FaxTalk Software Development Kit (SDK)

Introducing the FaxTalk SDK

This document introduces the FaxTalk Application Programming Interface (FaxTalk API), included in the FaxTalk Software Development Kit (FaxTalk SDK), a kit that enables you to enhance the functionality of FaxTalk™ version 8.0 or later.

About the FaxTalk SDK

The FaxTalk SDK enables software developers to extend the functionality of FaxTalk. The SDK uses the application programming interfaces (APIs), which employ Microsoft's Component Object Model (COM) technology.

The FaxTalk SDK is a set of COM objects, header files, sample code, and documentation designed to help you create programs that provide automated access to FaxTalk functionality. Using the FaxTalk SDK, you can set up automated access to:

- Submitting faxes to be sent with the FaxTalk send engine
- Selecting and filling in cover pages for faxes sent
- Read/Modify/Delete the FaxTalk Inbox, Outbox, Sent, and Transaction Log folder contents
- Send and receive status
- FaxTalk application configuration information
- Integrate client applications with the FaxTalk folders
- Conversion to and from FaxTalk image format files (e.g. convert between a FaxTalk image received and popular formats like PDF)
- Real time event notifications when an item in a FaxTalk Inbox, Outbox, Sent, or Transaction Log folder has changed

Each area of FaxTalk functionality is accessed through a FaxTalk SDK object. This document describes each FaxTalk SDK object and its functions.

What's Included

The FaxTalk SDK includes the following:

- ✓ **Application programming objects** – Includes the following objects: Send, Log, Coversheet, Configuration, Events, and Viewer
- ✓ **Application SDK documentation** – This document.
- ✓ **Sample code** – Located in the SDK folder of the FaxTalk application program install folder (Retail or Trial version). Includes a sample application and corresponding source code in C# (C Sharp).

Implementation Notes

The FaxTalk SDK assumes that you have at least a working knowledge of the Microsoft Win32 API, as well as a basic knowledge of Microsoft COM. When using this SDK, keep in mind the concepts and information described in the following sections:

Entry IDs

The event notification mechanism implemented by the Send object returns to the client the unique ID (entry ID) associated with a fax event in FaxTalk. This entry ID uniquely identifies this event (record) in the folder and it can be used by methods in the FaxTalk.APILOG object, which requires in-parameters of type EntryID.

Terminology

This section defines and explains terminology used in this document.

- **Folder** – Containers that hold Inbox, Outbox, and Transaction Log records (three Separate folders with no subfolders). Note that Outbox and Sent records are contained in the same folder.
- **EntryID** – Each record that is contained in a folder must have an unique entry ID (unique within That folder). This EntryID will not change over time, unless a database repair operation is performed and the repair/optimize option is enabled. This ensures that once a client application acquires the unique identifier, it will always be able to use that unique identifier to communicate with the same record in the folder.
- **Folder Record** – Each folder has 0 or more records, each of which is used to store the data associated with 1 fax, voice message, call, voice notification, or pager notification. Each record is uniquely identified and accessed with its EntryID.

Technical Support

Technical support is available. Please submit a Support Request from the Support section of our website www.faxtalk.com to initiate support.

Send Object

This section describes the Send object, a component of the FaxTalk Software Development Kit. You can use this object to send faxes from client applications using the FaxTalk send functionality.

Overview

The Send object allows client applications to send faxes using the FaxTalk send functionality normally available only through the FaxTalk Send dialog. Send jobs can include simple coversheet messages or multiple documents (for example, coversheet text, a Word or Excel document, and so on). All documents included in the send job are converted to FaxTalk format at submission time.

The Send object includes all the parameters available for composing and sending faxes. As in FaxTalk, some of these parameters can be set up on a recipient-by-recipient basis; while others are set up for a single send job (may include multiple recipients).

Send Object Identifiers

The Send object has the following ProgID and unique identifier:

ProgID: **FaxTalk.API**
CLSID: **{49EC7778-D69F-4EA7-9A34-DB008122734D}**

Implementation Notes

You can use the Send object as a standalone fax engine or in conjunction with other SDK objects. The Send object includes functionality that can return unique entryID identifiers for each send job submitted by client applications. Client applications can then use these entryIDs in the Log object methods to get additional information such as the current send status.

For each instance of the Send object, you can create only one send job with one or more recipients. After the job is successfully sent, you must destroy the old object and create a new Send object for the next send job.

Properties	Description / Syntax
To	Name of the recipient. Added to the list with the call to AddRecipient()
Number	The fax number. Added to the list with the call to AddRecipient()
AreaCode	The fax number area code. Added to the list with the call to AddRecipient()
Company	Recipient Company. Added to the list with the call to AddRecipient()
UseTrial	Specify **True** to set the default version to Trial or **False** to set the default version to Purchased. (default)
AlternateVersionBinding	Specify **True** to enable binding to the alternate version of FaxTalk (Trial/Purchased). If the Trial version is set as the default, and is not found, then the Purchased version will be attempted. If the Purchased version is the default, and is not found, then the Trial version will be attempted

	(default) or **False** to disable binding to the alternate version of FaxTalk (Trial/Purchased).
Configuration	Returns the FaxTalk API configuration object
EnableDiagnostics	Enables diagnostic logging. **Syntax:** HRESULT EnableDiagnostics(BOOL pVal) **pVal** - Set to FALSE to disable diagnostic logging or TRUE to enable diagnostic logging.
DialFlags *Available in Version 10, build 1634 or later*	Overrides the default dialing for all recipient numbers. **Syntax:** HRESULT DialFlags (short Val) **Val** - This is a bit field and the following bit positions are defined. Bit 0: Not used Bit 1: Dial as long distance Bit 2: Dial as entered

Methods	**Description / Syntax**
Init()	Initialize the FaxTalk API object. This method must be called prior to calling any other method.
Send(EntryId)	Call this method to send the converted attachments. The AddRecipient() must be called before this method. **EntryId** - If this parameter is set to 1, Faxtalk will prepare entry ID's for the send job. Otherwise, the entry ID is not prepared.
AddAttachmentFile(filename)	Call this method for each attachment to be converted and sent with the fax. **filename** - Fully qualified path to the file to be converted. The associated application will be called to print the file to the FaxTalk Print driver.
AddRecipient()	Call this method for each recipient. The Number property must be set prior to calling this method. The To, and Company properties may also be set prior to calling this method.
Done()	Disconnect from FaxTalk and clean up all references.
ShowSendScreen(ShowSendScreen)	Enables / Disables the Send a Fax UI when calling Send(). The default is enabled. Specify 0 to disable the user interface or 1 to enable the user interface

PrintToAttachment	Initializes the FaxTalk conversion to convert an attachment file and attaches the attachment to the list.
IsPrintingToAttachment	Returns True if the FaxTalk conversion component is printing to an attachment. Returns False if the printing has completed.
SetNumber(RecipientNumber)	**RecipientNumber** - The fax number. Added to the list with the call to AddRecipient()
SetTo(RecipientName)	**RecipientName** - Name of the recipient. Added to the list with the call to AddRecipient()
SetCompany(RecipientCompany)	**Recipient Company** - Recipient's company name. Added to the list with the call to AddRecipient()
SetHold(HoldFlag)	Specify 0 to not set the fax on hold (default) or specify 1 to set the fax on hold.
SetDate(DateString)	**DateString** - Date in MM/DD/YYYY format. If not set, current date is used.
SetTime(TimeString)	**TimeString** - Time in HH:MM:SS 24-hour format. If not set, current time is used.
SetCoverPage(filename)	**filename** - Fully qualified path to the coversheet file. The filename+extension can be used if the coversheet is in the existing User data folder.
SetSubject(subject)	**subject** - Set the coversheet subject text
SetCoverText(text)	**text** - Set the coversheet memo text
SetUseCover(usecover)	Specify 0 to not send coversheet or 1 to send a coversheet (default)
SetResolution(resolution)	Specify 0 to send using standard resolution or 1 to send using fine resolution. If not set, this function uses the default setting configured in the FaxTalk software.
IsEntryIDReady *Available in Version 8, build 2010 or later*	Returns TRUE if entry ID associated with index is ready to be retrieved. **Syntax:** HRESULT IsEntryIDReady(short Index, BOOL* pVal) **short Index** - The 0-based index of the recipient in the send job. **pVal** - Out parameter. Returns TRUE if entry ID associated with index is ready to be retrieved.
GetEntryID	Returns the entry ID associated with the event created for the recipient index in the send job.

Available in Version 8, build 2010 or later	**Syntax:** HRESULT GetEntryID(short Index, BSTR *pVal) **short Index** - The 0-based index of the recipient in the send job. **pVal** - Out parameter. The entry ID.
SetOffPeak *Available in Version 8, build 2010 or later*	Instructs FaxTalk to use the Off Peak feature. This function returns 1 on error and 0 on OK. If this function is not used, FaxTalk does not use the off-peak period by default. **Syntax:** HRESULT SetOffPeak(short OffPeak, SHORT *pVal) **short OffPeak** - If this parameter is set to 1, FaxTalk sends the job during the off-peak time period for this recipient. **pVal** - Out parameter. Set to 1 if an error occurred.
IsError *Available in Version 8, build 2010 or later*	Returns TRUE if there was an error. **Syntax:** HRESULT IsError(BOOL *pVal) **pVal** - Out parameter. Set to TRUE if an error occurred.
GetLastError *Available in Version 8, build 2010 or later*	Returns the last error and resets the internal error status to SDKLOGERROR_NOERROR. **Syntax:** HRESULT GetLastError(long *pVal) **pVal** - Out parameter. The last error.
SetAppData *Available in Version 10, build 1625 or later*	Provides a means for the application creating this send job to store application specific data with it (for later use by the application). **Syntax:** HRESULT SetAppData(BSTR AppID, BSTR newVal) **AppID** – An application identifier assigned by FaxTalk to uniquely identify the application.Maximum string lengh is 1. **newVal** – Application specific data stored. Maximum string lengh is 11.

Log Object

This section describes the Log object, a component of the FaxTalk Application Software Development Kit. You can use this object to integrate client applications with the FaxTalk folders. The Log Object is available in version 8.0 build 2010 or later.

Overview

This object allows client applications to integrate with the FaxTalk folders (that is, the Inbox, Outbox, Sent, and transaction Log folders and records contained within them). The Log object accesses records that were entered into the folders by a send or receive action. It does not perform the send or receive action itself. Typically, a client application might use the Log object to display a graphic representation of the FaxTalk folders and populate these folders with faxes and messages. In addition, a client application can obtain an EventID from the Send object, and look up the status of the record in the Outbox folder. Records are updated each time a fax is sent or received.

The Log object may be used in conjunction with other SDK objects (for example, the Send object) to extend the functionality provided by those objects.

Log Object Identifier

The Log object has the following ProgId and unique identifier:

ProgID: **FaxTalk.APILog**
CLSID: **{7B0CB17E-9542-49b9-8B6F-0B3BB1D1931B}**

Implementation Notes

Each record in a folder is referenced by a unique EntryID. These EntryIDs are passed to the client through counted strings (that is, BSTR in Visual C++). These strings are composed of a representation of binary data. When working with EntryIDs, do the following:

- Test to determine if their length is zero (0) or non-zero.
- Store them to pass back to FaxTalk through the objects to reference a particular item.

FaxTalk provides Inbox, Outbox, and Transaction Log folders, with no subfolders contained within them. Note that the Outbox folder contains both items waiting to be sent and items already successfully sent (the FaxTalk application further separates these into Outbox and Sent folders. These 3 folders have a standard name by which you can access them. For example, to find the send status of a record in the Outbox folder given an EventID from the Send object, you might call GetRecordListFirst and pass in STANDARDFOLDER_FAXTALK_OUTBOX as the first parameter. Because you are specifying a standard folder, the folder ID you pass in as the second parameter is ignored and can be an empty string. This call would return to you the EventID of the first record in your Outbox folder, if one exists. If there are no records in the Outbox folder, it returns the empty string as the EventID. If it did return an EventID, you can then make a call to GetRecordStatus passing in your EventID to find out the current status of the fax send record.

A client application can use the method GetFolderChangeCounter in a polling loop to determine when a folder has been changed and it’s contents need to be processed. Using this method a client application can significantly improve perfomance when accessing FaxTalk folders.

Properties	Description / Syntax
EnableDiagnostics	Enables diagnostic logging. **Syntax:** HRESULT EnableDiagnostics(BOOL pVal) **pVal** - Set to FALSE to disable diagnostic logging or TRUE to enable diagnostic logging.

Methods	Description / Syntax
GetFolderListFirst *(currently not implemented)*	Returns the FolderID of the first folder in the specified base folder. When enumerating the folders within a specific folder, call this function to get the first one. Follow up with calls to GetFolderListNext. This function returns a NULL BSTR if there are no folders. **Syntax:** HRESULT GetFolderListFirst(short StandardFolder, BSTR FolderID, BSTR* pVal) **StandardFolder** - In parameter. ID of the folder to search. If STANDARDFOLDER_NONE, use FolderID instead. **FolderID** - In parameter. ID of the folder to search, if STANDARDFOLDER_NONE was specified as the standardFolder. **pVal** - Out parameter. FolderID of the first folder in the specified base folder.
GetFolderListNext *(currently not implemented)*	Returns the FolderID of the next folder in the folder specified in GetFolderListFirst. When enumerating the folders within a specific folder, call GetFolderListFirst to get the first one. Follow up with calls to this method. This function returns a NULL BSTR if there is no next folder. **Syntax:** HRESULT GetFolderListNext(BSTR* pVal) **pVal** - Out parameter. FolderID of the next folder in the specified base folder.
GetRecordListFirst	Returns the EntryID of the first record in the specified folder. When enumerating the records within a specific folder, call this function to get the first one. Follow up with calls to GetRecordListNext. This function returns a NULL BSTR if there are no records. **Syntax:** HRESULT GetRecordListFirst(short StandardFolder, BSTR FolderID, BSTR* pVal) **StandardFolder** - In parameter. ID of the folder to search. If STANDARDFOLDER_NONE, use FolderID instead. **FolderID** - In parameter. ID of the folder to search, if

	STANDARDFOLDER_NONE was specified as the standardFolder. **pVal** - Out parameter. EntryID of the first record in the specified folder (if one exists).
GetRecordListNext	Returns the EntryID of the next record in the specified folder. When enumerating the records within a specific folder, call GetRecordListFirst to get the first one. Follow up with calls to this method. This function returns a NULL BSTR if there is no next record. **Syntax:** HRESULT GetRecordListNext(BSTR* pVal) **pVal** - Out parameter. EntryID of the next record in the specified folder (if one exists).
GetTo/SetTo	Gets/sets the record's To field. **Syntax:** HRESULT GetRecordTo(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordTo(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's To field.
GetNumber/SetNumber	Gets/sets the record's Number field. **Syntax:** HRESULT GetRecordNumber(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordNumber(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Number field.
GetSubject/SetSubject	Gets/sets the record's Subject field. **Syntax:** HRESULT GetRecordSubject(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordSubject(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Subject field. field.
GetCompany/SetCompany	Gets/sets the record's Company field. **Syntax:** HRESULT GetRecordCompany(BSTR EntryID, BSTR* pVal)

	Syntax: HRESULT SetRecordCompany(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Company field.
GetAttachmentCount/ SetAttachmentCount *(currently not implemented)*	Gets/sets the number of attachments for the record. **Syntax:** HRESULT GetRecordAttachmentCount(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetRecordAttachmentCount(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. EntryID of the record. **pVal -** The record's number of attachments field.
GetHold/SetHold	Gets/sets the record's Hold field. **Syntax:** HRESULT GetRecordHold(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetRecordHold(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Hold field.
GetDate/SetDate	Gets/sets the date of the record. **Syntax:** HRESULT GetRecordDate(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordDate(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Date field.
GetTime/SetTime	Gets/sets the time of the record. **Syntax:** HRESULT GetRecordTime(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordTime(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Time field.
GetType/SetType	Gets/sets the Record type. The record type can be one of:

	▪ EVENTTYPE_UNKNOWN – Record type is unknown. ▪ EVENTTYPE_LGFOLDER – Record is a log folder. ▪ EVENTTYPE_FAX – Record is a fax. ▪ EVENTTYPE_VOICE – Record is a voice message. **Syntax:** HRESULT GetType(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetType(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Type field.
GetStatus/SetStatus	Gets/sets the status of the record. This function returns are of the type TSDKSendEventStatus for entries in the STANDARDFOLDER_FAXTALK_OUTBOX, and TSDKRecvEventStatus for entries in STANDARDFOLDER_FAXTALK_LOG and STANDARDFOLDER_FAXTALK_RECEIVELOG. **Syntax:** HRESULT GetRecordStatus(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetRecordStatus(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's Status field.
GetCSID/SetCSID	Gets/sets the calling station identifier (CSID) of the record. **Syntax:** HRESULT GetRecordCSID(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordCSID(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The record's CSID field.
GetPageCount/ SetPageCount	Gets/sets the number of fax pages included in the record. **Syntax:** HRESULT GetRecordPageCount(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordPageCount(BSTR EntryID, BSTR pVal)

	EntryID - In parameter. EntryID of the record. **pVal** - The record's Page count field.
GetResolution/SetResolution	Gets/sets the resolution of the specified fax record. The possible values are: 1(0x1) = 204x98 2(0x2) = 204x196 4(0x4) = 200x100 8(0x8) = 200x200 -1 = Undefined or not a standard resolution **Syntax:** HRESULT GetRecordResolution(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetRecordResolution(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. The EntryID of the record. **pVal** - The record's Resolution field.
GetDuration/SetDuration	Gets/sets the record duration in seconds. **Syntax:** HRESULT GetRecordDuration(BSTR EntryID, long *pVal) **Syntax:** HRESULT SetRecordDuration(BSTR EntryID, long pVal) **EntryID** - In parameter. The EntryID of the record. **pVal** - The record's Duration field.
GetUsedECM/SetUsedECM	Gets/sets whether Error Correction Mode was used in the transmission of this record. This function returns a non-zero value if ECM was used; otherwise it returns zero. **Syntax:** HRESULT GetRecordUsedECM(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetRecordUsedECM(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. The EntryID of the record. **pVal** - The record's ECM field.
GetDeviceNameUsed/ SetDeviceNameUsed *(currently not implemented)*	Gets/sets the name of the device used to deliver record. **Syntax:** HRESULT GetRecordDeviceNameUsed(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetRecordDeviceNameUsed(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. The EntryID of the record.

	pVal - The record's Device field.
GetRetries/SetRetries	Gets/sets the number of retries associated with this record. **Syntax:** HRESULT GetRecordRetries(BSTR EntryID, SHORT *pVal) **Syntax:** HRESULT SetRecordRetries(BSTR EntryID, SHORT pVal) **EntryID** - In parameter. The EntryID of the record. **pVal** - The record's Retries field.
Done	Informs FaxTalk that folder processing is finished. This allows Faxtalk to close access to the folders and do necessary cleanup **Syntax:** HRESULT Done()
GetFolderChangeCounter *Available in Version 8, build 2010 or later*	Each time a record is added, deleted, or modified in a folder, an associated counter (long value) is incremented. This counter is never 0. This method allows a caller to pass in a counter which is compared to the internal folder counter, and a result of TRUE is returned if the counter has a different value. In addition, the contents of the user supplied counter is updated with the value of the folder counter. **Syntax:** HRESULT GetFolderChangeCounter(short StandardFolder, BSTR FolderID, long *pUpdateCount, BOOL* pVal) **StandardFolder** - In parameter. ID of the folder to search. **FolderID** - Not used. **pUpdateCount** - A pointer to the user supplied counter that is compared to the folder counter; this location is updated upon return. **pVal** - Out parameter. TRUE if the the two counters differ.
IsError *Available in Version 8, build 2010 or later*	Returns TRUE if there was an error. **Syntax:** HRESULT IsError(BOOL *pVal) **pVal** - Out parameter. Set to TRUE if an error occurred.
GetLastError	Returns the last error and resets the internal error status to SDKLOGERROR_NOERROR. **Syntax:** HRESULT GetLastError(long *pVal) **pVal** - Out parameter. The last error.

GetCIDNumber/SetCIDNumber *Available in Version 8, build 2530 or later*	Gets/sets the record's Caller ID Number field. **Syntax:** HRESULT GetCIDNumber(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetCIDNumber(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The records Caller ID Number field.
GetCIDMessage/SetCIDMessage *Available in Version 8, build 2530 or later*	Gets/sets the record's Caller ID Message field. **Syntax:** HRESULT GetCIDMessage(BSTR EntryID, BSTR* pVal) **Syntax:** HRESULT SetCIDMessage(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - The records Caller ID Message field.
DeleteMessage *Available in Version 8, build 2610 or later*	Deleted the message from the folder **Syntax:** HRESULT DeleteMessage(BSTR EntryID, BSTR pVal) **EntryID** - In parameter. EntryID of the record. **pVal** – In parameter. AccessBy method used.
GetResultCode/SetResultCode *Available in version 10.0 or later*	Gets/ sets the record's Result Code field. **Syntax:** HRESULT GetResultCode (BSTR EntryID, SHORT* pVal) **Syntax:** HRESULT SetResultCode (BSTR EntryID, SHORT pVal) **EntryID** - In parameter. EntryID of the record. **pVal** – The records Result Code field
GetFilename *Available in version 10.0 or later*	Gets the record's Filename field. **Syntax:** HRESULT GetFilename(BSTR EntryID, BSTR* pVal) **EntryID** - In parameter. EntryID of the record. **pVal** - Out parameter. The records Filename field.
LoadEventRaw *Available in version 10.0 or later*	Loads an opaque structure returned by the Events object, allowing the caller to get/set record fields.. **Syntax:** HRESULT LoadEventRaw(BSTR EventRaw, BSTR* EntryID)

	EventRaw - In parameter. An opaque structure returned by the Events object. **EntryID** - Out parameter. EntryID of the record.
GetAppData/SetAppData *Available in version 10.0 Buil;d 1625 or later*	Gets/sets the record's Application Data field. The calling application can only change this data if it was the original application that created it, or it hasn't been created yet. **Syntax:** HRESULT GetAppData(BSTR EntryID, BSTR AppID, BSTR* pVal) **Syntax:** HRESULT SetAppData(BSTR EntryID, BSTR AppID, BSTR newVal) **EntryID** - In parameter. EntryID of the record. **AppID** – An application identifier assigned by FaxTalk to uniquely identify the application.Maximum string lengh is 1. **newVal** – Application specific data stored. Maximum string lengh is 11.

CoverSheetCollection Object

This section describes the CoverSheetCollection object, a component of the FaxTalk Software Development Kit. You can use this object to enumerate the list of available coversheet templates and select 1 for use in the Send object.

Overview

FaxTalk API CoverSheetCollection object.

Log Object Identifier

The Coversheet object has the following ProgId and unique identifier:

ProgID: **FaxTalk.APICoverSheetCollection**
CLSID: **{E2E6CC16-DB71-465B-9B04-F42566F9412F}**

Properties	Description / Syntax
Count	Returns the number of coversheets in the list
EnableDiagnostics	Enables diagnostic logging. **Syntax:** HRESULT EnableDiagnostics(BOOL pVal) **pVal** - Set to FALSE to disable diagnostic logging or TRUE to enable diagnostic logging.

Methods	Description / Syntax
Item(index)	**index** - 0 based index into the coversheet list. Returns the coversheet name for the index.

Configuration Object

This section describes the Configuration object, a component of the FaxTalk Software Development Kit. You can use this object to access various FaxTalk installation settings

Overview

FaxTalk API Configuration object.

Configuration Object Identifier

The Configuration object has the following ProgId and unique identifier:

ProgID: **FaxTalk.APIConfig**
CLSID: **{36C97F22-5F39-43BB-B0BF-F1F13564F60C}**

Properties	Description / Syntax
PrinterName	Returns the FaxTalk printer name
ProductName	Returns the FaxTalk product name
HomeFolder	Returns the home folder (installation folder) of the FaxTalk product.
CoverSheets	Returns CoverSheet collection object
BuildVersion *Available in Version 8, build 2530 or later*	Returns the version and build number of the installed software. The format of the version data is w.x.y.zzzz where: w = Major version x = Minor version y = Patch version zzzz = Build version The product version and build number can also be read from the Windows Registry. To determine the version of FaxTalk check the “Build” string value under HKEY_LOCAL_MACHINE\SOFTWARE\Thought Communications\FaxTalk\Setup (for retail products) or HKEY_LOCAL_MACHINE\SOFTWARE\Thought Communications\FaxTalk Trial\Setup (for trial products). The “Build” value follows the same w.x.y.zzzz format.

Methods	Description / Syntax
GetDataFolder *Available in Version 10, build 1630 or later*	Returns the full path name to a FaxTalk data folder (e.g. Inbox or Outbox location). **Syntax:** HRESULT GetDataFolder(short Location, BSTR* pVal)

	Location – In parameter. An enum type TSDKStandardFolder indicating the requested folder path. **pVal** – Out parameter. The full path name to the folder.

Events Object

This section describes the Events object, a component of the FaxTalk Software Development Kit. This object was added starting with the first release of FaxTalk Version 10.0. You can use this object to receive notifications (either polling or asynchronously) when an entry in a FaxTalk folder is added, deleted, or modified. In the case of a modify event notification, the Event notification includes details on what changed in the folder entry.

Overview

The Event object allows client applications to be notified when an entry in a FaxTalk folder has changed. Applications can use this to tract the status of an individual entry, or maintain a current list of entries in a folder much the same way as the FaxTalk Inbox, Outbox, Sent, and Transactions folders.

Events Object Identifiers

The Events object has the following ProgID and unique identifier:

ProgID: **FaxTalk.APIEvents**
CLSID: **{D1FF0BCE-3085-45D2-82B3-2E2082E1AD60}**

Implementation Notes

You can choose to poll for events using GetEventRaw(), or you can be notified of a waiting event by passing a Windows Handle and Message ID in the call to Init(). If you choose the latter, when a new event is available, a call to the Windows API function SendMessage() is made using this Windows Handle and Message ID to notify your application that one or more events are waiting. Upon receiving this message, the application should call GetEventRaw() repeatedly until no more events are returned.

After creating an Events object and initializing it with a call to Init(), repeated calls to Open() and Close() can be made if the application wants to change which folders are reporting events, or reinitialize event reporting. Remember that you can also choose to enable event reporting for one or more folders at the same time. The return parameters from GetEventRaw() will indicated which folder the event applies to. Passing the returned Raw opaque structure to a Log object though the Log object method LoadEventRaw() will allow the reported event details to be read.

Properties	Description / Syntax
UseTrial	Specify **True** to set the default version to Trial or **False** to set the default version to Purchased. (default)
AlternateVersionBinding	Specify **True** to enable binding to the alternate version of FaxTalk (Trial/Purchased). If the Trial version is set as the default, and is not found, then the Purchased version will be attempted. If the Purchased version is the default, and is not found, then the Trial version will be attempted (default) or **False** to disable binding to the alternate version of FaxTalk (Trial/Purchased).
EnableDiagnostics	Enables diagnostic logging. **Syntax:** HRESULT EnableDiagnostics(BOOL pVal) **pVal** - Set to FALSE to disable diagnostic logging or TRUE to enable diagnostic logging.

Methods	Description / Syntax
Init	Initialize the FaxTalk Events object. This method must be called prior to calling any other method. **Syntax:** HRESULT Init()
Open	Start receiving event notifications. **Syntax:** HRESULT Open(long hWnd, UINT uNotifyId, DWORD dwEvents) **hWnd** - In parameter. A Windows Handle used in SendMessage() to notify the application of a waiting event. This can be NULL if polling is used. **uNotifyId** – In parameter. A Windows Message ID used in SendMessage() to notify the application of a waiting event. This can be 0 if polling is used. **dwEvents** – In parameter. One or more combined options indicating which folders to enable event notification.
GetEventRaw	Call this method to get an event notification. **Syntax:** HRESULT GetEventRaw(int *pType, int *pAction, BSTR* pEventRaw) **pType** – Out parameter. Indicates which folder this event applies to. **pAction** – Out parameter. Indicates whether this is an Add, Delete, or Modify event. **pEventRaw** – Out parameter. An opaque structure containing the event details. This can be passed to a Log object to get details of the event.
Close	Stop receiving event notifications. **Syntax:** HRESULT Close()
Done	Disconnect from Events object and clean up all references. **Syntax:** HRESULT Done()
IsError	Returns TRUE if there was an error. **Syntax:** HRESULT IsError(BOOL *pVal) **pVal** - Out parameter. Set to TRUE if an error occurred.
GetLastError	Returns the last error and resets the internal error status to SDKLOGERROR_NOERROR. **Syntax:** HRESULT GetLastError(long *pVal)\

pVal - Out parameter. The last error.

Viewer Object

This section describes the Viewer object, a component of the FaxTalk Software Development Kit. This object was added starting with the first release of FaxTalk Version 10.0. You can use this object to create FaxTalk format files needed in the Send object for sending faxes, and it can be used to convert received FaxTalk format files to other formats like PDF.

Overview

The Viewer object allows client applications to create FaxTalk format files needed in the Send object for sending a fax. Applications can use this to convert received faxes (in FaxTalk format) to other non-proprietary formats like PDF, TIFF, BMP, JPEG, and PNG. It can also be used to display a FaxTalk format file in the FaxTalk viewer.

Viewer Object Identifiers

The Viewer object has the following ProgID and unique identifier:

ProgID: **FaxTalk.APIViewer**
CLSID: **{9C4F7728-A2F7-4129-B989-A9CE0170563C}**

Implementation Notes

You can call the method FormatCapabilities() to find out what formats are supported for converting to and from FaxTalk format. After creating a Viewer object and calling Init() to initialize it, a client application calls OpenFile() to load the first file, and AttachFile() to add additional files. When the call is made to SaveAs(), these files are converted to the requested format and saved. If ShowViewerScreen is enabled, the files are displayed in a FaxTalk viewer instead of being converted and saved to the user named file.

Additional calls to OpenFile() reset the file list, and can be used to perform additional conversions without having to recreate the Viewer object.

Properties	Description / Syntax
UseTrial	Specify **True** to set the default version to Trial or **False** to set the default version to Purchased. (default)
AlternateVersionBinding	Specify **True** to enable binding to the alternate version of FaxTalk (Trial/Purchased). If the Trial version is set as the default, and is not found, then the Purchased version will be attempted. If the Purchased version is the default, and is not found, then the Trial version will be attempted (default) or **False** to disable binding to the alternate version of FaxTalk (Trial/Purchased).
EnableDiagnostics	Enables diagnostic logging. **Syntax:** HRESULT EnableDiagnostics(BOOL pVal) **pVal** - Set to FALSE to disable diagnostic logging or TRUE to enable diagnostic logging.
Quality *Available in Version 10, build 1634 or*	Used to set the JPEG compression level. Set to 100 if best quality desired.

later	**Syntax:** HRESULT Quality(short pVal) **pVal** - Set to a value between 0 and 100 inclusive.

Methods	Description / Syntax
FormatCapabilities	Returns the file formats supported for conversion to and from the FaxTalk file format. **Syntax:** HRESULT FormatCapabilities(SHORT *InputFormats, SHORT *OutputFormats) **InputFormats** - Out parameter. The list of file formats supported for conversion to FaxTalk file format. **OutputFormats** - Out parameter. The list of file formats supported for conversion from FaxTalk file format.
Init	Initialize the FaxTalk Viewer object. This method must be called prior to calling any other method. **Syntax:** HRESULT Init(long ParentWnd) **ParentWnd** - In parameter. An optional Windows Handle used for displaying error messages and conversion progress by the FaxTalk viewer server. None will be displayed if this parameter is 0. If it is not 0, then it will be used as the Parent Window to the messages displayed by the FaxTalk Viewer.
OpenFile	Indicate the first file to be converted. **Syntax:** HRESULT OpenFile(BSTR FileName, short Location) **FileName** - In parameter. The name of the file. **Location** – In parameter. The location of the file.
AttachFile	Add an additional file to be converted. **Syntax:** HRESULT AttachFile (BSTR FileName, short Location) **FileName** - In parameter. The name of the file. **Location** – In parameter. The location of the file.
SaveAs	Performs the file conversions, or brings up the FaxTalk viewer if ShowViewerScreen() is enabled. **Syntax:** HRESULT SaveAs(BSTR FileName, short Location, short Options) **FileName** - In parameter. The name of the file. **Location** – In parameter. The location of the file.

	Options - In parameter. Reserved.
ShowViewerScreen	Used to indicate to SaveAs() that the files should be displayed in the FaxTalk viewer. **Syntax:** HRESULT ShowViewerScreen (SHORT showviewerscreen) **showviewerscreen** - In parameter. Set to 1 if the files should displayed in the FaxTalk viewer.
Done	Disconnect from Viewer object and clean up all references. **Syntax:** HRESULT Done()
IsError	Returns TRUE if there was an error. **Syntax:** HRESULT IsError(BOOL *pVal) **pVal** - Out parameter. Set to TRUE if an error occurred.
GetLastError	Returns the last error and resets the internal error status to SDKLOGERROR_NOERROR. **Syntax:** HRESULT GetLastError(long *pVal)\ **pVal** - Out parameter. The last error.

Sample Code: FaxTalkAPIClient (C#)

The FaxTalk SDK includes a sample application called “FaxTalkAPIClient.exe” (located within the Samples folder found in SDK folder). This sample application is a C# .Net forms application that demonstrates the methods and properties of the FaxTalk API. The sample allows attaching multiple documents and sending to multiple recipients.

The API calls are in the Form1.cs source file.