage.h>
#include <dtNetGM/networkbridge.h>
#include <dtGame/message.h>
#include <dtGame/messagetype.h>
#include <dtGame/messagefactory.h>
#include <dtGame/basemessages.h>
#include <dtUtil/log.h>
#include <dtCore/system.h>
#include <OpenThreads/ScopedLock>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

Functions

- **IMPLEMENT_ENUM** (NetworkComponent::DestinationType)
- **IMPLEMENT_ENUM** (MessageActionCode)
- **IMPLEMENT_MANAGEMENT_LAYER** (NetworkComponent)

5.17 networkcomponent.h File Reference

```
#include <dtNetGM/export.h>
#include <string>
#include <gnelib.h>
#include <dtGame/gmcomponent.h>
#include <dtUtil/enumeration.h>
#include <OpenThreads/ReentrantMutex>
#include <deque>
```

Classes

- class **DestinationType**
enumeration class to address different stored connections
- class **MessageActionCode**
- class **NetworkComponent**
baseclass GMComponent to communicate as client - server

Namespaces

- namespace **dtCore**
- namespace **dtGame**
- namespace **dtNetGM**
*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.18 serverconnectionlistener.cpp File Reference

```
#include <dtNetGM/serverconnectionlistener.h>
#include <dtNetGM/clientconnectionlistener.h>
#include <dtNetGM/networkbridge.h>
#include <dtNetGM/networkcomponent.h>
#include <dtNetGM/servernetworkcomponent.h>
#include <gnelib/ConnectionParams.h>
#include <dtUtil/log.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.19 serverconnectionlistener.h File Reference

```
#include <dtNetGM/export.h>
```

```
#include <gnelib.h>
```

```
#include <dtCore/refptr.h>
```

Classes

- class **ServerConnectionListener**

This class is used as an interface to the GNE::Server connection.

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.20 servernetworkcomponent.cpp File Reference

```
#include <dtNetGM/servernetworkcomponent.h>
#include <dtNetGM/networkcomponent.h>
#include <dtNetGM/networkbridge.h>
#include <dtNetGM/serverconnectionlistener.h>
#include <dtNetGM/machineinfomessage.h>
#include <dtGame/basemessages.h>
```

Namespaces

- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

5.21 servernetworkcomponent.h File Reference

```
#include <dtGame/message.h>
#include <dtGame/machineinfo.h>
#include <dtNetGM/export.h>
#include <dtNetGM/networkcomponent.h>
```

Classes

- class **ServerNetworkComponent**

Namespaces

- namespace **dtGame**
- namespace **dtNetGM**

*The **dtNetGM** (p. 9) namespace contains networking classes.*

Index

- Symbols -

- ~ClientConnectionListener
 - dtNetGM::ClientConnectionListener, 12
- ~ClientNetworkComponent
 - dtNetGM::ClientNetworkComponent, 14
- ~DataStreamPacket
 - dtNetGM::DataStreamPacket, 17
- ~MachineInfoMessage
 - dtNetGM::MachineInfoMessage, 21
- ~MessagePacket
 - dtNetGM::MessagePacket, 25
- ~NetworkBridge
 - dtNetGM::NetworkBridge, 28
- ~NetworkComponent
 - dtNetGM::NetworkComponent, 32
- ~ServerConnectionListener
 - dtNetGM::ServerConnectionListener, 38
- ~ServerNetworkComponent
 - dtNetGM::ServerNetworkComponent, 40

- A -

- AcceptClient
 - dtNetGM::ServerNetworkComponent, 40
- AddConnection
 - dtNetGM::NetworkComponent, 32
- ALL_CLIENTS
 - dtNetGM::NetworkComponent::DestinationType, 19
- ALL_NOT_CLIENTS
 - dtNetGM::NetworkComponent::DestinationType, 19

- B -

- BuildFromMessage
 - dtNetGM::MessagePacket, 25

- C -

- ClientConnectionListener
 - dtNetGM::ClientConnectionListener, 12
- clientconnectionlistener.cpp, 43
- clientconnectionlistener.h, 44
- ClientNetworkComponent
 - dtNetGM::ClientNetworkComponent, 14
- clientnetworkcomponent.cpp, 45
- clientnetworkcomponent.h, 46
- Create
 - dtNetGM::ClientConnectionListener, 12
 - dtNetGM::ServerConnectionListener, 38
- CreateDataStream
 - dtNetGM::NetworkComponent, 32
- CreateMessage
 - dtNetGM::NetworkComponent, 33

- D -

- DataStreamPacket
 - dtNetGM::DataStreamPacket, 17
- datastreampacket.cpp, 47
- datastreampacket.h, 48
- DEFAULT_NAME
 - dtNetGM::ClientNetworkComponent, 15
 - dtNetGM::ServerNetworkComponent, 41
- DESTINATION

- dtNetGM::NetworkComponent::DestinationType, 19
- Disconnect
 - dtNetGM::NetworkBridge, 28
 - dtNetGM::NetworkComponent, 33
- DispatchNetworkMessage
 - dtNetGM::NetworkComponent, 33
- DROP
 - dtNetGM::MessageActionCode, 23
- DT_NETGM_EXPORT
 - export.h, 50
- dtCore, 7
- dtGame, 8
- dtNetGM, 9
 - IMPLEMENT_ENUM, 9
 - IMPLEMENT_MANAGEMENT_LAYER, 9
- dtnetgm.h, 49
- dtNetGM::ClientConnectionListener, 11
 - ~ClientConnectionListener, 12
 - ClientConnectionListener, 12
 - Create, 12
 - onConnect, 12
 - onConnectFailure, 12
 - onDisconnect, 12
 - onError, 12
 - onExit, 12
 - onFailure, 12
 - onNewConn, 12
 - onReceive, 12
 - sptr, 12
 - wptr, 12
- dtNetGM::ClientNetworkComponent, 13
 - ~ClientNetworkComponent, 14
 - ClientNetworkComponent, 14
 - DEFAULT_NAME, 15
 - GetMachineInfo, 14
 - GetServer, 14
 - IsConnectedClient, 14
 - mConnectedClients, 15
 - OnDisconnect, 14
 - ProcessInfoClientConnected, 14
 - ProcessNetClientNotifyDisconnect, 14
 - ProcessNetServerAcceptConnection, 14
 - ProcessNetServerRejectConnection, 15
 - ProcessNetServerRejectMessage, 15
 - SendRequestConnectionMessage, 15
 - SetupClient, 15
- dtNetGM::DataStreamPacket, 16
 - ~DataStreamPacket, 17
 - DataStreamPacket, 17
 - GetDataStreamId, 17
 - GetDataStreamSize, 17
 - GetIndex, 17
 - GetPacketCount, 17
 - GetPayloadBuffer, 17
 - GetPayloadSize, 17
 - getSize, 17
 - ID, 18
 - MAX_PAYLOAD, 18
 - readPacket, 17
 - SetDataStreamId, 17

- SetDataStreamSize, 17
- SetIndex, 17
- SetPacketCount, 17
- SetPayloadSize, 17
- sptr, 17
- wptr, 17
- writePacket, 17
- dtNetGM::MachineInfoMessage, 20
 - ~MachineInfoMessage, 21
 - GetHostName, 21
 - GetIPAddress, 21
 - GetMachineInfo, 21
 - GetMachineInfoName, 21
 - GetPing, 21
 - GetTimeStamp, 21
 - GetUniqueid, 21
 - MachineInfoMessage, 21
 - SetHostName, 21
 - SetIPAddress, 21
 - SetMachineInfo, 21
 - SetMachineInfoName, 21
 - SetPing, 21
 - SetTimeStamp, 22
 - SetUniqueid, 22
- dtNetGM::MessageActionCode, 23
 - DROP, 23
 - REJECT, 23
 - SEND, 23
 - WAIT, 23
- dtNetGM::MessagePacket, 24
 - ~MessagePacket, 25
 - BuildFromMessage, 25
 - FillMessage, 25
 - GetDestinatonId, 25
 - GetMessageId, 25
 - GetMessageParameters, 25
 - getSize, 26
 - GetSourceId, 26
 - ID, 26
 - MessagePacket, 25
 - OverrideDestination, 26
 - readPacket, 26
 - sptr, 25
 - wptr, 25
 - writePacket, 26
- dtNetGM::NetworkBridge, 27
 - ~NetworkBridge, 28
 - Disconnect, 28
 - GetHostDescription, 28
 - GetMachineInfo, 28
 - IsConnectedClient, 28
 - IsNetworkConnected, 28
 - NetworkBridge, 28
 - OnConnect, 28
 - OnConnectFailure, 28
 - OnDisconnect, 28
 - OnError, 28
 - OnExit, 28
 - OnFailure, 29
 - OnNewConnection, 29
 - OnReceive, 29
 - OnTimeout, 29
 - SendDataStream, 29
 - SetClientConnected, 29
 - SetMachineInfo, 29
- dtNetGM::NetworkComponent, 30
 - ~NetworkComponent, 32
 - AddConnection, 32
 - CreateDataStream, 32
 - CreateMessage, 33
 - Disconnect, 33
 - DispatchNetworkMessage, 33
 - GetConnectedClients, 33
 - GetConnection, 33
 - GetHostName, 33
 - GetMachineInfo, 33
 - IsGNetInitialized, 33
 - IsReliable, 33
 - IsServer, 33
 - IsShuttingDown, 33
 - mBufferMutex, 36
 - mConnections, 36
 - mMutex, 36
 - mRateIn, 36
 - mRateOut, 36
 - mReliable, 36
 - NetworkComponent, 32
 - OnAddedToGM, 33
 - OnBeforeSendMessage, 33
 - OnConnect, 34
 - OnConnectFailure, 34
 - OnDisconnect, 34
 - OnError, 34
 - OnExit, 34
 - OnFailure, 34
 - OnNewConnection, 34
 - OnReceivedDataStream, 34
 - OnReceivedNetworkMessage, 35
 - OnRemovedFromGM, 35
 - OnTimeout, 35
 - ProcessInfoClientConnected, 35
 - ProcessMessage, 35
 - ProcessNetClientNotifyDisconnect, 35
 - ProcessNetClientRequestConnection, 35
 - ProcessNetServerAcceptConnection, 35
 - ProcessNetServerRejectConnection, 35
 - ProcessNetServerRejectMessage, 36
 - ProcessTickLocal, 36
 - RemoveConnection, 36
 - SendNetworkMessage, 36
 - SetConnectionParameters, 36
 - ShutdownNetwork, 36
- dtNetGM::NetworkComponent::DestinationType, 19
 - ALL_CLIENTS, 19
 - ALL_NOT_CLIENTS, 19
 - DESTINATION, 19
- dtNetGM::ServerConnectionListener, 37
 - ~ServerConnectionListener, 38
 - Create, 38
 - getNewConnectionParams, 38
 - mInRate, 38
 - mMutex, 38
 - mOutRate, 38
 - mReliable, 38
 - mServerNetworkcomponent, 38
 - onListenFailure, 38
 - onListenSuccess, 38
 - ServerConnectionListener, 37
 - sptr, 37
 - wptr, 37
- dtNetGM::ServerNetworkComponent, 39
 - ~ServerNetworkComponent, 40

- AcceptClient, 40
- DEFAULT_NAME, 41
- mAcceptClients, 41
- OnDisconnect, 40
- OnListenFailure, 40
- OnListenSuccess, 40
- ProcessNetClientNotifyDisconnect, 40
- ProcessNetClientRequestConnection, 40
- SendConnectedClientMessage, 40
- SendInfoClientConnectedMessage, 40
- ServerNetworkComponent, 40
- SetupServer, 41
- E -**
- export.h, 50
 - DT_NETGM_EXPORT, 50
- F -**
- FillMessage
 - dtNetGM::MessagePacket, 25
- G -**
- GetConnectedClients
 - dtNetGM::NetworkComponent, 33
- GetConnection
 - dtNetGM::NetworkComponent, 33
- GetDataStreamId
 - dtNetGM::DataStreamPacket, 17
- GetDataStreamSize
 - dtNetGM::DataStreamPacket, 17
- GetDestinatonId
 - dtNetGM::MessagePacket, 25
- GetHostDescription
 - dtNetGM::NetworkBridge, 28
- GetHostName
 - dtNetGM::MachineInfoMessage, 21
 - dtNetGM::NetworkComponent, 33
- GetIndex
 - dtNetGM::DataStreamPacket, 17
- GetIPAddress
 - dtNetGM::MachineInfoMessage, 21
- GetMachineInfo
 - dtNetGM::ClientNetworkComponent, 14
 - dtNetGM::MachineInfoMessage, 21
 - dtNetGM::NetworkBridge, 28
 - dtNetGM::NetworkComponent, 33
- GetMachineInfoName
 - dtNetGM::MachineInfoMessage, 21
- GetMessageld
 - dtNetGM::MessagePacket, 25
- GetMessageParameters
 - dtNetGM::MessagePacket, 25
- getNewConnectionParams
 - dtNetGM::ServerConnectionListener, 38
- GetPacketCount
 - dtNetGM::DataStreamPacket, 17
- GetPayloadBuffer
 - dtNetGM::DataStreamPacket, 17
- GetPayloadSize
 - dtNetGM::DataStreamPacket, 17
- GetPing
 - dtNetGM::MachineInfoMessage, 21
- GetServer
 - dtNetGM::ClientNetworkComponent, 14
- getSize
 - dtNetGM::DataStreamPacket, 17
 - dtNetGM::MessagePacket, 26
- GetSourceId
 - dtNetGM::MessagePacket, 26
- GetTimeStamp
 - dtNetGM::MachineInfoMessage, 21
- GetUniqueId
 - dtNetGM::MachineInfoMessage, 21
- I -**
- ID
 - dtNetGM::DataStreamPacket, 18
 - dtNetGM::MessagePacket, 26
- IMPLEMENT_ENUM
 - dtNetGM, 9
- IMPLEMENT_MANAGEMENT_LAYER
 - dtNetGM, 9
- inc/ Directory Reference, 5
- inc/dtNetGM/ Directory Reference, 3
- IsConnectedClient
 - dtNetGM::ClientNetworkComponent, 14
 - dtNetGM::NetworkBridge, 28
- IsGnlInitialized
 - dtNetGM::NetworkComponent, 33
- IsNetworkConnected
 - dtNetGM::NetworkBridge, 28
- IsReliable
 - dtNetGM::NetworkComponent, 33
- IsServer
 - dtNetGM::NetworkComponent, 33
- IsShuttingDown
 - dtNetGM::NetworkComponent, 33
- M -**
- mAcceptClients
 - dtNetGM::ServerNetworkComponent, 41
- MachineInfoMessage
 - dtNetGM::MachineInfoMessage, 21
- machineinfomessage.cpp, 51
- machineinfomessage.h, 52
- mainpage.h, 53
- MAX_PAYLOAD
 - dtNetGM::DataStreamPacket, 18
- mBufferMutex
 - dtNetGM::NetworkComponent, 36
- mConnectedClients
 - dtNetGM::ClientNetworkComponent, 15
- mConnections
 - dtNetGM::NetworkComponent, 36
- MessagePacket
 - dtNetGM::MessagePacket, 25
- messagepacket.cpp, 54
- messagepacket.h, 55
- mInRate
 - dtNetGM::ServerConnectionListener, 38
- mMutex
 - dtNetGM::NetworkComponent, 36
 - dtNetGM::ServerConnectionListener, 38
- mOutRate
 - dtNetGM::ServerConnectionListener, 38
- mRateIn
 - dtNetGM::NetworkComponent, 36
- mRateOut
 - dtNetGM::NetworkComponent, 36
- mReliable

dtNetGM::NetworkComponent, 36
 dtNetGM::ServerConnectionListener, 38
 mServerNetworkcomponent
 dtNetGM::ServerConnectionListener, 38

- N -

NetworkBridge
 dtNetGM::NetworkBridge, 28
 networkbridge.cpp, 56
 networkbridge.h, 57
 NetworkComponent
 dtNetGM::NetworkComponent, 32
 networkcomponent.cpp, 58
 networkcomponent.h, 59

- O -

OnAddedToGM
 dtNetGM::NetworkComponent, 33
 OnBeforeSendMessage
 dtNetGM::NetworkComponent, 33
 OnConnect
 dtNetGM::NetworkBridge, 28
 dtNetGM::NetworkComponent, 34
 onConnect
 dtNetGM::ClientConnectionListener, 12
 OnConnectFailure
 dtNetGM::NetworkBridge, 28
 dtNetGM::NetworkComponent, 34
 onConnectFailure
 dtNetGM::ClientConnectionListener, 12
 OnDisconnect
 dtNetGM::ClientNetworkComponent, 14
 dtNetGM::NetworkBridge, 28
 dtNetGM::NetworkComponent, 34
 dtNetGM::ServerNetworkComponent, 40
 onDisconnect
 dtNetGM::ClientConnectionListener, 12
 OnError
 dtNetGM::NetworkBridge, 28
 dtNetGM::NetworkComponent, 34
 onError
 dtNetGM::ClientConnectionListener, 12
 OnExit
 dtNetGM::NetworkBridge, 28
 dtNetGM::NetworkComponent, 34
 onExit
 dtNetGM::ClientConnectionListener, 12
 OnFailure
 dtNetGM::NetworkBridge, 29
 dtNetGM::NetworkComponent, 34
 onFailure
 dtNetGM::ClientConnectionListener, 12
 OnListenFailure
 dtNetGM::ServerNetworkComponent, 40
 onListenFailure
 dtNetGM::ServerConnectionListener, 38
 OnListenSuccess
 dtNetGM::ServerNetworkComponent, 40
 onListenSuccess
 dtNetGM::ServerConnectionListener, 38
 onNewConn
 dtNetGM::ClientConnectionListener, 12
 OnNewConnection
 dtNetGM::NetworkBridge, 29
 dtNetGM::NetworkComponent, 34

OnReceive
 dtNetGM::NetworkBridge, 29
 onReceive
 dtNetGM::ClientConnectionListener, 12
 OnReceivedDataStream
 dtNetGM::NetworkComponent, 34
 OnReceivedNetworkMessage
 dtNetGM::NetworkComponent, 35
 OnRemovedFromGM
 dtNetGM::NetworkComponent, 35
 OnTimeOut
 dtNetGM::NetworkComponent, 35
 OnTimeout
 dtNetGM::NetworkBridge, 29
 OverrideDestination
 dtNetGM::MessagePacket, 26

- P -

ProcessInfoClientConnected
 dtNetGM::ClientNetworkComponent, 14
 dtNetGM::NetworkComponent, 35
 ProcessMessage
 dtNetGM::NetworkComponent, 35
 ProcessNetClientNotifyDisconnect
 dtNetGM::ClientNetworkComponent, 14
 dtNetGM::NetworkComponent, 35
 dtNetGM::ServerNetworkComponent, 40
 ProcessNetClientRequestConnection
 dtNetGM::NetworkComponent, 35
 dtNetGM::ServerNetworkComponent, 40
 ProcessNetServerAcceptConnection
 dtNetGM::ClientNetworkComponent, 14
 dtNetGM::NetworkComponent, 35
 ProcessNetServerRejectConnection
 dtNetGM::ClientNetworkComponent, 15
 dtNetGM::NetworkComponent, 35
 ProcessNetServerRejectMessage
 dtNetGM::ClientNetworkComponent, 15
 dtNetGM::NetworkComponent, 36
 ProcessTickLocal
 dtNetGM::NetworkComponent, 36

- R -

readPacket
 dtNetGM::DataStreamPacket, 17
 dtNetGM::MessagePacket, 26
 REJECT
 dtNetGM::MessageActionCode, 23
 RemoveConnection
 dtNetGM::NetworkComponent, 36

- S -

SEND
 dtNetGM::MessageActionCode, 23
 SendConnectedClientMessage
 dtNetGM::ServerNetworkComponent, 40
 SendDataStream
 dtNetGM::NetworkBridge, 29
 SendInfoClientConnectedMessage
 dtNetGM::ServerNetworkComponent, 40
 SendNetworkMessage
 dtNetGM::NetworkComponent, 36
 SendRequestConnectionMessage
 dtNetGM::ClientNetworkComponent, 15
 ServerConnectionListener

dtNetGM::ServerConnectionListener, 37
 serverconnectionlistener.cpp, 60
 serverconnectionlistener.h, 61
 ServerNetworkComponent
 dtNetGM::ServerNetworkComponent, 40
 sernetworkcomponent.cpp, 62
 sernetworkcomponent.h, 63
 SetClientConnected
 dtNetGM::NetworkBridge, 29
 SetConnectionParameters
 dtNetGM::NetworkComponent, 36
 SetDataStreamId
 dtNetGM::DataStreamPacket, 17
 SetDataStreamSize
 dtNetGM::DataStreamPacket, 17
 SetHostName
 dtNetGM::MachineInfoMessage, 21
 SetIndex
 dtNetGM::DataStreamPacket, 17
 SetIPAddress
 dtNetGM::MachineInfoMessage, 21
 SetMachineInfo
 dtNetGM::MachineInfoMessage, 21
 dtNetGM::NetworkBridge, 29
 SetMachineInfoName
 dtNetGM::MachineInfoMessage, 21
 SetPacketCount
 dtNetGM::DataStreamPacket, 17
 SetPayloadSize
 dtNetGM::DataStreamPacket, 17
 SetPing
 dtNetGM::MachineInfoMessage, 21
 SetTimeStamp
 dtNetGM::MachineInfoMessage, 22
 SetUniqueld
 dtNetGM::MachineInfoMessage, 22
 SetupClient
 dtNetGM::ClientNetworkComponent, 15
 SetupServer
 dtNetGM::ServerNetworkComponent, 41
 ShutdownNetwork
 dtNetGM::NetworkComponent, 36
 sptr
 dtNetGM::ClientConnectionListener, 12
 dtNetGM::DataStreamPacket, 17
 dtNetGM::MessagePacket, 25
 dtNetGM::ServerConnectionListener, 37
 src/ Directory Reference, 6
 src/dtNetGM/ Directory Reference, 4
- W -
 WAIT
 dtNetGM::MessageActionCode, 23
 wptr
 dtNetGM::ClientConnectionListener, 12
 dtNetGM::DataStreamPacket, 17
 dtNetGM::MessagePacket, 25
 dtNetGM::ServerConnectionListener, 37
 writePacket
 dtNetGM::DataStreamPacket, 17
 dtNetGM::MessagePacket, 26