Writing Tools C-Callable Functions

Providing a set of C-callable functions allows the Writing Tools API to present a consistent programming interface across platforms. These functions fall into three categories: sending a command message with wtcClSend( ) and wtcTlSend( ), sending a query message with wtqClSend( ) and wtqTlSend( ), and receiving a message with wtClReceive( ) and wtTlReceive( ).

wtcClSend

A writing tool client uses the wtcClSend( ) function to send a writing tool command message to the writing tool. No response is expected. All messages sent using this command should have a WTC_ prefix.

Prototype

```
WTSTATUS wtcClSend (
    WTCOMM      comm,
    WTMSGID     inmsgid,
    WTBUFP          inmsg,
    WTSIZE      inmsgsize,
    WTBUFP          inbuf,
    WTSIZE      inbufsize );
```

Parameters

comm

Platform-specific writing tool communications information.

inmsgid

The Writing Tools API message ID.

inmsg

A pointer to the buffer containing the structure corresponding to the message ID.

inmsgsize

The size in bytes of the input message structure.

inbuf

A pointer to the buffer containing any variable-length data related to the message.

inbufsize

The size in bytes of the input message data buffer.

Return Values

Returns WTS_OK if the message was sent. If there was a problem in sending the message, the appropriate member of the WTSTATUS type will be returned.

Example

The following example sends the WTC_INIT message to a writing tool.

```
WTCINIT initmsg;

initmsg.msgid = WTC_INIT;
initmsg.action = WTM_NORMAL;
initmsg.version = WTAPI_VERSION;

return  wtcClSend( comm,  WTC_INIT,  (WTBUFP)&initmsg, sizeof(WTCINIT), 0, 0 );
```

wtClReceive

Called by the Writing Tools API message layer when a writing tools message is sent to the writing tool client.

Prototype

```
WTSTATUS wtClReceive (
    WTCOMM      comm,
    WTMSGID     inmsgid,
    WTBUFP          inmsg,
    WTBUFP          inbuf,
    WTBUFP          rtmsg,
    WTBUFP          rtbuf,
    WTSIZEP         rtbufsizep );
```

Parameters

comm

Platform-specific writing tool communications information.

inmsgid

The Writing Tools API message ID.

inmsg

A pointer to the buffer containing the message structure.

inbuf

A pointer to the buffer containing the message variable-length data.

rtmsg

A pointer to the buffer to contain the return message structure.

rtbuf

A pointer to the buffer to contain the return message variable-length data.

rtbufsizep

A pointer to the buffer to hold the actual size of the return message variable-length data. This should be set to zero if no data is returned in *rtbuf*.

Return Values

Returns WTS_OK if the message was sent. If there was a problem in sending the message, the appropriate member of the WTSTATUS type will be returned.

Example

In the following code fragment, a wtClReceive( ) function receives a WTQ_INFOBLOCK message.

```
WTSTATUS wtClReceive( WTCOMM comm, WTMSGID inmsg, WTBUFP inmsg,
   WTBUFP inbuf, WTBUFP rtmsg, WTBUFP rtbuf, WTSIZEP rtbufsizep )
{
   WTSTATUS status;

   *rtbufsizep = 0;    /* set to zero */

   switch (inmsg)
   {

      case WTQ_INFOBLOCK:
         status = wtqInfoBlockReceive( inmsg, rtmsg, rtbuf );
         *rtbufsizep = ((WTRINFOBLOCKP)rtmsg)->winsize;
         break;
   }
```

```
    return status;
}
```

wtcTlSend

A writing tool uses the wtcTlSend( ) function to send a writing tool command message to the writing tool client. No response is expected. All messages sent using this command should have a WTC_ prefix.

Prototype

```
WTSTATUS wtcTlSend (
    WTCOMM      comm,
    WTMSGID     inmsgid,
    WTBUFP          inmsg,
    WTSIZE      inmsgsize,
    WTBUFP          inbuf,
    WTSIZE      inbufsize );
```

Parameters

comm

Platform-specific writing tool communications information.

inmsgid

The Writing Tools API message ID.

inmsg

A pointer to the buffer containing the structure corresponding to the message ID.

inmsgsize

The size in bytes of the input message structure.

inbuf

A pointer to the buffer containing any variable-length data related to the message.

inbufsize

The size in bytes of the input message data buffer.

Return Values

Returns WTS_OK if the message was sent. If there was a problem in sending the message, the appropriate member of the WTSTATUS type will be returned.

Example

The following example sends the WTC_RQINIT message to a writing tool.

```
WTCRQINIT initmsg;

initmsg.msgid = WTC_RQINIT;
initmsg.newdoc = WT_TRUE;
initmsg.size = 0;

return  wtcTlSend( comm,  WTC_INIT,  (WTBUFP)&initmsg, sizeof(WTCINIT), 0, 0 );
```

wtqClSend

A writing tool client uses the wtqClSend( ) function to send a writing tool query message to the writing tool. An appropriate response is expected from this command. The writing tool fills in the corresponding reply message buffers that have been provided. The message is then returned by the API. *Applications do not call a function to send a reply*. All messages sent using this command should have a WTQ_ prefix and all replies should have a WTR_ prefix.

Prototype

```
WTSTATUS wtqClSend (
    WTCOMM      comm,
    WTMSGID     inmsgid,
    WTBUFP          inmsg,
```

```
WTSIZE      inmsgsize,
WTBUFP        inbuf,
WTSIZE      inbufsize,
WTMSGID     rtmsgid,
WTBUFP        rtmsg,
WTSIZE      rtmsgsize,
WTBUFP        rtbuf,
WTSIZE      rtbufsize );
```

Parameters

comm	Platform-specific writing tool communications information.
inmsgid	The Writing Tools API message ID.
inmsg	A pointer to the buffer containing the structure corresponding to the input message ID.
inmsgsize	The size in bytes of the input message structure.
inbuf	A pointer to the buffer containing any variable-length data related to the input message.
inbufsize	The size in bytes of the input message data buffer.
rtmsgid	The Writing Tools API message ID of the return message.
rtmsg	A pointer to the buffer containing the structure corresponding to the return message ID.
rtmsgsize	The size in bytes of the return message structure.

rtbuf

A pointer to the buffer containing any variable-length data related to the return message.

rtbufsize

The size in bytes of the return message data buffer.

Return Values

Returns WTS_OK if the message was sent. If there was a problem in sending the message, the appropriate member of the WTSTATUS type will be returned.

wtqTlSend

A writing tool uses the wtqTlSend( ) function to send a writing tool query message to the writing tool client. An appropriate response is expected from this command. The writing tool client fills in the corresponding reply message buffers that have been provided. The message is then returned by the API. *Applications do not call a function to send a reply*. All messages sent using this command should have a WTQ_ prefix and all replies should have a WTR_ prefix.

Prototype

```
WTSTATUS wtqTlSend (
WTCOMM      comm,
WTMSGID     inmsgid,
WTBUFP          inmsg,
WTSIZE      inmsgsize,
WTBUFP          inbuf,
WTSIZE      inbufsize,
WTMSGID     rtmsgid,
WTBUFP          rtmsg,
WTSIZE      rtmsgsize,
WTBUFP          rtbuf,
WTSIZE      rtbufsize );
```

Parameters

comm

Platform-specific writing tool communications information.

inmsgid

The Writing Tools API message ID.

inmsg

A pointer to the buffer containing the structure corresponding to the input message ID.

inmsgsize

The size in bytes of the input message structure.

inbuf

A pointer to the buffer containing any variable-length data related to the input message.

inbufsize

The size in bytes of the input message data buffer.

rtmsgid

The Writing Tools API message ID of the return message.

rtmsg

A pointer to the buffer containing the structure corresponding to the return message ID.

rtmsgsize

The size in bytes of the return message structure.

rtbuf

A pointer to the buffer containing any variable-length data related to the return message.

rtbufsize

The size in bytes of the return message data buffer.

Return Values

Returns WTS_OK if the message was sent. If there was a problem in sending the message, the appropriate member of the WTSTATUS type will be returned.

Example

The following example requests the current paragraph from the writing tool client.

```
WTQTEXTBLOCK qtbmsg;
WTRTEXTBLOCK rtbmsg;
WTBUF text[500];
WTSTATUS status;

qtbmsg.msgid = WTQ_TEXTBLOCK;
qtbmsg.fromtext = WTU_PARAGRAPH;
qtbmsg.frompos = WTP_REL;
qtbmsg.fromloc = 0;
qtbmsg.totext = WTU_PARAGRAPH;
qtbmsg.topos = WTP_REL;
qtbmsg.toloc = 0;
qtbmsg.qtype = WTB_INIT;
qtbmsg.containers = WT_FALSE;
qtbmsg.size = 500;

status = wtqTlSend( comm, WTQ_TEXTBLOCK, (WTBUFP)&qtbmsg,
    sizeof(WTQTEXTBLOCK), 0, 0, WTR_TEXTBLOCK,
    (WTBUFP)&rtbmsg, sizeof(WTRTEXTBLOCK), &text, 500 );
```

wtTlReceive

Called by the Writing Tools API message layer when a writing tool message is sent to the writing tool.

Prototype

```
WTSTATUS wtTlReceive (
    WTCOMM      comm,
    WTMSGID     inmsgid,
    WTBUFP          inmsg,
    WTBUFP          inbuf,
    WTBUFP          rtmsg,
    WTBUFP          rtbuf,
    WTSIZEP         rtbufsizep );
```

Parameters

comm

Platform-specific writing tool communications information.

inmsgid

The Writing Tools API message ID.

inmsg

A pointer to the buffer containing the message structure.

inbuf

A pointer to the buffer containing the message variable-length data.

rtmsg

A pointer to the buffer to hold the return message structure.

rtbuf

A pointer to the buffer to hold the return message variable-length data.

rtbufsizep

A pointer to the buffer to hold the actual size of the return message variable-length data. This should be set to zero if no data is returned in *rtbuf*.

Return Values

Returns WTS_OK if the message was sent. If there was a problem in sending the message, the appropriate member of the WTSTATUS type will be returned.

Enumerated Types

The enumerated types in the WTAPI allow message parameters to be set to a specific value within a set of values. Each type used in the API is

described below. These enumerated types define the parameters used in both the C-callable interface and the API message structures.

WTMSGID

An enumerated list of all writing tool messages. These messages, their use, and associated message structures are covered individually in *WTAPI Messages* later in this section .

Members
WTC_NOMESSAGE
Message zero is not an API message.
WTC_RQINIT
Request API initialization.
WTC_INIT
Initialize API communication.
WTQ_INFOBLOCK
Request the client information block.
WTR_INFOBLOCK
Return the infomation block.
WTQ_UNITINFO
Request text unit infomation.
WTR_UNITINFO
Return unit infomation.
WTQ_READTOOLDATA
Request a tool data area.
WTR_READTOOLDATA
Return a tool data area.

WTQ_WRITETOOLDATA

Write a tool data area.

WTR_WRITETOOLDATA

Acknowledge tool data area write.

WTQ_TEXTBLOCK

Request a text block.

WTQ_NEXTTEXTBLOCK

Request the next text block.

WTR_TEXTBLOCK

Return a text block.

WTQ_READTEXTOBJECT

Request a text object.

WTR_READTEXTOBJECT

Return a text object.

WTQ_WRITETEXTOBJECT

Write a text object.

WTR_WRITETEXTOBJECT

Acknowledge text object written.

WTQ_GOTO

Go to an API position.

WTR_GOTO Acknowledge new position.

WTQ_HILITE Highlight an area.

WTR_HILITE	Acknowledge highlight.
WTQ_DEHILITE	Dehighlight an area.
WTR_DEHILITE	Acknowledge dehighlight.
WTQ_REPLACE	Replace text in a text block.
WTQ_UNDOREPLACE	Undo the last replace.
WTR_REPLACE	Acknowledge replacement.
WTC_RQCLACTIVE	Request client activation.
WTC_CLACTIVE	Client was activated.
WTC_CLINACTIVE	Client was deactivated.
WTC_TLINACTIVE	Tool was deactivated.
WTQ_TLACTIVE	Tool was reactivated.
WTR_TLACTIVE	Update tool on text query status.
WTC_RQTLTERM	Client notifies tool of termination request.
WTC_TLTERM	Tool terminates session.

WTSTATUS

A list of completion values returned by Writing Tools API functions and used in API messages.

Members

WTS_OK

OK status.

WTS_NOMEMORY

Not enough memory.

WTS_COMERROR

Error in WTAPI communication.

WTS_BADREQUEST

Bad WTAPI request message.

WTS_CLIENTERROR

Client internal error.

WTS_USER

User intervention.

WTS_QUITAPI

Termination requested.

WTS_TOOLBUSY

Writing tool is busy.

WTS_NOTSUPPORTED

API request is not supported.

WTBOOL

Used for Boolean conditions.

Members

WT_FALSE

False condition or flag.

WT_TRUE

True condition or flag.

WTPOS

Used to direct the positioning of the Writing Tools API cursor for subsequent actions. This position is not necessarily the same as the client's cursor or insertion point.

Members

WTP_REL

Relative to the last position.

WTP_BEG

Positioning from the block beginning.

WTP_END

Positioning from the block end.

WTMODE

Used in the WTC_INIT message to invoke the writing tool in a specific mode of execution. Allowing the client to invoke the writing tool in one of several modes causes the tool to appear to the user as an integral part of the client.

Members

WTM_NORMAL

Normal execution.

WTM_SETUP

Setup mode.

WTM_HELP

Help mode.

WTTOOL

Designates the general classification for the writing tool. As new types of writing tools are made available, WordPerfect Corporation will register them to avoid numbering conflicts.

Members
WTT_DEFAULT

Default writing tool type.

WTT_SPELLER

Spell checking.

WTT_THESAURUS

Thesaurus lookup.

WTT_GRAMMAR

Grammar checking.

WTT_DICTIONARY

Dictionary definitions.

WTT_HYPHENATION

Word hyphenation.

WTT_STATISTICS

Statistical analysis.

WTT_PHONETIC

Phonetic analysis.

WTT_SYNTAX

Syntactical analysis/compilation.

WTT_INPUT

Input method.

WTT_TRANSLATOR

Language translation.

WTTBQTYPE

Used in the *qtype* parameter of the WTQ_TEXTBLOCK query.

Using the WTB_INIT value indicates that a new query should be started.

The WTB_RESUME value is used to request the next value in the text query. It is equivalent to sending the WTQ_NEXTTEXTBLOCK message.

It is not necessary that the client and the writing tool have the same definition of the language boundary elements used in the text query. Because of this, the client may assume that it has sent all text necessary to complete the request. However, the writing tool may require additional text. Instead of initiating a new query, the writing tool may send another WTQ_TEXTBLOCK message, using the WTB_EXTEND value. It also sets the "to" positioning fields to extend the query boundary.

The writing tool may also have reason to request that a query be restarted without starting from the top boundary of the query. By using the WTB_ROLLBACK value, the query can be rolled back to a specific block in the query.

Members

WTB_INIT

Initialize a new query.

WTB_RESUME

Resume the query in progress.

WTB_EXTEND

Extend the query lower bound.

WTB_ROLLBACK

Roll the query back to a block.

WTENDBLOCK

Used in the WTR_TEXTBLOCK message to specify the reason the text buffer ended at a particular position.

Members

WTE_FULL

Text block buffer is full.

WTE_ENDQUERY

End of the query was reached.

WTE_NEWCONTAINER

Starting a nested text container.

WTE_ENDCONTAINER

End of current text container.

WTE_LANGCHANGE

Language change encountered.

WTE_ERROR

Error encountered.

Symbolic Constants

These include mask definitions for bit fields used in the Writing Tools API.

WTAPI_VERSION

The WTAPI_VERSION constant contains the API version under which the application was developed. Writing Tools and clients use this information to simplify compatibility problems among different API versions.

The constant contains the major API revision number in the high byte and the minor version number in the low byte (major * 256 + minor). Applications use this information to determine whether the applications can communicate. API versions that differ only in the minor version are forward and backward compatible.

WTDAMODE

The writing tool data areas allow writing tools to store data specific to their product with the client application. The Writing Tools API defines three storage modes for writing tool data. The WTDAMODE type is a bit field in which the first three bits correspond to these three modes. The next four bits correspond to *control cursor tool data area attributes*. The table below summarizes this information. A more detailed explanation of the bits follows the table.

Definition	**Value**	**Description**
WTD_PRODUCT	0x0001	The data is stored in a product area independent of the current document.
WTD_DOCUMENT	0x0002	The data is stored in a document data area.
WTD_CURSOR	0x0004	The data is stored at the current cursor

Definition	Value	Description
		location.
WTD_VISIBLE	0x0008	The client should put a visible marker at the location of the tool data.
WTD_TEMPORARY	0x0010	The tool data should be kept only for the current writing tool session.
WTD_START	0x0020	Start of a block of writing tool text in client.
WTD_END	0x0040	End of a block of writing tool text in client.

Writing Tool Data Storage Modes

Setting the WTD_PRODUCT bit (0x0001) specifies that the data is stored in a product area independent of the current document. This data can then be retrieved during any writing tool session with that client. This area is useful for client-specific setup information.

Setting the WTD_DOCUMENT bit (0x0002) specifies that the data is stored in a document data area. WordPerfect will store a document-specific supplemental dictionary in this area for the WordPerfect Speller.

Setting the WTD_CURSOR bit (0x0004) specifies that the tool data is stored at the current cursor location. A reference to the tool data area will be sent as a document object in the text and codes buffer, but only if the current text mode allows for formatting and object information. This area allows writing tools to store notes or other information at any location in a document.

Tool Data Area Cursor Control

The next four bits apply only to tool data areas at the cursor.

The WTD_VISIBLE bit (0x0008) is set if the client should put a visible marker at the location of the tool data. Otherwise, the presence of the tool data at the cursor is not shown on the document screen.

The WTD_TEMPORARY bit (0x0010) is set if the writing tool data should be kept at that location only for the current writing tool session. Otherwise, the data is kept at the cursor location for use in subsequent

writing tool sessions.

The WTD_START bit (0x0020) and WTD_END bit (0x0040) are used to mark a section of text in the client as belonging to the writing tool.

WTTEXT

Used to designate the different text formats used in the Writing Tools API. It is not necessary for a writing tool or a client to support all possible text representations. However, all applications should support the WTX_NATIVE type. The following values are defined:

Definition	Value	Description
WTX_NATIVE	0x00000001	Native operating system format
WTX_STYLE	0x00000002	Native operating system format with style information
WTX_UNICODE	0x00000004	Unicode
reserved	0x00000008	
WTX_WPWRD	0x00000010	WordPerfect word string format
WTX_WP6	0x00000020	WordPerfect 6.*x* document format
reserved	0x00000040	
reserved	0x00000080	
WTX_RTF	0x00000100	Microsoft RTF

WTUNIT

Used by the WTR_UNITINFO message to specify the text units that are available in the client application. The first 16 bits of this field will be pre-defined by the Writing Tools API. The last 16 may be defined by the client application, if needed. The following definitions are currently used.

Definition	Value	Description
WTU_DEFUNIT	0x00000000	Default client text unit
WTU_SELECTION	0x00000001	User-selected text
WTU_CHAR	0x00000002	Character units

Definition	Value	Description
WTU_WORD	0x00000004	Word units
WTU_LINE	0x00000008	Line units
WTU_SENTENCE	0x00000010	Sentence units
WTU_PARAGRAPH	0x00000020	Paragraph units
WTU_PAGE	0x00000040	Page units
WTU_DOCUMENT	0x00000080	Document units
WTU_ENTRYBOX	0x00000100	Text entry box units

WTACTION

Used in activation messages. The writing tool sends its current state to the client in the WTC_TLINACTIVE message as it is deactivated. The client sets the writing tool state in the WTC_CLACTIVE and WTC_CLINACTIVE messages.

When bit 0 is clear, the writing tool is suspended (WTA_SUSPEND). Setting this bit causes the writing tool to resume activity (WTA_RESUME).

Bit 1 is clear when the writing tool's input window is in front of the client window (WTA_FRONT). Setting the bit with the WTA_BACK constant puts the writing tool window behind the client.

Bit 2 determines whether or not the tool window is visible to the user. The WTA_SHOW constant specifies that the window is visible, although it may be partially or fully covered by other windows. The WTA_HIDE constant means the window is hidden or invisible to the user.

Bit 3 is clear when the writing tool appears as a window (WTA_WINDOW). If the tool appears as an icon, bit 3 is set (WTA_ICON).

WTINFO

Sent in the *newinfo* parameter of the WTR_TLACTIVE message. It is used to inform the writing tool what non-text information was changed during the client editing session.

The WTI_INFOBLOCK bit is set when the tool should request a new INFOBLOCK. This would happen, for example, if the user saved the current document under a new name.

The WTI_UNITINFO bit is set when the tool should send another WTQ_UNITINFO message. This would be necessary if the user went to a part of the text where units different than the original units were supported.

Platform-Specific Types

These types are used by the Writing Tools API, but are defined on a platform-specific level.

WTCOMM

An ID or *struct* * that contains the platform-specific information necessary for WTAPI communication.

WTBUF, WTBUFP

Equivalent to type *char* and *char* *. Buffers (arrays) of these types are used to hold WTAPI messages and data.

WTSIZE, WTSIZEP

Equivalent to type *unsigned int* and *unsigned int* *. Long integers may be used on some platforms.

WTCOUNT, WTCOUNTP

Equivalent to type *signed int* and *signed int* *. Long integers may be used on some platforms.

BIT FIELDS

The WTACTION, WTDAMODE, and WTINFO types are defined as 16-bit fields. The WTTEXT type and WTUNIT type are defined as 32-bit fields.

WTLANG, WTLANGP

WTLANG is a structure that contains a language designation. WTLANGP is a pointer to this structure. The fields in this structure are defined as follows:

script	script of text (byte)
language	language of text (byte)
region	region (dialect) of text (word)

WTAPI Messages

This section contains descriptions of Writing Tools API messages. Platform-independent parameters are covered. For each message we present the message structure, a description of this message, the defined types associated with this message, parameter descriptions, a description of any variable-length data that accompanies the message. The structure name for each message is the same as the enumerated type name with all underscores removed.

WTC_CLACTIVE

The client sends the WTC_CLACTIVE message to the writing tool whenever it becomes active or wishes to change the action of the writing tool.

Structure

typedef struct

```
{
    WTMSGID msgid;
    WTACTION    action;
} WTCCLACTIVE;
```

Types

WTCCLACTIVE

A structure containing the WTC_CLACTIVE message.

WTCCLACTIVEP

A pointer to the WTCCLACTIVE structure.

Members

msgid

WTC_CLACTIVE.

action

The action that the writing tool should take upon receipt of this message.

Data

None.

WTC_CLINACTIVE

The client sends the WTC_CLINACTIVE message whenever it loses activation.

Structure

```
typedef struct
{
    WTMSGID msgid;
    WTACTION    action;
} WTCCLINACTIVE;
```

Types

WTCCLINACTIVE

A structure containing the WTC_CLINACTIVE message.

WTCCLINACTIVEP

A pointer to the WTCCLINACTIVE structure.

Members

msgid

WTC_CLINACTIVE.

action

The action that the writing tool should take upon receipt of this message.

Data

None.

WTC_INIT

Sent after any operating-system-dependent functions have taken place, such as loading the application into memory and establishing a communication link. The client sends the WTC_INIT message to initiate a conversation with a writing tool.

Structure

```
typedef struct
{
    WTMSGID msgid;
    WTMODE      mode;
    int             version;
    Window          windowClient;
} WTCINIT;
```

Types

WTCINIT

A structure containing the WTC_INIT message.

WTCINITP

A pointer to the WTCINIT structure.

Members

msgid

WTC_INIT.

mode

Specifies the desired execution mode of the writing tool. When the user selects the writing tool from the client's tool menu, it invokes the writing tool in WTNORMAL mode, specifying normal writing tool execution. The writing tool may also appear in a client's Preferences menu. Selecting a tool from this location would cause the client to invoke the tool in WTSETUP mode. This allows the user to access writing tool preferences from within the client. Similarly, the user may wish to get help on a writing tool. The client can invoke the tool in WTHELP mode, indicating that the tool should present a help list to the user.

version

Contains the WTAPI version that the client uses.

windowClient

The window ID of the client application window. This parameter is specific to the UNIX platform.

Data

None.

WTC_RQCLACTIVE

A request for the client to become the active application. This allows the user to make editing changes directly in the client's edit buffer.

Structure

```
typedef struct
{
    WTMSGID msgid;
} WTCRQCLEDIT;
```

Types

WTCRQCLACTIVE

A structure containing the WTC_RQCLACTIVE message.

WTCRQCLACTIVEP

A pointer to the WTCRQCLACTIVE structure.

Members

msgid

WTC_RQCLACTIVE.

Data

None.

WTC_RQINIT

The writing tool sends the WTC_RQINIT message to request that the client initiate a writing tool session. This message allows a writing tool to request initiation of a writing tool session.

Structure

```
typedef struct
{
    WTMSGID msgid;
    WTBOOL      newdoc;
    WTSIZE    size;
} WTCRQINIT;
```

Types

WTCRQINIT

A structure containing the WTC_RQINIT message.

WTCRQINITP

A pointer to the WTCRQINIT structure.

Members

msgid

WTC_RQINIT.

newdoc

This parameter is set to WT_TRUE if the client should open a new (empty) document for the session. A WT_FALSE value indicates that the current document should be used.

size

If this value is non-zero, the writing tool has specified a document on disk for the writing tool session.

Data

The pathname to the document to be opened as the object of the writing tool session may be sent. The size of the path name is indicated by the *size* parameter. The path may specify an existing or a new file.

WTC_RQTLTERM

The client sends a WTC_RQTLTERM message to notify the writing tool that it wishes to terminate the writing tool session.

Structure

```
typedef struct
{
    WTMSGID msgid;
} WTCRQTLTERM;
```

Types

WTCRQTLTERM

A structure containing the WTC_RQTLTERM message.

WTCRQTLTERMP

A pointer to the WTCRQTLTERM structure.

Members

msgid

WTC_RQTLTERM.

Data

None.

WTC_TLINACTIVE

The writing tool sends the WTC_TLINACTIVE message whenever it loses activation.

Structure

```
typedef struct
{
    WTMSGID msgid;
    WTACTION    action;
} WTCTLINACTIVE;
```

Types

WTCTLINACTIVE

A structure containing the WTC_TLINACTIVE message.

WTCTLINACTIVEP

A pointer to the WTCTLINACTIVE structure.

Members

msgid

WTC_TLINACTIVE.

action

The actions the writing tool is currently taking.

Data

None.

WTC_TLTERM

The writing tool sends a WTC_TLTERM message to communicate the termination of the writing tool session.

Structure

```
typedef struct
{
    WTMSGID msgid;
} WTCTLTERM;
```

Types

WTCTLTERM

A structure containing the WTC_TLTERM message.

WTCTLTERMP

A pointer to the WTCTLTERM structure.

Members

msgid

WTC_TLTERM.

Data

None.