

# Authentication and Connection Monitoring Messages

- [ENCODING\\_REQUEST](#)
- [ENCODING\\_RESPONSE](#)
- [LOGON\\_REQUEST](#)
- [LOGON\\_RESPONSE](#)
- [LOGOFF](#)
- [HEARTBEAT](#)

## ENCODING\_REQUEST [s\_EncodingRequest structure] Client >> Server

**Requirements:** Not required for Servers. Required for Clients if the Client needs to discover the encoding the Server uses.

The [ENCODING\\_REQUEST](#) message is a message requesting to change the DTC encoding for messages.

For the procedure to work with this message, refer to [Encoding Request Sequence](#).

Hexadecimal representation for encoding request message using binary encoding which is requesting protocol buffer encoding: 0x10000600 0x08000000 0x04000000 0x44544300.

Hexadecimal representation for encoding request message using binary encoding which is requesting JSON encoding: 0x10000600 0x08000000 0x02000000 0x44544300.

Field Name	Field Description
<a href="#">[unsigned_int16]</a> Size	The standard message size field. Automatically set by constructor.

<a href="#">[unsigned_int16]</a> <b>Type</b>	<p>The standard message type field. Automatically set by constructor.</p> <p>To determine the field number for JSON, refer to this message type constant in the <a href="#">DTCProtocol.h</a> file.</p>
<a href="#">[int32]</a> <b>ProtocolVersion</b>	<p>The protocol version supported by the Client. Automatically set by constructor.</p>
<a href="#">[EncodingEnum]</a> <b>Encoding</b>	<p>The DTC message encoding the Client is requesting the Server to use.</p>
<a href="#">[char]</a> <b>ProtocolType</b>	<p>The <b>ProtocolType</b> field needs to be set to the text string "DTC".</p> <p>This field is automatically set with the binary encoding data structures.</p> <p>This field is used for the Server to know that it is communicating with a DTC compliant Client.</p>

## **ENCODING\_RESPONSE [s\_EncodingResponse structure] Server >> Client**

**Requirements:** Required for Servers. Required for Clients if the Client needs to discover the encoding the Server uses.

The [ENCODING\\_RESPONSE](#) is a message from the Server to the Client, telling the Client what message encoding it must use to communicate with the Server.

For the procedure to work with this message, refer to [Encoding Request Sequence](#).

<b>Field Name</b>	<b>Field Description</b>
<a href="#">[unsigned_int16]</a> <b>Size</b>	<p>The standard message size field. Automatically set by constructor.</p>

<a href="#">[unsigned int16]</a> <b>Type</b>	<p>The standard message type field. Automatically set by constructor.</p> <p>To determine the field number for JSON, refer to this message type constant in the <a href="#">DTCProtocol.h</a> file.</p>
<a href="#">[int32]</a> <b>ProtocolVersion</b>	<p>This field is automatically set by the constructor.</p>
<a href="#">[EncodingEnum]</a> <b>Encoding</b>	<p>The DTC message encoding to be used.</p> <p>This value may be different from the requested DTC encoding if the Server does not support the requested encoding from the Client.</p>
<a href="#">[char]</a> <b>ProtocolType</b>	<p>The <b>ProtocolType</b> field needs to be set to the text string "DTC".</p> <p>This field is automatically set with the binary encoding data structures.</p> <p>This field is used for the Client to know that it is communicating with a DTC compliant Server.</p>

## **LOGON\_REQUEST [s\_LogonRequest structure] Client >> Server**

The [LOGON\\_REQUEST](#) message is sent from the Client to the Server requesting to logon to the Server.

This is the very first message the Client sends to the Server before being allowed to send any other message other than the [ENCODING\\_REQUEST](#).

<b>Field Name</b>	<b>Field Description</b>
<a href="#">[unsigned int16]</a> <b>Size</b>	<p>The standard message size field. Automatically set by constructor.</p>
<a href="#">[unsigned int16]</a> <b>Type</b>	<p>The standard message type field. Automatically set by constructor.</p> <p>To determine the field number for JSON, refer to this message type constant in the <a href="#">DTCProtocol.h</a> file.</p>

<a href="#">[int32]</a> <b>ProtocolVersion</b>	The protocol version supported by the Client. Automatically set by constructor.
<a href="#">[char]</a> <b>Username</b>	Optional username for the server to authenticate the Client.
<a href="#">[char]</a> <b>Password</b>	Optional password for the server to authenticate the Client.
<a href="#">[char]</a> <b>GeneralTextData</b>	Optional general-purpose text string. For example, this could be used to pass a license key that the Server may require.
<a href="#">[int32]</a> <b>Integer_1</b>	Optional. General-purpose integer.
<a href="#">[int32]</a> <b>Integer_2</b>	Optional. General-purpose integer.
<a href="#">[int32]</a> <b>HeartbeatIntervallInSeconds</b>	<p>The interval in seconds that each side, the Client and the Server, needs to use to send <a href="#">HEARTBEAT</a> messages to the other side.</p> <p>This should be a value from anywhere from 5 to 60 seconds.</p> <p>This field is required.</p>

<a href="#">[char]</a> <b>TradeAccount</b>	<p>This is an optional field and this should only be set to a Trade Account identifier if that is required to logon by the Server. this would only be implemented in rare cases. Usually this would be the case if the logon is bound to a particular Trade Account and not changeable after the log in.</p> <p>The server is still required to implement the <a href="#">TRADE ACCOUNTS REQUEST</a> and <a href="#">TRADE ACCOUNT RESPONSE</a> messages.</p>
<a href="#">[char]</a> <b>HardwareIdentifier</b>	<p>Optional: This is the computer hardware identifier. The intention of this is that this will be implemented by the Client program developer on a case-by-case basis for specific Data/Trading service providers. It will be a reasonable implementation to uniquely identify a system and will not be publicly disclosed. It will never contain personally identifiable information.</p>
<a href="#">[char]</a> <b>ClientName</b>	<p>The Client name. This is a free-form text string.</p>
<a href="#">[int32]</a> <b>MarketDataTransmissionInterval</b>	<p>This is an optional field to be used by the Server which specifies in milliseconds, the delay with transmitting market data to the Client.</p> <p>For reasons of efficiency, the server may buffer data over this timeframe, and send data after this time frame expires.</p>

## **LOGON\_RESPONSE [s\_LogonResponse structure] Server >> Client**

This is a response message indicating either success or an error logging on to the Server.

<b>Field Name</b>	<b>Field Description</b>

<a href="#">[unsigned int16]</a> <b>Size</b>	The standard n constructor.
<a href="#">[unsigned int16]</a> <b>Type</b>	The standard n constructor.  To determine th message type c
<a href="#">[int32]</a> <b>ProtocolVersion</b>	This is automati
<a href="#">[LogonStatusEnum]</a> <b>Result</b>	<p>This can be set</p> <ul style="list-style-type: none"> <li>• LOGON_</li> <li>• LOGON_</li> <li>• LOGON_</li> <li>• LOGON_</li> </ul> <p><b>LOGON_ERRO</b> has been an eri try to reconnect</p> <p>The Server <b>LOGON_RECO</b> the Client to r address. The r <b>ReconnectAdd</b> connections to a</p>
<a href="#">[char]</a> <b>ResultText</b>	Optional freefor a successful c display this text
<a href="#">[char]</a> <b>ReconnectAddress</b>	Server address/ reconnect to. F Only used <b>LOGON_RECO</b>

<a href="#">[int32]</a> <b>Integer_1</b>	Optional. Gene communicate to
<a href="#">[char]</a> <b>ServerName</b>	Optional free-fo  It is recommen descriptive text  The length of th length strings ar
<a href="#">[unsigned int8]</a> <b>MarketDepthUpdatesBestBidAndAsk</b>	Set this to 1 to sending market ask updates. T update the best  Some Clients w prices from mar
<a href="#">[unsigned int8]</a> <b>TradingIsSupported</b>	Set this to 1 to Otherwise, the messages.
<a href="#">[unsigned int8]</a> <b>OCOOrdersSupported</b>	Set this to 1 to orders.
<a href="#">[unsigned int8]</a> <b>OrderCancelReplaceSupported</b>	Set this to ( <a href="#">CANCEL REPL</a>

<p><a href="#">[char]</a> <b>SymbolExchangeDelimiter</b></p>	<p>Some Clients use the SymbolExchange fields to be using the Exchange field to have a SymbolExchange field as a SymbolExchange delimiter for the SymbolExchange and the Exchange field.</p> <p>It is recommended that the Client will then use the following:</p> <ul style="list-style-type: none"> <li>• Symbol-Exchange</li> <li>• Symbol.Exchange</li> </ul> <p>If this field is used by the Client that the messages are not used.</p> <p>Even if the SymbolExchange text is used, the Client can use the <b>Exchange</b> field to combine the SymbolExchange and the Exchange field in the Definition response.</p> <p>When a Client uses the SymbolExchange field is set, then the Client is set to use the Symbol and Exchange fields separately.</p>
<p><a href="#">[unsigned int8]</a> <b>SecurityDefinitionsSupported</b></p>	<p>Set to 1 if the Client supports the SecurityDefinitionsSupported messages.</p>
<p><a href="#">[unsigned int8]</a> <b>HistoricalPriceDataSupported</b></p>	<p>Set this to 1 if the Client supports the <a href="#">HISTORICAL_PRICE_DATA</a> field.</p>
<p><a href="#">[unsigned int8]</a> <b>ResubscribeWhenMarketDataFeedAvailable</b></p>	<p>Set this to 1, if the Client supports the <a href="#">MARKET_DATA_FEED</a> data feed is resubscribed and market data is previously tracked.</p>



<p><a href="#">[unsigned int8]</a> <b>MarketDepthsSupported</b></p>	<p>Set this to <a href="#">MARKET_DEP</a></p> <p>The default is 0.</p>
<p><a href="#">[unsigned int8]</a> <b>OneHistoricalPriceDataRequestPerConnection</b></p>	<p>The server <b>OneHistoricalF</b> to 1 in the <a href="#">LOC</a> that it only will i per network con</p> <p>After the first re connection will time by the Ser of historical pri implementation compression is</p>
<p><a href="#">[unsigned int8]</a> <b>BracketOrdersSupported</b></p>	<p>Set this to 1 to orders.</p>
<p><a href="#">[unsigned int8]</a> <b>UseIntegerPriceOrderMessages</b></p>	<p>With the intege August 2020, th</p>
<p><a href="#">[unsigned int8]</a> <b>UsesMultiplePositionsPerSymbolAndTradeAccount</b></p>	<p>If the Server ca for a specific Sy to <b>UsesMultipleP</b> to 1.</p> <p>When the serv <b>PositionIdentif</b> message to the</p> <p>When the Clier knows that it ca than one Trade Trade Acc <a href="#">POSITION_UPI</a> handle this appi</p>

[\[unsigned int8\]](#) **MarketDataSupported**

Set this to  
[MARKET\\_DATA](#)

The default is 1.

## LOGOFF [[s\\_Logoff structure](#)] Server >> Client and Client >> Server

A **LOGOFF** is a message which can be sent either by the Client or the Server to the other side. It indicates that the Client or the Server is logging off and going to be closing the connection.

When one side receives this message, it should expect the connection will be closed. It should not be expected that any messages will follow the **LOGOFF** message, and it should close the network connection and consider it finished. The side receiving this message can send a **LOGOFF** message to the other side before closing the connection.

Field Name	Field Description
<a href="#">[unsigned int16]</a> <b>Size</b>	The standard message size field. Automatically set by constructor.
<a href="#">[unsigned int16]</a> <b>Type</b>	The standard message type field. Automatically set by constructor.  To determine the field number for JSON, refer to this message type constant in the <a href="#">DTCPProtocol.h</a> file.
<a href="#">[char]</a> <b>Reason</b>	<b>Reason</b> is a character string indicating the reason for the log off from either the Client or the Server.
<a href="#">[unsigned int8]</a> <b>DoNotReconnect</b>	When <b>DoNotReconnect</b> is set to a 1, this indicates to the other side that a reconnect to the opposite side should not occur automatically.

## HEARTBEAT [[s\\_Heartbeat structure](#)] Server >> Client and Client >> Server

Both the Client and the Server need to send to the other side a heartbeat at the interval specified by the **HeartbeatIntervalInSeconds** member in the [LOGON\\_REQUEST](#).

There are no required member fields to set in this message. The purpose of the **HEARTBEAT** message is so that the Client or the Server can determine whether the other side is still connected.

It is recommended that if there is a loss of **HEARTBEAT** messages from the other side, for twice the

amount of the **HeartbeatIntervalInSeconds** time that it is safe to assume that the other side is no longer present and the network socket should be then gracefully closed.

The Server may choose to send a heartbeat message every second to the Client. In this particular case, it is recommended the Client use a minimum time of about 5 to 10 seconds without a heartbeat to determine the loss of the connection rather than the standard of twice the amount of the heartbeat time interval.

Field Name	Field Description
<a href="#">[unsigned int16]</a> <b>Size</b>	The standard message size field. Automatically set by constructor.
<a href="#">[unsigned int16]</a> <b>Type</b>	<p>The standard message type field. Automatically set by constructor.</p> <p>To determine the field number for JSON, refer to this message type constant in the <a href="#">DTCTProtocol.h</a> file.</p>
<a href="#">[unsigned int32]</a> <b>NumDroppedMessages</b> (Optional)	<p>The server can optionally set this to indicate the number of messages that were not sent through to the Client because of a buffer overflow on the server side because the Client was not processing the data fast enough or some other network issue.</p> <p>The Server should only drop high-frequency market data messages. In no case should a server ever drop trading related messages.</p>
<a href="#">[t_DateTime]</a> <b>CurrentDateTime</b> (Optional)	This Date-Time value can be optionally set by the Client/Server when sending this message.

---

\*Last modified Tuesday, 22nd November, 2022.