FUJ00098221

RiposteCreatePriorityMessage

The **RiposteCreatePriorityMessage** function creates a priority message.

unsigned long RiposteCreatePriorityMessage (
unsigned long *dwGroupId*,
unsigned char **szMessage,*
unsigned long *dwMessageFlags,*
unsigned long *dwTranAction,*
TRAN_HANDLE **phTran,*
unsigned long **pdwNum,*
unsigned long *dwExpiry,*
unsigned long *dwDisconnectInterval,*
unsigned long *dwIntervalOperator*
);

Parameters

dwGroupId
The group identifier of the group in which the message will be created. If the host message server is not a correspondence server, specify zero to use the group of the host message server.

szMessage
A null-terminated string containing a list of message attributes in attribute grammar format (see Attribute Grammar). This list must not contain any attributes which use reserved attribute names. See Riposte32 Constants for a list of reserved attribute names. To avoid potential conflicts with reserved attribute names, it is recommended that applications encapsulate their attributes within an attribute named **RIPOSTE_DATA_ATTRIBUTE**, which is explicitly not reserved.

dwMessageFlags
A bitmask of message creation flags. If no message creation flags are specified, the message will be created as a normal user message (see Remarks below). Message creation flags can be combined using a bitwise OR. See RiposteCreateMessageEx for a list of currently defined message creation flags.

dwTranAction
A transaction flag which specifies transaction management. See RiposteCreateMessageEx for a list of defined transaction flags.

phTran
A pointer to a transaction handle which was created by RiposteStartTransaction (or a previous call to **RiposteCreateMessageEx**), or a pointer to NULL, depending on the value of *dwTranAction*.

pdwNum
A pointer to a longword which will receive the message sequence number of the created message. The sequence number will be one greater than the sequence number of the last message created by the host message server for this group. This parameter is optional and may be NULL.

dwExpiry
Specifies the message expiry of the message in days. The message expiry determines when the message will expire and become subject to archiving. The message will expire at the end of the day (in UTC time) *dwExpiry* days past the current date (in UTC time). If *dwExpiry* is zero, then the message expiry will be set according to the configuration parameter DefaultMessageExpiry. Note that if *dwExpiry* is less than the configuration parameter MinMessageExpiry, then the message expiry will be set to MinMessageExpiry, and if *dwExpiry* is greater than the configuration parameter MaxMessageExpiry, then the message expiry will be set to MaxMessageExpiry.

dwDisconnectInterval

FUJ00098221

Specifies the disconnect interval in milliseconds. This value will be applied to the current value of the disconnect time on any non-permanent connections through which the message is forwarded based on the value of *dwIntervalOperator*. See Remarks for more details.

dwIntervalOperator
Specifies an interval operator which controls how the value specified by *dwDisconnectInterval* will be combined with the current value of the disconnect time on any non-permanent connections through which the message is forwarded (see Remarks for more details). A list of defined interval operators is shown in the table below:

RIPOSTE_NULL_INTERVAL_OPERATOR	The current disconnect time will not be modified. *dwDisconnectInterval* is ignored in this case.
RIPOSTE_ASSIGNMENT_INTERVAL_OPERATOR	The current disconnect time will be replaced by the current time plus *dwDisconnectInterval*
RIPOSTE_MIN_INTERVAL_OPERATOR	The current disconnect time will be replaced by the current time plus *dwDisconnectInterval* provided this value is less than the current disconnect time.
RIPOSTE_MAX_INTERVAL_OPERATOR	The current disconnect time will be replaced by the current time plus *dwDisconnectInterval* provided this value is greater than the current disconnect time.

Return Values

The function returns 0 if successful, otherwise it returns an error code which may be an RPC error code or a Windows NT error code. Use RiposteErrorString to get a text description of the error. For a list of the most common return values, see RiposteCreateMessageEx:

Remarks

RiposteCreatePriorityMessage creates a priority message. See RiposteCreateMessageEx for general information on creating messages.

RiposteCreatePriorityMessage functions identically to RiposteCreateMessageEx except that the created message is a priority message, which will cause the message to be forwarded to all message servers in the specified group, including message servers which must be reached via non-permanent connections. The parameters *dwDisconnectInterval* and *dwIntervalOperator* affect how such non-permanent connections will be managed.

The forwarding process for a priority message works similarly to that of normal messages, except that if a

FUJ00098221

neighboring message server is only reachable via a non-permanent connection, that connection will be opened if it is not currently connected so that the priority message can be forwarded to the neighbor. Note that this applies not only to the host message server, but to any message server which receives the forwarded message. Once a non-permanent connection has been established, it will be kept open until the disconnect time for the connection has been reached (and the neighbors have completed synchronization - see RiposteDefineNode for more details). The disconnect time for a non-permanent connection is initially set to the current time, and then modified each time a priority message is forwarded across the connection according to the *dwDisconnectInterval* and *dwIntervalOperator* parameters.

For example, if *dwDisconnectInterval* is set to 30000 and *dwIntervalOperator* is set to **RIPOSTE_MIN_INTERVAL_OPERATOR**, any non-permanent connection through which the message is forwarded will be kept open for at least 30 seconds (unless the disconnect time is subsequently modified by another priority message). In a client/server message exchange, this allows the client to ensure that any non-permanent connections are kept up while it is waiting for a response from the server, thus avoiding having to re-open the connection when the server sends its response. In this scenario the priority message created when the server responds might have *dwDisconnectInterval* set to 0 and *dwIntervalOperator* set to **RIPOSTE_ASSIGNMENT_INTERVAL_OPERATOR**, which would cause any non-permanent connections to be closed once the message was forwarded and the neighbors were synchronized.

Note that as with non-priority messages, the message will not be forwarded until the transaction in which it has been created has been committed - see RiposteCreateMessageEx for more details.

To create a non-priority message, use RiposteCreateMessageEx.

If a transaction is created by using the **RIPOSTE_TRAN_START** transaction flag, the calling application must either subsequently call **RiposteCreateMessage** using the **RIPOSTE_TRAN_END** transaction flag, or call RiposteEndTransaction or RiposteUndoTransaction to explicitly commit or abort the transaction.

Note that TRAN_HANDLE is an RPC context handle. *phTran* must specify either a pointer to a valid transaction handle, or a pointer to NULL. Even if *dwTranAction* is set to **RIPOSTE_TRAN_ATOMIC**, *phTran* cannot be NULL, it must specify a pointer to NULL.

See Also

RiposteCreateMessageEx, RiposteStartTransaction, RiposteEndTransaction, RiposteUndoTransaction, RiposteDefineNode

Riposte32 Constants, Riposte32 Functions, Riposte32 Messaging Functions