P2PChannelTransport Class

Wraps communication between the local and remote computer, handled by one or more **Connection** objects. It monitors these connections and acts as the intermediary for peer to peer data between **Transport** and the **Connection** object. Call **SendPacket** to send data, and connect to **SignalReadPacket** to read incoming data. **P2PTransport** creates this channel at the request of the **Session** object. See Transports, Channels, and Connections for more information.

These methods are not thread-safe. Any method that does not have a calling thread specified can be called from any thread.

Syntax

```
class P2PTransportChannel : public TransportChannelImp
                            public talk_base::MessageHandler
```

Methods

The following table lists the public methods of **P2PTransportChannel**.

Name	Description
~P2PTransportChannel	Destructor. Deletes references to all **PortAllocatorSession** objects that it holds.
bool **writable**()	Indicates whether the channel is currently writable. This method must be called from the worker thread.
Connection* **best_connection**()	Retrieves the best connection for this channel, as determined by preference and writability.
const std::vector<Connection*> **connections**()	Retrieves a list of **Connection** objects managed by this object.
int **GetError**()	Retrieves the last error that the channel had.
int **SendPacket**(const char **data*, size_t *len*)	Sends *data* of size *len* bytes to the connected computer over the best connection.
int **SetOption**(Socket::Option *opt*, int *value*)	Sets various sending optionsl. Currently, the only values are as follows: • **OPT_DONTFRAGMENT** 1 or 0 (true or false) Specifies whether data blocks sent to **Send** can be split up and sent in multiple packets. True means they cannot.
P2PTransportChannel(const std::string &*name*, const std::string &*session_type*,	Constructor. This object must be created in the worker thread. • *name* An arbitrary value used to identify the channel. This value is passed down and reused for the port objects that it creates. The file share sample creates this name in FileShareSession::RequestConnectedStream and

P2PTransport* *transport*, PortAllocator **allocator*);	FileShareSession::OnProxyAccept; the voice chat example in the VoiceChannel constructor. • *session_type* A unique session type associated with the session that created this object. See **Session:session_type** for more information. • *transport* The **Transport** object creating this object. • *allocator* The **PortAllocator** subclass created by the client.
std::string& **name**()	A unique name associated with this channel, created inside the code. This name is used to differentiate between multiple channels in the same **Transport**.
Transport* **GetTransport**()	Returns the **Transport** object that manages this channel.
void **Connect**()	Begins the process of attempting to make a connection to the other client.
void **OnChannelMessage**(const buzz::XmlElement* *msg*)	Received an incoming message. Typically the incoming message is a candidate message describing the remote candidate, which triggers **P2PTransportChannel** to create connections to the remote candidate.
void **OnMessage**(Message **pmsg*)	The listener for multithreaded requests.
void **OnSignalingReady**()	Called by **SocketManager** to indicate that the signaling thread is ready.
void **Reset**()	Resets the object to the same state it had after the constructor was called. This method must be called from the worker thread.

Signals

SignalChannelMessage<TransportChannelImpl*, buzz::XmlElement*>
Sends an outgoing message through the **Session** object.

SignalRequestSignaling
Sent to indicate that the signaling thread is live.

SignalReadPacket< TransportChannel *, const char *, size_t >
Sent when the socket has a packet of data from the other computer.

SignalConnectionMonitor<P2PTransportChannel*>
Sent with a pointer to itself when the connection state changes.

SignalReadibleState<TransportChannel* >
Sent when the channel is readible.

SignalWritableState<TransportChannel*>
Sent when the channel is writable.

SignalConnectionMonitor< P2PTransportChannel * >
Sent when the list of available connections is being evaluated. You should not need to listen for this signal.

SignalRouteChange <TransportChannel*, const talk_base::SocketAddress&>
Sent when there is a change in the way that packets are being routed. The address indicates the address of the first hop in the new route, if this is known. If this cannot be determined or is not well-defined, then the channel may give an address of 0.

Attributes: public

Declaration file: talk/p2p/base/p2ptransportchannel.h

Last updated March 23, 2012.