ADVANTAGE TECHNOLOGIES

opentext™

OpenText RightFax CE 22.2

API

Reference Guide

Edition

OpenText RightFax CE 22.2 API

May 05, 2022

Open Text Corporation
275 Frank Tompa Drive
Waterloo, Ontario, Canada
N2L 0A1
(519) 888-7111
http://www.opentext.com

Copyright Statement

Disclaimer

No Warranties and Limitation of Liability. Every effort has been made to ensure the accuracy of the features and techniques presented in this publication. However, Open Text Corporation and its affiliates accept no responsibility and offer no warranty whether expressed or implied, for the accuracy of this publication.

Contents

Chapter 1: Introduction

Refer to the current RightFax user documentation for detailed descriptions of the functions and constants defined in the RightFax API. The RightFax API is intended for experienced C and C++ programmers, and no attempt is made to explain programming techniques, except if they are specific to the use of the RightFax API. Application developers who are not familiar with RightFax should plan to work closely with an experienced RightFax administrator in designing and developing custom applications with the RightFax API.

The RightFax API is distributed as a Windows 32-bit dynamically linked library (DLL). Versions for Windows 16-bit and OS/2 are no longer maintained or supported.

Note that:

- Unless otherwise noted, functions in the RightFax API are non-atomic. Take care that your application does not try to modify the same data from more than one point at the same time.
- Each server handle should only be used by one thread at a time and list handles should only be modified on one thread at a time.
- The library is reentrant, however, and thus completely thread-safe.

The RightFax API is a constantly evolving library, used both internally by RightFax product engineers and externally by RightFax customers. It is updated frequently. The definitive source for the latest information on the API is the Rfapi.h header file. Although this is a standard C/C++ header file, it includes information and comments that all developers using the RightFax API will find useful.

Enabling security for applications created using the API

Most functions in the RightFax API require that users be authenticated. Therefore, after making a call to RFaxCreateServerHandle(), you must make a call to RFaxVerifyUser() or RFaxVerifyUser2() to verify the user and enable security for all subsequent API calls.

If you close the server handle (RFaxCloseServerHandle()) or sign off the user (RFaxUnVerifyUser()), you must re-authenticate the user by calling RFaxVerifyUser() or RFaxVerifyUser2() again before making any other API calls.

The following sample code shows how and when to call one of the RFaxVerfiyUser functions:

```
// RightFax server connection parameters

LPSTR lpServerName = "127.0.0.1"; // RightFax
server name or IP address

LPSTR lpUserId = "Administrator"; // User ID to
connect to server as

LPSTR lpUserPwd = ""; // User password

USHORT uConnProtocol = COMMPROTOCOL_TCPIP; //
Connection protocol used to connect to RightFax

HANDLE rfHandle = NULL;
```

```
DWORD dwRet = 0;
FAXINFO_11 fi11;

// Create a handle to the RightFax server
cout << "Connecting to RightFax server '" <<
lpServerName << "'..." << endl;
rfHandle = RFaxCreateServerHandle3(
lpServerName, // Server name or IP Address
uConnProtocol, // Connection protocol
&dwRet, // Error returned if connection is
unsuccessful
FALSE, // Disable version check?
FALSE, // Use SOAP proxy for RPC calls?
NULL // If using SOAP proxy, the URL to the
SOAP proxy server
);
if ((rfHandle == NULL) || (dwRet != RFAX_
SUCCESS))
{
char szErr[256];

_snprintf_s(szErr, sizeof(szErr), "Unable to
open server handle: %d", dwRet); throw new
exception (szErr);
}
LOGININFO pInfo;
cout << "Verifying user account '" << lpUserId
<< "'..." << endl;
dwRet = RFaxVerifyUser2(
rfHandle,

lpUserId,
lpUserPwd,
TRUE,
&pInfo,
TRUE
);
if (dwRet != RFAX_SUCCESS)
{
cout << "Unable to validate user account '" <<
lpUserId << "': " << dwRet << endl;
RFaxCloseServerHandle(rfHandle);
// Exit
}
//
// Do work here
//
RFaxCloseServerHandle(rfHandle);
```

Chapter 2: Setting functions

The system functions are used for system-wide settings.

RFaxDeleteSetting()

Deletes a system setting.

C Prototype

```
DWORD RFAXAPI RFaxDeleteSetting(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszName
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of the setting to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetSettingList()

This function gets a handle to a list of settings.

C Prototype

```
DWORD RFAXAPI RFaxGetSettingList(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszName,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	0 or setting name to look for Use * for wildcard.
lpLH	On success, returns a handle to a list of SETTINGLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadSetting()

Loads a system setting.

C Prototype

```
DWORD RFAXAPI RFaxLoadSetting(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE pszName,
 OUT PSETTINGINFO_10 psetting
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of the setting to load.
psetting	Returns the SETTINGINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadSetting2() below

RFaxLoadSetting3() below

RFaxLoadSetting2()

Loads a system setting into the specified local file.

C Prototype

```
DWORD RFAXAPI RFaxLoadSetting2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszName,
 IN RFAXCSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of the setting to load.
pszFileName	Full path to file where setting should be written.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadSetting() on the previous page

RFaxLoadSetting3() below

RFaxLoadSetting3()

Loads a system setting into the specified block of memory.

C Prototype

```
DWORD RFAXAPI RFaxLoadSetting3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE pszName,
 OUT RFAXSTR pszBuffer,
 IN USHORT usBufferSize
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of the setting to load.
pszBuffer	Memory where setting should be written.

usBufferSize	Number of bytes available at pszBuffer.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadSetting() on page 22

RFaxLoadSetting2() on the previous page

RFaxSaveSetting()

Saves a system setting.

C Prototype

```
DWORD RFAXAPI RFaxSaveSetting(
 IN RFAXSERVERHANDLE hServer,
 IN PSETTINGINFO_10 psetting
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
psetting	Returns the SETTINGINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveSetting2()

Saves a system setting.

C Prototype

```
DWORD RFAXAPI RFaxSaveSetting2(
 IN RFAXCSTR pszName,
 IN RFAXCSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of setting.
pszFileName	Full path to file containing setting.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveSetting3()

Saves a system setting.

C Prototype

```
DWORD RFAXAPI RFaxSaveSetting3(
 IN RFAXCSTR pszName,
 IN RFAXCSTR pszBuffer
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of setting.
pszBuffer	Setting text.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 3: General server functions

The general server functions are used to connect to a RightFax server, read and write to files, and get lists of files from the RightFax folders.

RFaxCloseServerHandle()

Closes a server handle created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().

C Prototype

```
void RFAXAPI RFaxCloseServerHandle(
 IN RFAXSERVERHANDLE hServer
);
```

Arguments

hServer	Specify the handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

Return Values

None.

Remarks

Closes any open communication channels and frees up memory used by RFaxCreateServerHandle() or *RFaxCreateServerHandle2()* on the next page.

RFaxCreateServerHandle()

Creates a unique server handle to talk to the RightFax server.

C Prototype

```
RFAXSERVERHANDLE RFAXAPI RFaxCreateServerHandle
(
 IN RFAXCSTR lpServerName,
 IN USHORT usCommProtocol,
 OUT OPTIONAL DWORD *lpdwError
);
```

Arguments

lpServerName	Computer name of the RightFax server to communicate with. Do not include the leading backslashes (\\). If COMMPROTOCOL_??? is used for the usCommProtocol argument, then the server name can be specified as a TCP/IP address (x.x.x.x) or DNS host name, e.g. “faxserv1.acme.com.”
usCommProtocol	Specify one of the COMMPROTOCOL_??? constants to indicate the communications protocol that should be used to communicate with the RightFax server.
lpdwError	If the function fails, this will return one of the RFAXERR_??? constants.

Return Values

Returns the server handle to the specified RightFax server.

Remarks

The handle returned by this function is required by almost every API call. If for some reason the handle is not being created, check to make

sure that all of the API files are located in the same folder. The most common reason for a failure is that the RF*.NDR files are not in the same folder as the RFWIN32.DLL. Such a failure will return the error RFAXERR_NDRNOTFOUND (10011).

The act of creating a server handle with RFaxCreateServerHandle does not necessarily initiate communications with the server. In other words, the RFaxCreateServerHandle function will succeed even if the target fax server is not running. The RFaxGetServerInfo() and RFaxGetServerInfo2() functions are a useful means for pinging a server.

Every time a handle is created, it uses some memory. It is important to remember to close server handles, otherwise your application will leak memory. Handles returned by RFaxCreateServerHandle(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), and RFaxCreateServerHandle2() can be used interchangeably.

See Also

RFaxCreateServerHandle2() below

RFaxCreateServerHandle3() on the next page

RFaxCreateServerHandle4() on page 29

RFaxCreateServerHandle4() on page 29

RFaxCreateServerHandle2()

This function creates a unique server handle to the specified RightFax server.

C Prototype

```
RFAXSERVERHANDLE RFAXAPI
RFaxCreateServerHandle2(
 IN RFAXCSTR lpServerName,
 IN USHORT usCommProtocol,
 OUT OPTIONAL DWORD *lpdwError,
 IN BOOL fDisableVersionCheck
);
```

Arguments

lpServerName	Computer name of the RightFax server to communicate with. Do not include the leading backslashes (\\). If COMMPROTOCOL_??? is used for the usCommProtocol argument, then the server name can be specified as a TCP/IP address (x.x.x.x) or DNS host name, e.g. “faxserv1.acme.com”.
usCommProtocol	Specify one of the COMMPROTOCOL_??? constants to indicate the communications protocol that should be used to communicate with the RightFax server.
lpdwError	If the function fails, this will return one of the RFAXERR_??? constants.
fDisableVersionCheck	When set to True, prevents the automatic loading of server information during handle creation. RFaxCreateServerHandle2 will communicate with the fax server to acquire version and build information when this argument is zero, or False. This differs from RFaxCreateServerHandle() where no communications are actually done during the call. By passing a non-zero value in this argument to RFaxCreateServerHandle2, communications are postponed until the returned handle is used in a subsequent API call which, by necessity, must communicate with the server.

Return Values

Returns the server handle to the RightFax server.

Remarks

The handle returned by this function is required by almost every API call. If for some reason the handle is not being created, check to make sure that all of the API files are located in the same folder. The most common reason for a failure is that the RF*.NDR files are not in the

same folder as the RFWIN32.DLL. Such a failure will return the error RFAXERR_NDRNOTFOUND (10011).

Every time a handle is created, it uses some memory. It is important to remember to close your server handles, otherwise your application will leak memory. Handles returned by RFaxCreateServerHandle(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), and RFaxCreateServerHandle2() can be used interchangeably.

See Also

RFaxCreateServerHandle() on page 26

RFaxCreateServerHandle3() below

RFaxCreateServerHandle4() on the next page

RFaxCreateServerHandle4() on the next page

RFaxCreateServerHandle3()

This function creates a unique server handle to the specified RightFax server.

C Prototype

```
RFAXSERVERHANDLE RFAXAPI
RFaxCreateServerHandle3(
 IN RFAXCSTR lpServerName,
 IN USHORT usCommProtocol,
 OUT OPTIONAL DWORD *lpdwError,
 IN BOOL fDisableVersionCheck,
 IN BOOL fUseSoapProxy,
 IN RFAXCSTR lpSoapProxy
);
```

Arguments

lpServerName	Computer name of the RightFax server to communicate with. Do not include the leading backslashes (\\). If COMMPROTOCOL_??? is used for the usCommProtocol argument, then the server name can be specified as a TCP/IP address (x.x.x.x) or DNS host name, e.g. “faxserv1.acme.com”.
usCommProtocol	Specify one of the COMMPROTOCOL_??? constants to indicate the communications protocol that should be used to communicate with the RightFax server.
lpdwError	If the function fails, this will return one of the RFAXERR_??? constants.
fDisableVersionCheck	When set to True, prevents the automatic loading of server information during handle creation. RFaxCreateServerHandle2 will communicate with the fax server to acquire version and build information when this argument is zero, or False. This differs from RFaxCreateServerHandle() where no communications are actually done during the call. By passing a non-zero value in this argument to RFaxCreateServerHandle2, communications are postponed until the returned handle is used in a subsequent API call which, by necessity, must communicate with the server.
fUseSoapProxy	Should all of the RPC calls be proxied through a Soap endpoint?
lpSoapProxy	If fUseSoapProxy is set to True then this value needs to be the URL for the SOAP proxy server.

Return Values

Returns the server handle to the RightFax server.

Remarks

The handle returned by this function is required by almost every API call. If for some reason the handle is not being created, check to make sure that all of the API files are located in the same folder. The most common reason for a failure is that the RF*.NDR files are not in the same folder as the RFWIN32.DLL. Such a failure will return the error RFAXERR_??? (10011).

Every time a handle is created, it uses some memory. It is important to remember to close your server handles, otherwise your application will leak memory. Handles returned by RFaxCreateServerHandle(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), and RFaxCreateServerHandle2() can be used interchangeably.

See Also

RFaxCreateServerHandle() on page 26

RFaxCreateServerHandle2() on page 27

RFaxCreateServerHandle4() below

RFaxCreateServerHandle4() below

RFaxCreateServerHandle4()

This function creates a unique server handle to the specified RightFax server.

C Prototype

```
RFAXSERVERHANDLE RFAXAPI
RFaxCreateServerHandle4(
 IN RFAXCSTR lpServerName,
 IN USHORT usCommProtocol,
 OUT OPTIONAL DWORD *lpdwError,
 IN BOOL fDisableVersionCheck,
 IN BOOL fUseSoapProxy,
 IN RFAXCSTR lpSoapProxy
 IN RFAXCSTR lpLangSelection
);
```

Arguments

lpServerName	Computer name of the RightFax server to communicate with. Do not include the leading backslashes (\\). If COMMPROTOCOL_??? is used for the usCommProtocol argument, then the server name can be specified as a TCP/IP address (x.x.x.x) or DNS host name, e.g. “faxserv1.acme.com”.
usCommProtocol	Specify one of the COMMPROTOCOL_??? constants to indicate the communications protocol that should be used to communicate with the RightFax server.
lpdwError	If the function fails, this will return one of the RFAXERR_??? constants.
fDisableVersionCheck	When set to True, prevents the automatic loading of server information during handle creation. When set to zero or False, RFaxCreateServerHandle4 will communicate with the fax server to acquire version and build information. By passing a non-zero value in this argument to RFaxCreateServerHandle4, communications are postponed until the returned handle is used in a subsequent API call which, by necessity, must communicate with the server.
fUseSoapProxy	Should all of the RPC calls be proxied through a Soap endpoint?
lpSoapProxy	If fUseSoapProxy is set to True then this value needs to be the URL for the SOAP proxy server.
lpLangSelection	Optional string to set the langID to be used for localization of strings.

Return Values

Returns the server handle to the RightFax server.

Remarks

The handle returned by this function is required by almost every API call. If for some reason the handle is not being created, check to make sure that all of the API files are located in the same folder. The most common reason for a failure is that the RF*.NDR files are not in the same folder as the RFWIN32.DLL. Such a failure will return the error RFAXERR_??? (10011).

Every time a handle is created, it uses some memory. It is important to remember to close your server handles, otherwise your application will leak memory. Handles returned by RFaxCreateServerHandle(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), and RFaxCreateServerHandle2() can be used interchangeably.

See Also

RFaxCreateServerHandle() on page 26

RFaxCreateServerHandle2() on page 27

RFaxCreateServerHandle3() on page 28

RFaxCreateServerHandle3() on page 28

RFaxCreateServerHandle3() on page 28

RFaxCreateServerHandle5()

This function creates a unique server handle to the specified RightFax server.

C Prototype

```
RFAXSERVERHANDLE RFAXAPI
RFaxCreateServerHandle4(
 IN RFAXCSTR lpServerName,
 IN USHORT usCommProtocol,
 OUT OPTIONAL DWORD *lpdwError,
 IN BOOL fDisableVersionCheck,
);
```

Arguments

lpServerName	Computer name of the RightFax server to communicate with. Do not include the leading backslashes (\\). If COMMPROTOCOL_??? is used for the usCommProtocol argument, then the server name can be specified as a TCP/IP address (x.x.x.x) or DNS host name, e.g. “faxserv1.acme.com”.
usCommProtocol	Specify one of the COMMPROTOCOL_??? constants to indicate the communications protocol that should be used to communicate with the RightFax server.
lpdwError	If the function fails, this will return one of the RFAXERR_??? constants.
fDisableVersionCheck	When set to True, prevents the automatic loading of server information during handle creation. When set to zero or False, RFaxCreateServerHandle4 will communicate with the fax server to acquire version and build information. By passing a non-zero value in this argument to RFaxCreateServerHandle4, communications are postponed until the returned handle is used in a subsequent API call which, by necessity, must communicate with the server.

Return Values

Returns the server handle to the RightFax server.

Remarks

The handle returned by this function is required by almost every API call. If for some reason the handle is not being created, check to make sure that all of the API files are located in the same folder. The most common reason for a failure is that the RF*.NDR files are not in the same folder as the RFWIN32.DLL. Such a failure will return the error RFAXERR_??? (10011).

Every time a handle is created, it uses some memory. It is important to remember to close your server handles, otherwise your application will

leak memory. Handles returned by RFaxCreateServerHandle(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), RFaxCreateServerHandle2(), and RFaxCreateServerHandle2() can be used interchangeably.

See Also

RFaxCreateServerHandle() on page 26

RFaxCreateServerHandle2() on page 27

RFaxCreateServerHandle3() on page 28

RFaxCreateServerHandle4() on page 29

RFaxDeleteConnection()

This function deletes an external connection.

C Prototype

```
DWORD RFAXAPI RFaxDeleteConnection(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the connection to delete.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadConnection() on page 43

RFaxSaveConnection() on page 47

RFaxDeleteService()

Delete information about a service.

C Prototype

```
DWORD RFAXAPI RFaxDeleteService(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the service information to delete.
ulReserved	Reserved for future use. Pass 0.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

See Also

RFaxLoadService() on page 44

RFaxSaveService() on page 48

RFaxUpdateService() on page 50

RFaxFileCopy()

Copies a file on the RightFax server. The file can be copied to the same or another folder.

C Prototype

```
DWORD RFAXAPI RFaxFileCopy(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogServerSrcDir,
 IN RFAXCSTR pszSrcFile,
 IN unsigned short usLogServerDstDir,
 IN RFAXCSTR pszDstFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogServerSrcDir	Source folder on the RightFax server. Specify one of the LOGDIR_??? constants.
pszSrcFile	Name of the source file in the usLogServerSrcDir folder. Specify the file name in 8.3 format.
usLogServerDstDir	Destination folder on the RightFax server. Specify one of the LOGDIR_??? constants.
pszDstFile	Name of the file to be copied to the usLogServerDestDir folder. If it is different from pszSrcFile, then it will be renamed. Specify the file name in 8.3 format.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Useful for quickly copying a file from one folder on the RightFax server to another folder on the RightFax server without any direct file sharing privileges on the RightFax server. To copy the file within the same folder, specify a name in pszDestFile that is different from the name specified in pszSrcFile.

RFaxFileCopyType()

This function converts an image from a logical directory to a specified format and copies it to the same or another folder.

C Prototype

```
DWORD RFAXAPI RFaxFileCopyType(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogServerSrcDir,
 IN RFAXSTR pszSrcFile,
 IN unsigned short usDestFormat,
 IN unsigned short usLogServerDstDir,
 IN RFAXSTR pszDstFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogServerSrcDir	Source folder on the RightFax server. Specify one of the LOGDIR_??? constants.
pszSrcFile	Name of the source file in the usLogServerSrcDir folder. Specify the file name in 8.3 format.
usDestFormat	Specify one of the FILECOPY_TYPE__??? formats.
usLogServerDstDir	Destination folder on the RightFax server. Specify one of the LOGDIR_??? constants.

pszDstFile	Name of the file to be copied to the usLogServerDestDir folder. If it is different from pszSrcFile, then it will be renamed. Specify the file name in 8.3 format.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Converts an image from a logical directory to a specific format specified by usDestFormat and copies it to the same or another folder.Useful for quickly converting and copying a file from one folder on the RightFax server to another folder on the RightFax server without any direct file sharing privileges on the RightFax server. To copy the file within the same folder, specify a name in pszDestFile that is different from the name specified in pszSrcFile.

See Also

RFaxFileWrite() on page 37

RFaxFileRead() on page 35

RFaxFileDelete() below

RFaxFileDelete()

Deletes a file on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxFileDelete(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogServerSrcDir,
 IN RFAXCSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogServerSrcDir	Source folder of to delete. Specify one of the LOGDIR_??? constants.
pszFileName	Specify the name of the file in the usLogServerSrcDir to delete. Specify the file name in 8.3 format. May contain '*' or '?' wildcards.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

In general, this function should not be used to clean up images associated with a fax, because the RFaxDeleteFax() function will take care of image file cleanup automatically.

See Also

RFaxFileWrite() on page 37

RFaxFileRead() on page 35

RFaxFileGetSpace()

Returns the number of bytes total and free on the physical disk where the logical folder lives.

C Prototype

```
DWORD RFAXAPI RFaxFileGetSpace(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 OUT unsigned long long *pullBytesFree,
 OUT unsigned long long *pullBytesTotal
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	Specify the one of the LOGDIR_??? constants to indicate the folder on the RightFax server to copy from.
pullBytesFree	Number of bytes free on success.
pullBytesTotal	Number of bytes total on success.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxFileList()

Retrieves a list of files from a RightFax folder on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxFileList(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXSTR pszFileSpec,
 IN HWND hWndCallBack,
 IN UINT uiMsg,
 IN WPARAM wParam
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	Folder from which to retrieve a list of files. Specify one of the LOGDIR_??? constants.
pszFileSpec	File search specification. Specify the file in 8.3 format. May contain '*' or '?' wildcards. **Example** “FILE*.301”, “*.DLL”, “B???????.301”
hWndCallBack	Handle of the window to pass the names of the files being listed. See Remarks section below for details.
uiMsg	Message to send to hWndCallback when notifying it of the file names being listed. See Remarks section below for details.
wParam	Value to pass in the wParam of the message sent to hWndCallBack when notifying it of the file names being listed. The use of this value is dictated by the caller and is merely passed along by RFaxFileList to the specified window. See Remarks section below for details.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function enumerates a list of files from a server folder and sends back a Windows message for each file found. The message sent to the window specified by hWndCallBack will use the message ID specified in uiMsg. The wParam of the message is that specified on the call to RFaxFileList. The wParam is normally used by the calling application to get back an object reference (this) or pointer. The lParam of the messages is a pointer to a string containing the file name. Messages are sent back to the application using the SendMessage WIN32 API. Because these messages are sent with

the SendMessage API, your code must be reentrant with respect to the use of RFaxFileList.

See Also

RFaxFileWrite() on page 37

RFaxFileRead() below

RFaxFilePurge()

Purges a file from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxFilePurge(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXCSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	Specify the one of the LOGDIR_??? constants to indicate the folder on the RightFax server to copy from.
pszFileName	Specify the name of the file in the usLogServerSrcDir to purge. Specify the file name in 8.3 format. May contain '*' or '?' wildcards.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxFileRead()

Copies a file from the RightFax server to the local drive.

C Prototype

```
DWORD RFAXAPI RFaxFileRead(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXSTR pszSourceFile,
 IN RFAXSTR pszDestinationDir
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	Specify the one of the LOGDIR_??? constants to indicate the folder on the RightFax server to copy from.
pszSourceFile	Specify the name of the source file on the server to copy. The name must be in 8.3 format. Do not specify a path, because the usLogicalServerDir argument identifies the location of the file on the server.
pszDestinationDir	Specify the folder you want the pszSourceFile to be copied to. This must be relative to the computer executing the API command, and the folder must exist. The file identified by pszSourceFile will be written to this folder.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets files from the RightFax server without any direct file sharing and/or rights to the server.

See Also

RFaxFileWrite() on the next page

RFaxFileCopy() on page 32

RFaxFileDelete() on page 33

RFaxFileRead2()

Copies a file from the RightFax server to the local drive.

C Prototype

```
DWORD RFAXAPI RFaxFileRead2(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXSTR pszSourceFile,
 IN RFAXSTR pszDestinationFileName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	Specify the one of the LOGDIR_??? constants to indicate the folder on the RightFax server to copy from.
pszSourceFile	Specify the name of the source file on the server to copy. The name must be in 8.3 format. Do not specify a path, because the usLogicalServerDir argument identifies the location of the file on the server.
pszDestinationFileName	Specify the file name for the client side file.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets files from the RightFax server without any direct file sharing and/or rights to the server.

See Also

RFaxFileRead() on the previous page

RFaxFileWrite() on the next page

RFaxFileCopy() on page 32

RFaxFileDelete() on page 33

RFaxFileRead3()

Copies a file from the RightFax server to the local drive.

C Prototype

```
DWORD RFAXAPI RFaxFileRead3(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXSTR pszSourceFile,
 IN RFAXSTR pszDestinationPath
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	Specify the one of the LOGDIR_??? constants to indicate the folder on the RightFax server to copy from.
pszSourceFile	Specify the name of the source file on the server to copy. The name must be in 8.3 format. Do not specify a path, because the usLogicalServerDir argument identifies the location of the file on the server.
pszDestinationPath	Full path to the target file (the directory must exist).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets files from the RightFax server without any direct file sharing and/or rights to the server.

See Also

RFaxFileRead() on page 35

RFaxFileRead2() on the previous page

RFaxFileWrite() below

RFaxFileCopy() on page 32

RFaxFileDelete() on page 33

RFaxFileReadWhitelist()

Copies the whitelist.txt file from the RightFax server to the local drive.

C Prototype

```
DWORD RFAXAPI RFaxFileReadWhitelist(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszDestinationDir,
 IN RFAXCSTR pszDestinationFileName,);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszDestinationDir	Specify the folder where you want to copy the whitelist.txt file. This must be relative to the computer executing the API command, and the folder must exist. Do not specify a file name.
pszDestinationFileName	File name for client side file or NULL for the default "whitelist.txt".

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets the whitelist.txt file from the RightFax server without any direct file sharing and/or rights to the server.

Compatibility

Compatible with RightFax versions 16.2 and later.

See Also

RFaxFileWrite() below

RFaxFileCopy() on page 32

RFaxFileDelete() on page 33

RFaxFileWrite()

Copies a file to the RightFax server from the local drive.

C Prototype

```
DWORD RFAXAPI RFaxFileWrite(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXSTR pszDestinationFile,
 IN RFAXSTR pszLocalFileSpec
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

usLogicalServerDir	Specify one of the LOGDIR_??? constants to indicate the folder that the local file will be copied to.
pszDestinationFile	Destination file name in 8.3 format. The file will be placed in the LOGDIR_??? as specified in the usLogicalServerDir argument. If the file name specified here is different from the file specified in pszLocalFileSpec, then the server will rename the file when it copies it.
pszLocalFileSpec	Specify the full path to the local file. The path must be fully qualified and include all subdirectories. You can specify UNC format for a local drive specification.

Return Values

Return	Result
RFAXERR_??? (0)	Success
RFAXERR_???	Failure

Remarks

This function is useful for copying files from the local drive to the RightFax server. The most common use for this function is to copy files to the RightFax\Outgoing folder on the RightFax server so that a fax can be generated. Because this function uses RPCs to copy the files to the RightFax server, no file share rights are necessary. Because the destination directories are specified by constants, such as LOGDIR_??? for the RightFax\Outgoing folder, it is not necessary for the client application to search for those directories on the server. This allows RightFax to change the location of these directories without affecting the client applications.

See Also

RFaxFileRead() on page 35

RFaxFileCopy() on page 32

RFaxFileDelete() on page 33

RFaxGetCollectiveList()

Gets a list of servers in the collective in a Shared Services Environment.

C Prototype

```
DWORD RFAXAPI RFaxGetCollectiveList(
 IN RFAXSERVERHANDLE hServer,
 OUT LPSTR list
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
list	Returns string containing names of servers thought to be 'up.' **Note** Caller must allocate a string buffer for list parameter of at least 200 characters.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with RightFax versions 10.5 and later.

See Also

RFaxGetCollectiveList2() below

RFaxGetCollectiveList2()

Gets a list of servers that are down in the collective in a Shared Services system. The names of the servers are delimited by ^.

C Prototype

```
DWORD RFAXAPI RFaxGetCollectiveList2(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usOption,
 OUT RFAXSTR pszList,
 IN DWORD dwListLen
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
usOption	See COLLECTIVELISTOPT_???
pszList	Returns the list of servers in the Shared Services system.
dwListLen	Specify the number of characters that can be written to pszList.

Return dwListLenValues

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with RightFax versions 10.5 and later.

See Also

RFaxGetCollectiveList() on the previous page

RFaxGetConnectionList()

This function gets a list of external connections.

C Prototype

```
DWORD RFAXAPI RFaxGetConnectionList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel 10,
 IN ULONG ulTypes,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 10.
ulTypes	Masked types of connections. See masks in *CONNECTIONTYPE_??? on page 403*.
*lpLH	If return value is 0, returns a handle to list of CONNECTIONINFO_# structures depending upon usInfoLevel parameter.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadConnection() on page 43

RFaxGetFileList2()

Gets a list of files from the specified server directory.

C Prototype

```
DWORD RFAXAPI RFaxGetFileList2(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usLogicalServerDir,
 IN RFAXCSTR pszFileSpec,
```

```
    IN short sInfoLevel,
    OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLogicalServerDir	One of the LOGDIR_??? constants.
pszFileSpec	File name (8.3 format, relative to logical directory) to list. May contain * and ? wildcards.
sInfoLevel	Currently must be 0.
*lpLH	On success, this argument returns a handle to the list of FILELISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Gets a list of files that match the files specified in the **pszFileSpec** argument from the folder specified in the **usLogicalServerDir** argument.

See Also

RFaxFileRead() on page 35

RFaxGetFullCollectiveList()

Gets a list of servers in a Shared Services environment.

C Prototype

```
DWORD RFAXAPI RFaxGetFullCollectiveList(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
*lpLH	If return == 0, returns a handle to list of SETTINGLISTELEMENT0 structures

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with RightFax versions 10.5 and later.

See Also

RFaxGetCollectiveList2() on page 38

RFaxGetProtocol()

Get the communication protocol being used for this server handle.

C Prototype

```
DWORD RFAXAPI RFaxGetProtocol(
 IN RFAXSERVERHANDLE hServer,
 OUT USHORT* pusCommProtocol
);
```

Arguments

hServer	Specify a handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszServerName	Out: protocol.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxGetSemaphore()

This function creates a semaphore for blocking work on the same task by all members in a Shared Services environment.

C Prototype

```
DWORD RFAXAPI RFaxGetSemaphore(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszSemaphoreName
 IN USHORT usFlags
);
```

Arguments

hServer	Specify a handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszSemaphoreName	Name of the semaphore.
usFlags	See GETSEMAPHOREFLAGS_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The caller must call RFaxReleaseSemaphore() to release this lock. When the process that called this method exits, the semaphore will be released.

RFaxGetServerInfo()

Gets server information from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetServerInfo(
 IN RFAXSERVERHANDLE hServer,
 OUT PSERVERINFO pServerInfo
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
pServerInfo	Returns the SERVERINFO structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

To ensure that the server can interpret your API calls, it is a good idea to check version information before running your API application.

See Also

RFaxGetServerInfo2() on the next page

RFaxGetServerInfo2()

Gets server information from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetServerInfo2(
 IN RFAXSERVERHANDLE hServer,
 OUT PSERVERINFO2 pServerInfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pServerInfo	Returns the SERVERINFO structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

To ensure that the server can interpret your API calls, it is a good idea to check version information before running your API application.

This version of the RFaxGetServerInfo() function returns additional server attributes. RFaxGetServerInfo() is available for backward compatibility.

See Also

RFaxGetServerInfo() on the previous page

RFaxGetServerInfo3()

Gets server information from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetServerInfo3(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 OUT PSERVERINFO2 pServerInfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Specify the level of information to return from the RightFax server. Currently must be 2 or 3.
pServerInfo	Returns the SERVERINFO or SERVERINFO3 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

To ensure that the server can interpret your API calls, it is a good idea to check version information before running your API application.

This version of the RFaxGetServerInfo() function returns additional server attributes. RFaxGetServerInfo() is available for backward compatibility.

See Also

RFaxGetServerInfo() on the previous page

RFaxGetServerName()

This function gets the name of the RightFax server that this handle is currently referencing.

C Prototype

```
DWORD RFAXAPI RFaxGetServerName(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXSTR pszServerName,
 IN DWORD dwSize
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszServerName	Buffer for server name.
dwSize	Number of characters that can be written to pszServerName.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_BADKEY	The specified user cannot be found.

Remarks

RFaxGetServiceList()

Delete information about a service.

C Prototype

```
DWORD RFAXAPI RFaxGetServiceList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN RFSERVICETYPE serviceType,
 IN ULONG ulReserved
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 10.
serviceType	Type of service. See *RFSERVICETYPE_???* on page 474.
ulReserved	Reserved for future use. Pass 0.
lpLH	If return value is 0, returns a handle to list of SERVICEINFO_# structures depending upon usInfoLevel parameter.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

RFaxLoadConnection()

This function loads information about an external connection.

C Prototype

```
DWORD RFAXAPI RFaxLoadConnection(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN USHORT usInfoLevel 10,
 IN ULONG ulReserved,
 OUT RFAXPVOID pinfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	
usInfoLevel	Information level 10.
ulReserved	Reserved for future use. Specify 0.
pinfo	CONNECTIONINFO_#.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxLoadConnectionByKey()

This function loads information about an external connection.

C Prototype

```
DWORD RFAXAPI RFaxLoadConnectionByKey(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN RFAXCSTR pszName
 IN USHORT usInfoLevel 10,
 IN ULONG ulReserved,
 OUT RFAXPVOID pinfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Key by which to find the connection. See *LOADCONN_KEY_???* on page 446.
pszName	Value to find.
usInfoLevel	Information level 10.
ulReserved	Reserved for future use. Specify 0.
pinfo	CONNECTIONINFO_#.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxLoadService()

Load information about a service.

C Prototype

```
DWORD RFAXAPI RFaxLoadService(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT RFAXPVOID pinfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	
usInfoLevel	Information level 10.
ulReserved	Reserved for future use. Pass 0.
pinfo	SERVICEINFO_#.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

See Also

RFaxDeleteService() on page 31

RFaxSaveService() on page 48

RFaxUpdateService() on page 50

RFaxLoadToken()

Gets the latest token, if any, for the given connection.

C Prototype

```
DWORD RFAXAPI RFaxLoadToken(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hConnection,
 IN USHORT usInfoLevel,
 IN USHORT usReserved,
 OUT RFAXPVOID pinfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hConnection	Reserved.
usInfoLevel	Information level 10 for standard reply or11 for large string reply.
usReserved	Reserved for future use. Pass 0.
pinfo	TOKENINFO_#.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

RFaxPingServer()

Pings the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxPingServer(
 IN RFAXCSTR lpServerName,
 IN DWORD dwPingTimeout
);
```

Arguments

lpServerName	Computer name of the RightFax server to ping.
dwPingTimeout	Amount of time to wait for the command.

Return Values

Pings the specified RightFax server.

RFaxPopupCheckMessage()

Logs the user on to the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxPopupCheckMessage(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
```

```
    IN unsigned char* lpNetworkAddress,
    IN long lPopupMessageLen,
    OUT RFAXSTR lpPopupMessageOut
   );
```

Arguments

hServer	Specify the handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to log in.
lpNetworkAddress	Network address of server.
lNetworkAddressLen	Length of lpNetworkAddress.
lpPopupMessageOut	Reason for login failure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxPopupLogin()

Responds to the Login popup for the the RightFax server and logs in the user.

C Prototype

```
DWORD RFAXAPI RFaxPopupLogin(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN unsigned char* lpNetworkAddress,
 IN long lNetworkAddressLen
);
```

Arguments

hServer	Specify the handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to log in.
lpNetworkAddress	Network address of server.
lNetworkAddressLen	Length of lpNetworkAddress.

Return Values

Logs the user into RightFax server.

RFaxPopupLogout()

Logs the user off from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxPopupLogout(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN unsigned char* lpNetworkAddress
);
```

Arguments

hServer	Specify the handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to log in.
lpNetworkAddress	Network address of server.

Return Values

Logs the user off from the RightFax server.

RFaxReleaseSemaphore()

This function releases a semaphore retrieved via RFaxGetSemaphore ().

C Prototype

```
DWORD RFAXAPI RFaxReleaseSemaphore(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszSemaphoreName
);
```

Arguments

hServer	Specify a handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszSemaphoreName	Name of the semaphore.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveConnection()

This function saves information about an external connection.

C Prototype

```
DWORD RFAXAPI RFaxSaveConnection(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel 10,
 IN RFAXPVOID pinfo,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 10.
pinfo	CONNECTIONINFO_# to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadConnection() on page 43

RFaxDeleteConnection() on page 31

RFaxSaveServerInfo()

Save server information to the fax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveServerInfo(
 IN RFAXSERVERHANDLE hServer,
 IN PSERVERINFO pServerInfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
pServerInfo	The SERVERINFO structure containing fields to be saved.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function saves the specified server information to the fax server. Only the following fields from the SERVERINFO structure can be set via this function (the remainder will be ignored):

- szBillDesc1
- szBillDesc2
- sVerifyCodes
- acReqFields
- ulFlags

RFaxSaveService()

Save information about a service.

C Prototype

```
DWORD RFAXAPI RFaxLoadService(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 OUT RFAXPVOID pinfo,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 10.
pinfo	SERVICEINFO_# to save.
ulReserved	Reserved for future use. Pass 0.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

See Also

RFaxDeleteService() on page 31

RFaxLoadService() on page 44

RFaxUpdateService() on page 50

RFaxSetCommunicationOptions()

This function sets communication options for the server.

C Prototype

```
DWORD RFAXAPI RFaxSetCommunicationOptions(
 IN RFAXSERVERHANDLE hServer,
 IN DWORD dwMaxRetries,
 IN DWORD dwRetrySleepInterval
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
dwMaxRetries	Number of times to retry a command when pipes are busy.
dwRetrySleepInterval	Amount of time to sleep between each retry in milliseconds.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

To ensure that the server can interpret your API calls, it is a good idea to check version information before running your API application.

RFaxSetConnectionAttributes()

Sets connection attributes for this module for every connection established after this call.

C Prototype

```
void RFAXAPI RFaxSetConnectionAttributes(
 CONNECTIONATTRIBUTES connectionattributes
);
```

Arguments

connectionattributes	CONNECTIONATTRIBUTES.

See Also

RFaxSetConnectionAttributes2() below

RFaxSetConnectionAttributes2()

Sets connection attributes for a particular connection.

C Prototype

```
DWORD RFAXAPI RFaxSetConnectionAttributes2(
 IN RFAXSERVERHANDLE hServer,
 CONNECTIONATTRIBUTES connectionattributes,
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
connectionattributes	CONNECTIONATTRIBUTES.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSetConnectionAttributes() above

RFaxSetThreadResolution()

The API can now detect concurrent usage of a single connection (RFAXSERVERHANDLE) by multiple threads. This function can be used to specify the method to handle these situations.

C Prototype

```
DWORD RFAXAPI RFaxSetThreadResolution(
 IN RFAXSERVERHANDLE hServer,
 IN ApiThreadResolution threadResolution
);
```

Arguments

hServer	Handle of the RightFax server.
threadResolution	A value of RFAXTHREADING_??? (ApiThreadResolution).

RFaxStringVersion()

Gets server version from the fax server in the form of a string.

C Prototype

```
DWORD RFAXAPI RFaxStringVersion(
 IN USHORT usInfoLevel,
 IN PSERVERINFO2 pServerInfo,
 IN USHORT usReserved,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

usInfoLevel	Must be 2 or 3.
pServerInfo	Pointer to SERVERINFO or SERVERINFO structure.
usReserved	
pszBuff	
usBuffLen	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxGetServerInfo() on page 41

RFaxGetServerInfo2() on page 42

RFaxGetServerInfo3() on page 42

RFaxUpdateService()

Tells the specified service that something has changed causing it to reread configuration.

C Prototype

```
DWORD RFAXAPI RFaxUpdateService(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the service that should update.
ulReserved	Reserved for future use. Pass 0.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

See Also

RFaxDeleteService() on page 31

RFaxLoadService() on page 44

RFaxSaveService() on page 48

Chapter 4: Server status functions

The server status functions provide an array of STATUSVALUEENTRY structures that contain status information and statistics about a specified RightFax server.

RFaxCloseStatusHandle()

Closes a status handle created by RFaxCreateStatusHandle().

C Prototype

```
void RFAXAPI RFaxCloseStatusHandle(
 IN RFAXSTATUSHANDLE hServer
);
```

Arguments

hServer	Specify the handle that was created using RFaxCreateStatusHandle().

Return Values

None.

Remarks

Closes any open communication channels and frees up memory used by RFaxCreateStatusHandle().

RFaxCreateStatusHandle()

Creates a unique server status handle to be used with the status functions described in this chapter.

C Prototype

```
RFAXSTATUSHANDLE RFAXAPI RFaxCreateStatusHandle
(
 IN RFAXCSTR pServerName,
 IN USHORT usCommProtocol,
 OUT OPTIONAL DWORD *pdwError
);
```

Arguments

pServerName	Computer name of the RightFax server to communicate with. Do not include the leading backslashes (\\). If RFSTATPROTOCOL_TCPIP is used for the usCommProtocol argument, then the server name can be specified as a TCP/IP address (x.x.x.x) or DNS host name, e.g. “faxserv1.acme.com”.
usCommProtocol	Specify one of the RFSTATPROTOCOL_??? constants to indicate the communications protocol that should be used to communicate with the RightFax server.
pdwError	If RFaxCreateStatusHandle fails, this will return one of the RFAXSTS_??? constants.

Return Values

Returns the status handle to the RightFax server.

Remarks

The handle returned by this function is required by the API call RFaxStatusGetValues().

Every time a handle is created it uses some memory. It is important to remember to close your handles, otherwise you application will leak memory.

The act of creating a status handle does not necessarily initiate communications with the server. In other words, the RFaxCreateStatusHandle function will succeed even if the target server is not running.

See Also

RFaxCloseStatusHandle() on the previous page

RFaxStatusGetValues()

Closes a status handle created by RFaxCreateStatusHandle().

C Prototype

```
DWORD RFAXAPI RFaxStatusGetValues (
 IN RFAXSTATUSHANDLE hServer,
 IN unsigned long ulCount,
 IN OUT STATUSVALUEENTRY *aValues
);
```

Arguments

hServer	Specify the handle that was created using RFaxCreateStatusHandle().
ulCount	Indicates the number of elements in the array passed in the aValues argument. The array must already be allocated by the caller.
aValues	Pointer to an array of STATUSVALUEENTRY structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The model of status gathering using this API call is as follows:

- Allocate an array of one or more STATUSVALUEENTRY structures.
- Set the ulDataID member of each array element to indicate the needed data.
- Set the ulIndex1 and/or ulIndex2 member of each array element if the ulDataID requires it.
- Call RFaxStatusGetValues with the array

Upon return from RFaxStatusGetValues, each element will be filled with the requested value

The RFaxStatusGetValues API communicates with the RightFax RPC Server module and the RightFax BoardServer module. When a status value with an ID of SVE_BRD_??? is requested, the BoardServer module is called to obtain the value. All other values are obtained through the RPC server module. The appropriate module must be running or started on the target server, otherwise the value cannot be obtained.

Example This example queries a fax server to determine how long the fax server and BoardServer modules have been running:

```
STATUSVALUEENTRY aVals[2];
DWORD dwError;
RFAXSTATUSHANDLE hStat;

memset(aVals, 0, sizeof(aVals)); // initialize
all elements to zero
aVals[0].ulDataID = SVE_FS_SECONDSUP; // time
```

```
fax server has been running
aVals[1].ulDataID = SVE_BRD_SECONDSUP; // time
board server has been running
hStat = RFaxCreateStatusHandle("MYSERVER",
RFSTATPROTOCOL_TCPIP, &dwError);
if (hStat) {
 dwError = RFaxStatusGetValues(hStat, 2,
aVals);
 if (dwError == 0) {
 if (aVals[0].ulError == 0)
 printf("Server has been running for %d
secs\n", aVals[0].ulData);
 else
 printf("Error %d obtaining value\n", aVals
[0].ulError);
 if (aVals[1].ulError == 0)
 printf("BrdSrv has been running for %d
secs\n", aVals[1].ulData);
 else
 printf("Error %d obtaining value\n", aVals
[1].ulError);
 }
 RFaxCloseStatusHandle(hStat);
}
```

Chapter 5: Audit functions

The audit functions are used to get a list of connections from the RightFax server.

RFaxDeleteAudit()

This function sets an audit entry of type Login to AdminLogOut and hides it from the connections view in EFM.

C Prototype

```
DWORD RFAXAPI RFaxDeleteAudit(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hAuditEntry,
 IN USHORT usOptions
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hAuditEntry	Handle of audit entry to delete.
usOptions	See DELETEAUDITS_OPTIONS_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetAuditsList()

This function retrieves a list of all the audits stored on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetAuditsList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usOptions,
 IN BOOL fAscending,
 IN USHORT usOrder,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG* plTotalAudits,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usOptions	See AUDITS_OPTIONS_???.
fAscending	TRUE for ascending; FALSE for descending.
usOrder	Order in which to load the paged records. See AUDITS_USERCONNECTIONORDER_???
lPageIndex	0-based index of page to return.

usPageSize	Number of records per page.
plTotalAudits	When not null, receives total audit records.
lpLH	On success, returns a handle to a list of AUDITLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 6: Fax attribute functions

The fax attribute functions dynamically associate attributes with a fax.

RFaxDeleteFaxTag()

Deletes an attribute associated with the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFaxTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN RFAXCSTR pszName
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax whose attribute will be deleted.
pszName	Name of the attribute to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetFaxTagList()

Get attributes associated with the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxGetFaxTagList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax whose attributes will be listed.
lpLH	0 or a handle to a list oTAGINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadFaxTag()

Loads an attribute associated with the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxLoadFaxTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN RFAXCSTR pszName,
 OUT PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of fax to load attribute from.
pszName	Name of the attribute to load.
lpTI10	Returns the TAGINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveFaxTag()

Save an attribute associated with the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxSaveFaxTag(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNew,
 IN PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNew	TRUE if the attribute is new to the fax referred to in lpTI10 hOwner.
lpTI10	Returns the TAGINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 7: Fax handling functions

The fax handling functions open a specific fax or a list of faxes and manipulate faxes on the RightFax server.

RFaxApproveFax()

Marks a fax as approved or disapproved.

C Prototype

```
DWORD RFAXAPI RFaxApproveFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN RFAXCSTR pszApproverID,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN BOOL bDisapprove
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to mark as approved or disapproved.
pszApproverID	User ID of approver or disapprover. Required.
pszNotes	Notes to associate with this approval or disapproval. 0 or Null for none.
bDisapprove	True to disapprove, False to approve.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

You must specify the user ID of the approver or disapprover. Notes are optional.

See Also

RFaxLoadFax() on page 96

RFaxMarkFax() on page 100

RFaxArchiveFax()

Saves a fax with XML data for use with EDC processing.

C Prototype

```
DWORD RFAXAPI RFaxArchiveFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax
 IN DWORD dwReserved
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to archive.
dwOptions	See ARCHIVEFAXOPT_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The fax with the XML data will be saved to %InstallFolder%\Archive, unless a different location is specified in EDC configuration.

RFaxCombineFaxes()

Combines two or more faxes into one fax.

C Prototype

```
DWORD RFAXAPI RFaxCombineFaxes(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulNumFaxes,
 IN PCOMBINEINFO lpCombineInfo,
 IN RFAXCSTR lpUserID,
 OUT OPTIONAL FAXHANDLE *phNewFaxHandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulNumFaxes	The number of COMBINEINFO structures you specified in the lpCombineInfo argument.
lpCombineInfo	Array of COMBINEINFO structures.
lpUserID	Specify the user ID of the owner of the new fax.
phNewFaxHandle	Returns the handle of the new fax that was generated from the combine.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The total number of pages in the combined fax cannot exceed 999. If the total number of pages exceeds 999, RFAXERR_INVALID_PARAMETER will be returned. Note that unlike RFaxCombineLibDocs(), this function saves and kicks the new fax returning only the handle.

See Also

RFaxCombineFaxes2() below

RFaxCombineFaxes3() on the next page

RFaxCombineFaxes2()

Combines two or more faxes into one fax with the option to add the delegate ID.

C Prototype

```
DWORD RFAXAPI RFaxCombineFaxes2(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulNumFaxes,
 IN PCOMBINEINFO lpCombineInfo,
 IN RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpDelegateID,
 OUT OPTIONAL FAXHANDLE *phNewFaxHandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulNumFaxes	The number of COMBINEINFO structures you specified in the lpCombineInfo argument.
lpCombineInfo	Array of COMBINEINFO structures.
lpUserID	Specify the user ID of the owner of the new fax.
lpDelegateID	Specify the delegate ID of the user's delegate. If null or empty, only the user ID is used.
phNewFaxHandle	Returns the handle of the new fax that was generated from the combine. Only valid if function returns success (0).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The total number of pages in the combined fax cannot exceed 999. If the total number of pages exceeds 999, RFAXERR_INVALID_PARAMETER will be returned. Note that unlike RFaxCombineLibDocs(), this function saves and kicks the new fax returning only the handle.

See Also

RFaxCombineFaxes() on the previous page

RFaxCombineFaxes() on the previous page

RFaxCombineFaxes3()

Combines two or more faxes into one fax and returns the entire FAXINFO_11 structure. Acting user (delegate/admin) is acquired from the login info.

C Prototype

```
DWORD RFAXAPI RFaxCombineFaxes2(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulNumFaxes,
 IN PCOMBINEINFO lpCombineInfo,
 IN RFAXCSTR lpOwnerID,
 OUT OPTIONAL PFAXINFO_11
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulNumFaxes	The number of COMBINEINFO structures you specified in the lpCombineInfo argument.
lpCombineInfo	Array of COMBINEINFO structures.
lpOwnerID	Specify the user ID of the owner of the new fax.
pinfo	Specify the delegate ID of the user's delegate. If null or empty, only the user ID is used.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The total number of pages in the combined fax cannot exceed 999. If the total number of pages exceeds 999, RFAXERR_INVALID_PARAMETER will be returned. Note that unlike RFaxCombineLibDocs(), this function saves and kicks the new fax returning only the handle.

See Also

RFaxCombineFaxes() on page 59

RFaxCombineFaxes() on page 59

RFaxConvertEmailToFaxInfo()

This function converts an email to a fax.

C Prototype

```
DWORD RFAXAPI RFaxConvertEmailToFaxInfo(
 IN RFAXSTR pszEmail,
 OUT FAXINFO_11* pFaxInfo
);
```

Arguments

pszEmail	String containing email address.
pFaxInfo	Address of FAXINFO_11 buffer

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Return Value

Length of email address.

RFaxConvertEmailToPhoneItem()

This function converts an email to a fax.

C Prototype

```
DWORD RFAXAPI RFaxConvertEmailToPhoneItem(
 IN RFAXSTR pszEmail,
 OUT PHONEITEM* pPhoneItem
);
```

Arguments

pszEmail	Character buffer to receive email address
pPhoneItem	See PHONEITEM

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Return Value

Length of email address

RFaxConvertFaxInfoToEmail()

This function converts an email to a fax.

C Prototype

```
DWORD RFAXAPI RFaxConvertFaxInfoToEmail(
 IN FAXINFO_11* pFaxInfo,
 OUT RFAXSTR pszEmail,
 IN DWORD cbEmail
);
```

Arguments

pFaxInfo	Address of FAXINFO_11 structure.
pszEmail	Character buffer to receive email address.
cbEmail	Length of pszEmail buffer passed in.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxConvertPhoneItemToEmail()

This function converts a phone item to email.

C Prototype

```
DWORD RFAXAPI RFaxConvertPhoneItemToEmail(
 IN PHONEITEM* pPhoneItem,
 OUT RFAXSTR pszEmail,
 IN DWORD cbEmail
);
```

Arguments

pPhoneItem	See PHONEITEM.
pszEmail	
cbEmail	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxCopyEnhancedImages()

This function copies all of the images associated with the fax that are referred to in the lpFI10 argument to the folder specified in the lpLocalDir argument.

C Prototype

```
DWORD RFAXAPI RFaxCopyEnhancedImages(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpLocalDir,
 IN OPTIONAL RFAXCSTR lpLocalImageRoot,
 IN RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLocalDir	Specify the destination folder for the images.
lpLocalImageRoot	Specify the destination base file name for the images without an extension or a dot (.). If not specified, the source image names are used.
lpFI	Specify a FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadFax() on page 96

RFaxCopyImages2()

This function copies all of the images associated with the fax that are referred to in the lpFI10 argument to the folder specified in the lpLocalDir argument.

C Prototype

```
DWORD RFAXAPI RFaxCopyImages2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXCSTR lpLocalDir,
 IN OPTIONAL FAXCSTR lpLocalImageRoot,
 IN RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN USHORT usOpts
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLocalDir	Specify the destination folder for the images.
lpLocalImageRoot	Specify the destination base file name for the images without an extension or a dot (.). If not specified, the source image names are used.
lpFI	Specify a FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
usOpts	Use the LOADFAX_??? constants to specify the fax elements to copy (e.g., body and/or cover sheet).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxCreateCVL()

This function creates a local .cvl file. CVL files are used by the fax server to perform many tasks. It relies on the following functions:

RFaxCreateCVLOpen()	Opens the CVL file and prepares to write to it.
RFaxCreateCVLAddAttachment()	Adds an attachment to the CVL file.
RFaxCreateCVLUpload()	Uploads the local CVL to the server.
RFaxCreateCVLClose()	Closes the CVL file.

The pFI parameter to RFaxCreateCVLOpen() can be used by any of the other functions above so the memory pointed to by pFI must not go out of scope or get otherwise deallocated before the RFaxCreateCVLClose() call. And, finally, note that the RFaxCreateCVLClose() function call invalidates the hCVL handle so this should be the last call made using this handle (e.g., RFaxCreateCVLUpload() should be called before RFaxCreateCVLClose().

C Prototype

```
DWORD RFAXAPI RFaxCreateCVL(
 IN RFAXSERVERHANDLE hServer,
 IN OUT RFAXPVOID pFI,
 IN short sFaxInfoLevel,
 IN RFAXCSTR pszLocalPath,
 IN RFAXCSTR pszLocalName,
 IN RFAXCSTR pszRemoteName,
 IN ULONG ulFlags,
 IN RFAXLISTHANDLE hlAttachments
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle2().
pFI	Some fields may be modified by this function. See the note in *Remarks* on the next page.
sFaxInfoLevel	Level of data structure pointed to by pFI (FAXINFO_10 or FAXINFO_11).
pszLocalPath	Local directory in which to create CVL, If NULL or "" then current directory is used.
pszLocalName	Local name (no extension) for CVL, if NULL or "" then unique name is used. See CREATECVLFLAG_LOCALNAMEISCOMPLETE for alternate usage.

pszRemoteName	Remote name (no extension) for CVL, if NULL or "" then unique name is used.
ulFlags	See CREATECVLFLAG_???.
hlAttachments	Pointers to CVL_ATTACHITEM structureCVL_ATTACHITEM

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you specify the UPLOADFILE flag (see CREATECVLFLAG_???, then RFaxCreateLST() will upload the local file to the server. The server file is created in the server's %RightFaxDir%\OUTGOING folder with pszRemoteName as the name (and .cvl as the extension). If pszRemoteName is NULL then a unique name is generated for you.

> **Note** The server file name is always copied into pFI->inputfilename. RFaxCreateCVL deletes the local file after uploading it to the server. If the document is never saved or fails to save after the CVL file is uploaded to the server, the application should delete the server side CVL file (see RFaxFileDelete).

RFaxCreateCVL is a single function solution for generating CVL files. The following lower-level functions are also available for situations where more control is needed:

- RFaxCreateCVLOpen() opens the CVL file and prepares to write to it
- RFaxCreateCVLAddAttachment() adds an attachment to the CVL file
- RFaxCreateCVLUpload() uploads the local CVL to the server
- RFaxCreateCVLClose() closes the CVL file

RFaxCreateCVLAddAttachment()

This function adds the specified attachment to the CVL file. The function is ignored if the attachment is marked as deleted or is native. This is not considered an error.

If an error occurs while writing,the function can be called for additional attachments, but no further attempt is made to write to the file.

C Prototype

```
DWORD RFAXAPI RFaxCreateCVLAddAttachment(
 IN RFAXCVLHANDLE hCVL,
 IN BOOL bUpload,
 IN unsigned short usLogDir,
 IN BOOL bDoDelete,
 IN CVL_PATTACHITEM pAttachment
 );
```

Arguments

hCVL	Handle created by RFaxCreateCVLOpen()
bUpload	TRUE= Upload pAttachment->szFileName to the server folder specified by sLogDir (LOGDIR_NONE is not supported this case). FALSE=Do not upload, in which case sLogDir indicates where the server can find the file. This is ignored for attachments of type CVL_ATTACHTYPE_LIBDOC.

usLogDir	Directory to which to upload the attachment file. If not uploading, then this is where the server should look for the attachment file. • LOGDIR_NONE: pAttachment->szFileName is complete UNC file name • LOGDIR_OUTGOING: pAttachment->szFileName is 'FILE.EXT' and file is already in server's outgoing folder, whatever that may be • LOGDIR_IMAGE: pAttachment->szFileName is 'FILE.EXT' and file is already in server's image folder, whatever that may be. This is ignored for attachments of type CVL_ATTACHTYPE_LIBDOC.
bDoDelete	Add DELETE command to CVL to delete attachment file from server after it has been processed.
pAttachment	Specify the CVL_ATTACHITEM structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Adds a specified attachment to a CVL file.

See Also

RFaxCreateCVLUpload() on the next page

RFaxCreateCVLClose() below

RFaxCreateCVLClose()

This function closes an open CVL file and optionally forces its deletion. The local file will be deleted if any errors occurred while writing or no recipients were written.

.

C Prototype

```
DWORD RFAXAPI RFaxCreateCVLClose(
 IN RFAXCVLHANDLE hCVL,
 IN BOOL bDeleteLocal
 );
```

Arguments

hCVL	Handle created by RFaxCreateCVLOpen()
bDeleteLocal	TRUE=Delete local CVL file

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Closes an ope open CVL file and optionally forces its deletion.

See Also

RFaxCreateCVLOpen() below

RFaxCreateCVLOpen()

This function opens a CVL file and prepares to write to it.

C Prototype

```
DWORD RFAXAPI RFaxCreateCVLOpen(
 IN RFAXSERVERHANDLE hServer,
 IN OUT RFAXPVOID pFI,
 IN short sFaxInfoLevel,
 IN RFAXCSTR pszLocalPath,
 IN RFAXCSTR pszLocalName,
 IN RFAXCSTR pszRemoteName,
 IN ULONG ulFlags,
 OUT RFAXCVLHANDLE* phCVL
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle2().
pFI	Some fields may be modified by this function AND subsequent CVL function calls (see below)
sFaxInfoLevel	Level of data structure pointed to by pFI (FAXINFO_10 or FAXINFO_11)
pszLocalPath	Local directory in which to create CVL, If NULL or "" then current directory is used
pszLocalName	Local name (no extension) for CVL, if NULL or "" then unique name is used. See CREATECVLFLAG_LOCALNAMEISCOMPLETE for alternate usage.
pszRemoteName	Remote name (no extension) for CVL, if NULL or "" then unique name is used.
ulFlags	See CREATECVLFLAG_???.
phCVL	Handle is returned here, pass this to other RFaxCreateCVL()* functions

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you specify the UPLOADFILE flag (see CREATECVLFLAG_???, then RFaxCreateLST will upload the local file to the server. The server file is created in the server's %RightFaxDir%\OUTGOING folder with pszRemoteName as the name (and .cvl as the extension). If pszRemoteName is NULL then a unique name is generated for you.

> **Note** The server file name is always copied into pFI->inputfilename. RFaxCreateCVL deletes the local file after uploading it to the server. If the document is never saved or fails to save after the CVL file is uploaded to the server, the application should delete the server side CVL file (see RFaxFileDelete).

RFaxCreateCVL is a single function solution for generating CVL files. The following lower-level functions are also available for situations where more control is needed:

- RFaxCreateCVLOpen - opens the CVL file and prepares to write to it
- RFaxCreateCVLAddAttachment - adds an attachment to the CVL file
- RFaxCreateCVLUpload - uploads the local CVL to the server
- RFaxCreateCVLClose - closes the CVL file

RFaxCreateCVLUpload()

This function uploads the local CVL to the server. The function must be called before the CVL is closed. After calling this function additional attachments cannot be written to the CVL .

If no attachments were written, then the CVL will not be uploaded but this is NOT considered an error.

If any errors occurred while writing then the upload will fail.

C Prototype

```
DWORD RFAXAPI RFaxCreateCVLUpload(
 IN RFAXCVLHANDLE hCVL
 );
```

Arguments

hCVL	Handle created by RFaxCreateCVLOpen()

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Closes an ope open CVL file and optionally forces its deletion.

RFaxCreateLST()

This function creates a local LST file. LST files are used by the fax server to expand a single job into multiple jobs. These jobs can use different transport types and other options, such as to and from information, bill codes, delay send times, priorities, recipient CSIDs, and unique IDs.

C Prototype

```
RFAXLISTHANDLE RFAXAPI RFaxCreateLST(
 IN RFAXSERVERHANDLE hServer,
 IN OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN RFAXCSTR pszLocalPath
 IN RFAXCSTR pszLocalName,
 IN RFAXCSTR pszRemoteName,
 IN ULONG ulLSTFlags,
 IN RFAXLISTHANDLE hlRecipients
);
```

Arguments

hServer	Server handle created with RFaxCreateServerHandle2().
lpFI	Some fields may be modified by this function. See below.
sFaxInfoLevel	Level of data structure pointed to by lpFI (FAXINFO_10 or FAXINFO_11)
pszLocalPath	Local directory in which to create LST. If none specified, the current directory will be used.
pszLocalName	Local name (no extension) for LST. If none specified, unique name is used. See CREATELSTFLAG_LOCALNAMEISCOMPLETE for alternate usage.
pszRemoteName	Remote name (no extension) for LST. If none specified, unique name is used.
ulLSTFlags	See CREATELSTFLAG_???.
hlRecipients	List of RECIPIENTLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If a job has the FAXFLAG_MULTICODES flag, the fax server will look for a file with a base name of FAXINFO_##.inputfilename and an extension of "LST" in the %RightFaxDir%\OUTGOING folder on the server.

For each recipient in the LST file, the fax server will create a job using values from the original job combined with values from the recipient entry. Once all recipients are expanded into jobs, the LST file is deleted.

Note that RFaxCreateLST will set the FAXFLAG_MULTICODES flag for you if you specify CREATELSTFLAG_UPLOADFILE.

RFaxCreateLST creates a local LST file in the pszLocalPath folder with pszLocalName as the name (and .lst as the extension). If pszLocalPath is NULL then the current directory is used. If pszLocalName is NULL then a temporary unique name is generated for you.

If you specify the CREATELSTFLAG_UPLOADFILE flag, then RFaxCreateLST will upload the local file to the server. The server file is created in the server's %RightFaxDir%\OUTGOING folder with pszRemoteName as the name (and .lst as the extension). If pszRemoteName is NULL then a unique name is generated for you.

Note that the server file name is always copied into lpFI->inputfilename. RFaxCreateLST deletes the local file after uploading it to the server. If the document is never saved or fails to save after the LST file is uploaded to the server, the application should delete the server side LST file (see RFaxFileDelete()).

RFaxCreateLST is a single function solution for generating LST files. The following lower-level functions are also available for situations where more control is needed:

- RFaxCreateLSTOpen() opens the LST file and prepares to write to it.
- RFaxCreateLSTWriteRecipient() writes a recipients to the LST file.
- RFaxCreateLSTUpload() uploads the local LST to the server.
- RFaxCreateLSTClose() closes the LST file.
- RFaxCreateLSTInitRecipient() initializes a recipient structure.

Note that RFaxCreateLST itself relies on these lower-level functions.

Compatibility

Compatible with all RightFax versions.

RFaxCreateLSTClose()

This function closes an open LST file and optionally forces its deletion. The local file will also be deleted if any errors occur while writing or no recipients were written.

C Prototype

```
RFAXLISTHANDLE RFAXAPI RFaxCreateLSTClose(
 IN RFAXLSTFILEHANDLE hLSTFile,
 IN BOOL bDeleteLocal
);
```

Arguments

hLSTFile	Handle created with RFaxCreateLSTOpen().
lpFI	Some fields may be modified by this function. See below.
bDeleteLocal	TRUE=Delete local LST file.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Closes an open LST file and optionally forces its deletion.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateLST() on the previous page

RFaxCreateLSTOpen() on the next page

RFaxCreateLSTInitRecipient()

This function initializes the specified recipient structure to safe values.

C Prototype

```
RFAXLISTHANDLE RFAXAPI
RFaxCreateLSTInitRecipient(
 IN RECIPIENTLISTELEMENT0* pRecipient
);
```

Arguments

pRecipient	Specify the RECIPIENTLISTELEMENT0 structure to initialize.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Initializes the specified recipient structure.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateLST() on page 67

RFaxCreateLSTOpen() below

RFaxCreateLSTWriteRecipient() on the next page

RFaxCreateLSTUpload() on the next page

RFaxCreateLSTClose() on the previous page

RFaxCreateLSTOpen()

This function opens the LST file and prepares to write to it.

C Prototype

```
RFAXLISTHANDLE RFAXAPI RFaxCreateLSTOpen(
 IN RFAXSERVERHANDLE hServer,
 IN OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN RFAXCSTR pszLocalPath
 IN RFAXCSTR pszLocalName,
 IN RFAXCSTR pszRemoteName,
 IN ULONG ulLSTFlags,
 OUT RFAXLSTFILEHANDLE* phCreateLST
);
```

Arguments

hServer	Server handle created with RFaxCreateServerHandle2().
lpFI	Some fields may be modified by this function. See below.
sFaxInfoLevel	Level of data structure pointed to by lpFI (FAXINFO_10 or FAXINFO_11)
pszLocalPath	Local directory in which to create LST. If none specified, the current directory will be used.
pszLocalName	Local name (no extension) for LST. If none specified, unique name is used. See CREATELSTFLAG_LOCALNAMEISCOMPLETE_ for alternate usage.
pszRemoteName	Remote name (no extension) for LST. If none specified, unique name is used.
ulLSTFlags	See CREATELSTFLAG_???.
phCreateLST	Handle is returned here, pass this to other RFaxCreateLST() functions (RFaxCreateLSTWriteRecipient(), RFaxCreateLSTUpload(), RFaxCreateLSTClose (),RFaxCreateLSTInitRecipient())

Return Values

Return	Result
The handle of the list	Success
Null	Failure

Remarks

This function opens the LST file and prepares to write to it.

Compatibility

Compatible with all RightFax versions.

RFaxCreateLSTUpload()

This function uploads the local LST file to the server. The function must be called before the LST is closed. After this function is called, no additional recipients can be written to the LST file.

If any errors occur while writing or no recipients were written, the upload will fail.

C Prototype

```
RFAXLISTHANDLE RFAXAPI RFaxCreateLSTUpload(
 IN RFAXLSTFILEHANDLE hLSTFile,
 IN RECIPIENTLISTELEMENT0* pRecipient
);
```

Arguments

hLSTFile	Handle created with RFaxCreateLSTOpen().
pRecipient	The RECIPIENTLISTELEMENT0 structure.

Return Values

Return	Result
The handle of the list	Success
Null	Failure

Remarks

Uploads the local LST file to the server.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateLST() on page 67

RFaxCreateLSTOpen() on the previous page

RFaxCreateLSTClose() on page 68

RFaxCreateLSTOpen() on the previous page

RFaxCreateLSTWriteRecipient()

This function writes the specified recipient data to the LST file. If the LSTFLAG_DONOTPROCESS flag is set in the recipient's ulLstFlags, then the recipient is NOT written to the file. This is not considered an error, the recipient is simply ignored.

If an error occurs while writing, it is safe to continue calling this function for additional recipients, although no further attempt is made to write to the file.

C Prototype

```
RFAXLISTHANDLE RFAXAPI
RFaxCreateLSTWriteRecipient(
 IN RFAXLSTFILEHANDLE hLSTFile,
 IN RECIPIENTLISTELEMENT0* pRecipient
);
```

Arguments

hLSTFile	Handle created with RFaxCreateLSTOpen().
pRecipient	Specify the RECIPIENTLISTELEMENT0 structure.

Return Values

Return	Result
The handle of the list	Success
Null	Failure

Remarks

Writes the specified recipient data to the LST file.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateLST() on page 67

RFaxCreateLSTOpen() on page 69

RFaxCreateLSTClose() on page 68

RFaxCreateLSTOpen() on page 69

RFaxDeleteFax()

Deletes a fax from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function call will mark for deletion all of the images associated with a specified fax, and it will remove the fax from the fax list. However, this function will not remove the fax record from the database. Because the fax has been marked as deleted, it will not be visible in most client interfaces (such as FaxUtil), and it will not be loaded with any of the RFaxGetFaxList() functions. The RFaxQueryFaxesToCSV(), RFaxEnumAllFaxes(), and RFaxEnumAllFaxes2() functions will still load faxes that have been marked as deleted. Faxes marked for deletion are purged from the fax database based on the purge age defined for the user's group. Purging takes place once every 24 hours during the automatic server maintenance cycle.

See Also

RFaxDeleteFax2() below

RFaxDeleteFax2()

Deletes a fax from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFax2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle
 IN DWORD dwFlags
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the fax to delete or of fax in folder of faxes to delete if dwFlags contains DELETEFAX_FLAGS_FAXESBYFOLDER.
dwFlags	One or more of the DELETEFAX_FLAGS_??? constants.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is the same as the RFaxDeleteFax() function, and you can specify DELETEFAX_FLAGS_??? constants.

See Also

RFaxDeleteFax() on the previous page

RFaxDeleteFilter()

Deletes a filter for a filtered list of faxes.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFilter(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFilter
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFilter	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxGetFiltersList() on page 85

RFaxLoadFilter() on page 99

RFaxSaveFilter() on page 110

RFaxEnumAllFaxes()

Extracts fax information for all users according to arguments specified.

C Prototype

```
DWORD RFAXAPI RFaxEnumAllFaxes(
 IN RFAXSERVERHANDLE hServer,
 IN short sEnumCtrl,
 IN PRFAXQUERYINFO lpQueryInfo,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sEnumCtrl	Specify 0 to start the enumeration, 1 to continue the enumeration, or 2 to end the enumeration.
lpQueryInfo	Specify the RFAXQUERYINFO structure.
lpFI10	Returns a FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used for data extraction and reporting purposes. It differs from the RFaxGetFaxList() functions in a few important ways. First, both active and deleted faxes are returned (this is one of the few methods of retrieving deleted fax records that have not yet been purged). Second, faxes for all users are returned. Third, the function returns a single fax for each call, whereas RFaxGetFaxList() returns all faxes in a single call.

It is critical that any enumeration started with this function be ended with a call where sEnumCtrl equals 2. Without a call to close the enumeration, the data communications channel to the server is held open and, more importantly, a database service thread on the server is held in use. Because the fax server maintains a pool of database threads (up to 15) to service all requests, holding up a thread indefinitely will adversely affect the server. Note that a call to RFaxCloseServerHandle() will implicitly cause an enumeration close, so it is valid to call RFaxCloseServerHandle() during cleanup while handling an error during an enumeration loop. Only one enumeration can be running at any given time for each server handle. If you want to run multiple enumerations at one time, you must create separate server handles for each instance.

Ensure that the number of concurrent enumeration loops using the RFaxEnumAllFaxes series of functions be kept to a minimum. This is important because the functions hold open a communication session and database service thread (on the fax server) to enhance their performance.

See Also

RFaxEnumAllFaxes2() below

RFaxEnumAllFaxes3() on the next page

RFaxEnumAllFaxes4() on the next page

RFaxEnumAllFaxes2()

Enumerates a particular set of faxes filtered by date.

C Prototype

```
DWORD RFAXAPI RFaxEnumAllFaxes2(
 IN RFAXSERVERHANDLE hServer,
 IN short sEnumCtrl,
 IN PRFAXQUERYINFO2 lpQueryInfo,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sEnumCtrl	Specify 0 to start the enumeration, 1 to continue the enumeration, or 2 to end the enumeration.
lpQueryInfo	Specify the RFAXQUERYINFO2 structure.
lpFI10	Returns a FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxEnumAllFaxes() except lpQueryInfo uses RFAXQUERYINFO2, which has more detailed options.

See Also

RFaxEnumAllFaxes() on the previous page

RFaxEnumAllFaxes3() on the next page

RFaxEnumAllFaxes4() on the next page

RFaxEnumAllFaxes3()

Enumerates a particular set of faxes filtered by date.

C Prototype

```
DWORD RFAXAPI RFaxEnumAllFaxes3(
 IN RFAXSERVERHANDLE hServer,
 IN short sEnumCtrl,
 IN PRFAXQUERYINFO3 lpQueryInfo,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sEnumCtrl	Specify 0 to start the enumeration, 1 to continue the enumeration, or 2 to end the enumeration.
lpQueryInfo	Specify the RFAXQUERYINFO3 structure.
lpFI10	Returns a FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxEnumAllFaxes() and RFaxEnumAllFaxes2() except lpQueryInfo uses RFAXQUERYINFO3, which has more detailed options.

See Also

RFaxEnumAllFaxes() on page 72

RFaxEnumAllFaxes2() on the previous page

RFaxEnumAllFaxes4() below

RFaxEnumAllFaxes4()

Enumerates a particular set of faxes filtered by date.

C Prototype

```
DWORD RFAXAPI RFaxEnumAllFaxes4(
 IN RFAXSERVERHANDLE hServer,
 IN short sEnumCtrl,
 IN PRFAXQUERYINFO3 lpQueryInfo,
 IN short sInfoLevel,
 OUT RFAXPVOID lpReturnInfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sEnumCtrl	Specify 0 to start the enumeration, 1 to continue the enumeration, or 2 to end the enumeration.
lpQueryInfo	Specify the RFAXQUERYINFO3 structure.
sInfoLevel	Specify the level of information. Must be 10 or 11.
lpReturnInfo	On success, returns a FAXINFO_10 or FAXINFO_11 structure depending on the sInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to RFaxEnumAllFaxes2() and RFaxEnumAllFaxes3() except the sInfoLevel argument lets you return either a a FAXINFO_10 or FAXINFO_11 structure.

See Also

RFaxEnumAllFaxes() on page 72

RFaxEnumAllFaxes2() on page 73

RFaxEnumAllFaxes3() on the previous page

RFaxFormFromFax()

Creates a form from the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxFormFromFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN SHORT sRemoveLine,
 IN SHORT sSize,
 IN LPCTSTR pszUser,
 IN PFORMLISTELEMENT0 pForm
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax from which to create a form.
sRemoveLine	Specify line removal option: 0 = Normal merge 1 = Form fix-up with blank out TTI 2 = Form fix-up with cut out TTI
sSize	Only used if sRemoveLine > 0. Specify an adjustment on the height of this form: 0 = Don't adjust 1 = Letter 2 = Legal 3 = A4
pszUser	ID of the user motivating this operation. Required.
pForm	A FOLDERLISTELEMENT0. At a minimum the "code" field must be filled in. If the "handle" field has a value, then the existing form identified by that handle is replaced with the new form created from the fax.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If the form list element's "handle" field is a valid handle, the form it refers to will be replaced by the created form. If the handle field is 0 or Null, the form will be created from scratch. The form list element "code" field must be filled in.

RFaxForwardFax()

Forwards or routes a fax from one user to another.

C Prototype

```
DWORD RFAXAPI RFaxForwardFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usFolder,
 IN BOOL bRoute,
 IN OPTIONAL RFAXSTR pszUserList,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN OPTIONAL RFAXCSTR pszOwnerID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the fax to forward.
usReserved	Reserved - currently must be zero.
bRoute	Set to True if you want to perform a route instead of a forward. A route deletes the fax from the originator's fax mailbox. If pszUserList or sUserList is set to Null, this argument must be False.
pszUserList	A list of all of the recipients, separated by commas. Maximum of 128 bytes. If Null is specified, then the function forward to a fax machine.
pszNotes	This argument is only used if pszUserList is used. You can provide notes that will appear in the history of the fax.
pszOwnerID	Only used if pszUserList is provided and bRoute = False.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function can forward a fax from one user to a list of several users, it can route a fax from one user to another user, or it can create a new "empty" fax that contains the body of the original fax.

See Also

RFaxForwardFax2() below

RFaxForwardFax3() on the next page

RFaxForwardFax4() on page 78

RFaxForwardFax5() on page 79

RFaxForwardFax6() on page 80

RFaxForwardFax2()

Forwards or routes a fax from one user to another.

C Prototype

```
DWORD RFAXAPI RFaxForwardFax2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usFolder,
 IN BOOL bNoKick,
 IN BOOL bRoute,
 IN OPTIONAL RFAXSTR pszUserList,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN OPTIONAL RFAXCSTR pszOwnerID,
 OUT OPTIONAL FAXHANDLE *lphNewFax
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to forward.
usReserved	Reserved - currently must be zero.
bNoKick	When this argument is set to True, the phone number is not cleared and the fax is not auto-kicked. Only applies when forwarding to a fax machine.
bRoute	Set to True if you want to perform a route instead of a forward. A route deletes the fax from the originator's fax mailbox. If pszUserList or sUserList is set to Null, this argument must be False.

pszUserList	A list of recipients, separated by commas. Maximum of 128 bytes. If Null is specified then the function will forward to a fax machine.
pszNotes	This argument is only used if pszUserList is used. You can provide notes that will appear in the history of the fax.
pszOwnerID	Only used if pszUserList is provided and bRoute = False.
lphNewFax	Returns the handle of the new fax that is generated. Only valid when forwarding to a fax machine.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function adds flexibility to the forward-to-fax machine option. It gives you the ability to forward the fax without losing any information and to get the handle of the fax back so you can update the fax information.

See Also

RFaxForwardFax() on page 75

RFaxForwardFax3() below

RFaxForwardFax4() on the next page

RFaxForwardFax5() on page 79

RFaxForwardFax6() on page 80

RFaxForwardFax3()

Forwards or routes a fax from one user to another. If pszUserList is longer than 128 characters, use RFaxForwardFax4() instead.

C Prototype

```
DWORD RFAXAPI RFaxForwardFax3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usFolder,
 IN BOOL bNoKick,
 IN BOOL bRoute,
 IN OPTIONAL RFAXSTR pszUserList,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN OPTIONAL RFAXCSTR pszOwnerID,
 OUT OPTIONAL FAXHANDLE *lphNewFax
 IN OPTIONAL SHORT sFaxInfoSize
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to forward.
usReserved	Reserved - currently must be zero.
bNoKick	When this argument is set to True, the phone number is not cleared and the fax is not auto-kicked. Only applies when forwarding to a fax machine.
bRoute	Set to True if you want to perform a route instead of a forward. A route deletes the fax from the originator's fax mailbox. If pszUserList or sUserList is set to Null, this argument must be False.
pszUserList	A list of recipients, separated by commas. Maximum of 128 bytes. If Null is specified then the function will forward to a fax machine.
pszNotes	This argument is only used if pszUserList is used. You can provide notes that will appear in the history of the fax.

pszOwnerID	Only used if pszUserList is provided and bRoute = False.
lphNewFax	Returns the handle of the new fax that is generated. Only valid when forwarding to a fax machine.
sFaxInfoSize	Number of bytes of the data structure pointed to by lpNewFax.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function adds flexibility to the forward-to-fax machine option. It gives you the ability to forward the fax without losing any information and to get the handle of the fax back so you can update the fax information.

See Also

RFaxForwardFax() on page 75

RFaxForwardFax2() on page 76

RFaxForwardFax4() below

RFaxForwardFax5() on the next page

RFaxForwardFax6() on page 80

RFaxForwardFax4()

Forwards or routes a fax from one user to another.

C Prototype

```
DWORD RFAXAPI RFaxForwardFax4(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usFolder,
 IN BOOL bNoKick,
 IN BOOL bRoute,
 IN OPTIONAL RFAXCSTR pszUserList,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN OPTIONAL RFAXCSTR pszOwnerID,
 OUT OPTIONAL FAXHANDLE *lphNewFax
 IN OPTIONAL SHORT sFaxInfoSize
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to forward.
usReserved	Reserved - currently must be zero.
bNoKick	When this argument is set to True, the phone number is not cleared and the fax is not auto-kicked. Only applies when forwarding to a fax machine.
bRoute	Set to True if you want to perform a route instead of a forward. A route deletes the fax from the originator's fax mailbox. If pszUserList or sUserList is set to Null, this argument must be False.
pszUserList	A list of recipients, separated by commas. Can be more than 128 bytes. If Null is specified, the function will forward to a fax machine.
pszNotes	This argument is only used if pszUserList is used. You can provide notes that will appear in the history of the fax.

pszOwnerID	Only used if pszUserList is provided and bRoute = False.
lphNewFax	Returns the handle of the new fax that is generated. Only valid when forwarding to a fax machine.
sFaxInfoSize	Number of bytes of the data structure pointed to by lpNewFax.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function adds flexibility to the forward-to-fax machine option. It gives you the ability to forward the fax without losing any information and to get the handle of the fax back so you can update the fax information.

Use rfaxforwardfax4 rather than rfaxforwardfax3 if the pszuserlist is longer than 128 characters.

See Also

RFaxForwardFax() on page 75

RFaxForwardFax2() on page 76

RFaxForwardFax3() on page 77

RFaxForwardFax5() below

RFaxForwardFax6() on the next page

RFaxForwardFax5()

Forwards or routes a fax from one user to another.

C Prototype

```
DWORD RFAXAPI RFaxForwardFax5(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN USHORT usFolder,
 IN BOOL bNoKick,
 IN BOOL bRoute,
 IN OPTIONAL RFAXCSTR pszUserList,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN OPTIONAL RFAXCSTR pszOwnerID,
 IN OPTIONAL ULONG ulRouteFlags,
 OUT OPTIONAL FAXHANDLE *lphNewFax
 IN OPTIONAL SHORT sFaxInfoSize
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the fax to forward.
usReserved	Reserved - currently must be zero.
bNoKick	When set to True, the phone number is not cleared and the fax is not auto-kicked. Only applies when forwarding to a fax machine.
bRoute	Set to True if you want to perform a route instead of a forward. A route deletes the fax from the originator's fax mailbox. If pszUserList or sUserList is set to Null, this argument must be False.
pszUserList	A list of recipients, separated by commas. If Null is specified then the function will forward to a fax machine.

pszNotes	This argument is only used if pszUserList is used. You can provide notes that will appear in the history of the fax.
pszOwnerID	Only used if pszUserList is provided and bRoute = False.
ulRouteFlags	Only used if bRoute is set to TRUE. See HISTROUTEFLAG_???.
lphNewFax	Returns the handle of the new fax that is generated. Only valid when forwarding to a fax machine.
sFaxInfoSize	Number of bytes of the data structure pointed to by lpNewFax.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function adds flexibility to the forward-to-fax machine option. It gives you the ability to forward the fax without losing any information and to get the handle of the fax back so you can update the fax information.

See Also

RFaxForwardFax() on page 75

RFaxForwardFax2() on page 76

RFaxForwardFax3() on page 77

RFaxForwardFax4() on page 78

RFaxForwardFax6() below

RFaxForwardFax6()

Forwards a fax to any number of users specified by their user handles. Does not forward to fax machines and does not route.

C Prototype

```
DWORD RFAXAPI RFaxForwardFax6(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN USHORT usFolder,
 IN OPTIONAL FAXHANDLE pahUsers,
 IN OPTIONAL LONG lUserCount,
 IN OPTIONAL RFAXCSTR pszNotes,
 IN OPTIONAL RFAXCSTR pszOwnerID,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the fax to forward.
usReserved	Reserved - currently must be zero.
pahUsers	A list o user handles for the intended recipients.
lUserCount	Number of elements in array specified by pahUsers. If 0, then forward to fax machine is assumed.
pszNotes	This argument is only used if pahUsers is provided. 128 bytes max.
pszOwnerID	Only used if pahUsers is provided and bRoute = False.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to forward a fax to users that you specify by their handles. Do not use to route or forward to fax machines.

See Also

RFaxForwardFax() on page 75

RFaxForwardFax2() on page 76

RFaxForwardFax3() on page 77

RFaxForwardFax4() on page 78

RFaxForwardFax5() on page 79

RFaxGetFaxList()

Gets a list of faxes in a specified user's fax mailbox.

C Prototype

```
DWORD RFAXAPI RFaxGetFaxList(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN USHORT usInfoLevel,
 IN USHORT usFolder,
 IN USHORT usFilter,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to get the fax list from.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2). See the lpLH argument and the Remarks section.
usFolder	Specify the FAXFOLDER_??? to load. Specify 0 to load all of the folders.
usFilter	Specify the FAXFILTER_??? constant to filter on. Specify 0 to load all faxes.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_NOTFOUND	The fax list is empty.
RFAXERR_NOT_ENOUGH_MEMORY	No more memory exists to build the list. Try calling RFaxGetFaxList2() specifying a smaller list size.
RFAXERR_INVALID_HANDLE	The hServer argument is invalid. Be sure to call RFaxCreateServerHandle() or RFaxCreateServerHandle2() to obtain a valid RightFax server handle.
RFAXERR_INVALID_PARAMETER	The lpLH argument is Null. Be sure to declare lpLH and pass its address using the address operator (&).

Remarks

This function loads a fax list and can filter on several fields (usFolder and usFilter). This function is useful for obtaining a quick list of all faxes, similar to the display in FaxUtil.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFaxList2() below

RFaxGetFaxList3() on the next page

RFaxGetFaxList4() on the next page

RFaxGetFaxList2()

Retrieves a list of faxes from a specified user's fax mailbox. You can specify the maximum number of items to read into the list.

C Prototype

```
DWORD RFAXAPI RFaxGetFaxList2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN USHORT usInfoLevel,
 IN USHORT usFolder,
 IN USHORT usFilter,
 IN LONG lMaxRead,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to get the fax list from.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2). See the lpLH argument and the Remarks section.
usFolder	Specify the FAXFOLDER_??? to load. Specify 0 to load all of the folders.
usFilter	Specify the FAXFILTER_??? constant to filter on. Specify 0 to load all faxes.
lMaxRead	Specify the maximum number of elements to load. Set to 0 to load all.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxGetFaxList() except you can specify the maximum number of elements to load.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFaxList() on the previous page

RFaxGetFaxList3() on the next page

RFaxGetFaxList4() on the next page

RFaxGetFaxList3()

Retrieves a list of faxes from a specified user's fax mailbox. You can specify the maximum number of items to read into the list.

C Prototype

```
DWORD RFAXAPI RFaxGetFaxList3(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN DWORD dwOptions,
 IN USHORT usInfoLevel,
 IN USHORT usFolder,
 IN USHORT usFilter,
 IN LONG lMaxRead,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to get the fax list from.
dwOptions	Specify a combination of the FAXGETLIST_??? flags by bitwise ORing them together, or specify 0 for none.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2). See the lpLH argument and the Remarks section.
usFolder	Specify the FAXFOLDER_??? to load. Specify 0 to load all of the folders.
usFilter	Specify the FAXFILTER_??? constant to filter on. Specify 0 to load all faxes.
lMaxRead	Specify the maximum number of elements to load. Set to 0 to load all.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxGetFaxList2() except you can specify additional options via FAXGETLIST_??? flags in dwOptions.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFaxList() on page 81

RFaxGetFaxList2() on the previous page

RFaxGetFaxList4() below

RFaxGetFaxList4()

Retrieves a list of faxes from a specified user's fax mailbox. You can specify the maximum number of items to read into the list.

C Prototype

```
DWORD RFAXAPI RFaxGetFaxList4(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN DWORD dwOptions,
 IN USHORT usInfoLevel,
 IN USHORT usFolder,
 IN USHORT usFilter,
 IN LONG lMaxRead,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to get the fax list from.
dwOptions	Specify a combination of the FAXGETLIST_??? flags by bitwise ORing them together, or specify 0 for none.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2). See the lpLH argument and the Remarks section.
usFolder	Specify the FAXFOLDER_??? to load. Specify 0 to load all of the folders.
usFilter	Specify the FAXFILTER_??? constant to filter on. Specify 0 to load all faxes.
lMaxRead	Specify the maximum number of elements to load. Set to 0 to load all.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxGetFaxList2() except you can specify additional options via FAXGETLIST_??? flags in dwOptions.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFaxList() on page 81

RFaxGetFaxList2() on page 82

RFaxGetFaxList3() on the previous page

RFaxGetFilteredFaxList()

Retrieves a filtered list of faxes. You can specify the maximum number of items to read into the list.

C Prototype

```
DWORD RFAXAPI RFaxGetFilteredFaxList(
 IN RFAXSERVERHANDLE hServer,
 IN DWORD dwOptions,
 IN USHORT usInfoLevel,
 IN RFAXFAXFILTER pFilter,
 IN LONG lMaxRead,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
dwOptions	Specify a combination of the FAXGETLIST_??? flags by bitwise ORing them together, or specify 0 for none.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2).
pFilter	Specify the RFAXFAXFILTER to filter on in loading. Specify 0 to load all faxes.
lMaxRead	Specify the maximum number of elements to load. Set to 0 to load all.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxGetFaxList2() except you can specify additional options via FAXGETLIST_??? flags in dwOptions.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetFiltersList()

Loads a filter for a filtered list of faxes.

C Prototype

```
DWORD RFAXAPI RFaxGetFiltersList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usOpt,
 OUT RFAXLISTHANDLE FAR* lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Handle of owning user or NULL for unowned.
usOpt	Reserved for future use. Specify 0.
lpLH	If return value is 0, returns a handle to a list of *FAXFILTERINFO_10* on page 318 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxDeleteFilter() on page 72

RFaxLoadFilter() on page 99

RFaxSaveFilter() on page 110

RFaxGetNewFax()

Gets the oldest unviewed, unprinted, received fax from the specified user's fax box but does not mark it as viewed.

C Prototype

```
DWORD RFAXAPI RFaxGetNewFax(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN OPTIONAL RFAXSTR lpDestDir,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to check for new faxes.
lpDestDir	If this optional argument is specified, then fax will be copied to that folder.
lpFI10	Returns a FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function searches for the oldest unviewed, unprinted, received fax in a user's mailbox. If a fax is found, it is returned in the FAXINFO_10 structure pointed to by the lpFI10 argument.

The handle to the fax will be found in lpFI10 handle. The handle can be used to subsequently mark the fax as viewed [RFaxMarkFax(h,fi10.handle,MARKFAX_VIEWED)] or delete the fax. If one of these two things is not done and RFaxGetNewFax is called again, the same fax will be returned.

Be aware that RFaxGetNewFax is not multi-user safe. That is, if multiple processes or users are calling RFaxGetNewFax against the same mailbox/user, then the same fax may be returned to both processes/users. If you need to dequeue faxes from a mailbox/user, then the RFaxGetNextFax() or RFaxGetNextFax2() should be used.

RFaxGetNextFax()

Gets the oldest unviewed, unprinted, received fax from the specified user's fax box and marks it as viewed.

C Prototype

```
DWORD RFAXAPI RFaxGetNextFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user's fax box to check for new faxes.
lpFI10	Returns the FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxGetNextFax and RFaxGetNextFax2() search for the oldest unviewed, unprinted, received fax in a user's mailbox. If such a fax is

found, then it is returned in the FAXINFO_10 structure pointed to by the lpFI10 argument, and the fax is marked as viewed. The finding and marking operations are atomic and are executed on the fax server, so multiple processes/users calling this function at the same time cannot get the same fax.

See Also

RFaxGetNewFax() on the previous page

RFaxGetNextFax2()

Gets the oldest unviewed, unprinted, received fax from the specified user's fax box and marks it as viewed.

C Prototype

```
DWORD RFAXAPI RFaxGetNextFax2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the ID of the user fax box that you want to check for new faxes.
lpFI10	Returns the FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Same as RFaxGetNextFax() except you can specify the user ID instead of the handle of the user. See remarks under RFaxGetNextFax() and RFaxGetNewFax() for more information.

RFaxGetPagedFaxList()

Gets a paged list of faxes in a specified user's fax mailbox.

C Prototype

```
DWORD RFAXAPI RFaxGetPagedFaxList(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN DWORD dwOptions,
 IN USHORT usInfoLevel,
 IN USHORT usFolder,
 IN USHORT usFilter,
 IN USHORT usOrder,
 IN BOOL fAscending,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG plTotalFaxes,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to get the fax list from.
dwOptions	See FAXGETLIST_??? flags.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2). See the lpLH argument and the Remarks section.

usFolder	Specify the FAXFOLDER_??? to load. Specify 0 to load all of the folders.
usFilter	Specify the FAXFILTER_??? constant to filter on. Specify 0 to load all faxes.
usOrder	Order in which to load the paged records. See the FAXORDER_??? constants.
fAscending	TRUE =ascending; FALSE =descending.
lPageIndex	0-based index of page to return.
usPageSize	Number of faxes per page.
plTotalFaxes	When not null, receives total faxes matching filter for use in paging.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_NOTFOUND	The fax list is empty.
RFAXERR_NOT_ENOUGH_MEMORY	No more memory exists to build the list. Try calling RFaxGetFaxList2() specifying a smaller list size.
RFAXERR_INVALID_HANDLE	The hServer argument is invalid. Be sure to call RFaxCreateServerHandle() or RFaxCreateServerHandle2() to obtain a valid RightFax server handle.
RFAXERR_INVALID_PARAMETER	The lpLH argument is Null. Be sure to declare lpLH and pass its address using the address operator (&).

Remarks

This function loads a fax list and can filter on several fields (usFolder and usFilter). This function is useful for obtaining a quick list of all faxes, similar to the display in FaxUtil.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetPagedFaxList2() below

RFaxGetPagedFaxList2()

Gets a paged list of faxes in a specified user's fax mailbox.

C Prototype

```
DWORD RFAXAPI RFaxGetPagedFaxList2(
 IN RFAXSERVERHANDLE hServer,
 IN GETPAGEDFAXLISTPARAMETERS* parameters
 OUT LONG* plTotalFaxes,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

parameters	
plTotalFaxes	When not null, receives total faxes matching filter for use in paging.
lpLH	If return = 0, returns a handle to list of *FAXLISTELEMENT0* on page 326 structures.

Return Values

Return	Result

Remarks

See Also

RFaxGetPagedFaxList() on page 87

RFaxGetPagedFilteredFaxList()

Retrieves a filtered paged list of faxes. You can specify the maximum number of items to read into the list.

C Prototype

```
DWORD RFAXAPI RFaxGetFilteredFaxList(
 IN RFAXSERVERHANDLE hServer,
 IN DWORD dwOptions,
 IN USHORT usInfoLevel,
 IN RFAXFAXFILTER pFilter,
 IN USHORT usOrder,
 IN BOOL fAscending,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG* plTotalFaxes,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
dwOptions	Specify a combination of the FAXGETLIST_??? flags by bitwise ORing them together, or specify 0 for none.
usInfoLevel	Specify the level of information to return from the RightFax server (0, 1, or 2).
pFilter	Specify the RFAXFAXFILTER to filter on in loading.
usOrder	Order in which to load the paged records. See the FAXORDER_??? constants.
fAscending	TRUE =ascending; FALSE =descending.
lPageIndex	0-based index of page to return.
usPageSize	Number of faxes per page.
plTotalFaxes	When not null, receives total faxes matching filter for use in paging.
lpLH	Returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 structures depending on the value of the usInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxGetFaxList2() except you can specify additional options via FAXGETLIST_??? flags in dwOptions.

This function returns a FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2 function depending on the usInfoLevel argument.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetPagedFaxList()

RFaxGetUserMessage()

This function loads the desktop notification message for a newly received fax.

C Prototype

```
DWORD RFAXAPI RFaxGetUserMessage(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN LONG newFaxTime,
 IN USHORT usOption,
 OUT PUSERMESSAGE pUserMessage
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user's fax box to check for new faxes.
newFaxTime	Always look for new faxes after this time.
usOption	Specifies the message to return. See USERMSG_OPTION__???
pUser Message	Pointer to USERMESSAGE structure containing the content of the message.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxInitFax()

Initializes a new fax record/object based on the user's default values.

C Prototype

```
DWORD RFAXAPI RFaxInitFax(
 IN RFAXSERVERHANDLE hServer,
 IN PFAXINFO_10 lpFI10,
 IN RFAXCSTR lpUserID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI10	Pass in an empty FAXINFO_10 structure.
lpUserID	Pass in the ID of the user for which the fax should be initialized. The user determines the ownership of the fax, the default cover sheet values, and more.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

You should initialize the FAXINFO_10 structure with this function before filling the FAXINFO_10 structure with specific values. The FAXINFO_10 structure is initialized to base values and is populated with the user's default values.

See Also

RFaxInitFax2() below

RFaxInitFax3() on the next page

RFaxInitFax4() on page 93

RFaxInitFax5() on page 94

RFaxInitFax2()

Initializes a new fax based on one of a several possible keys. You can only select one key (specified with the sWhichKey argument) to pass to the RightFax server. The specified key will require a value for its argument, and the arguments for all other keys must be set to 0. If sWhichKey is set to 1, this function is identical to RFaxInitFax().

C Prototype

```
DWORD RFAXAPI RFaxInitFax2(
 IN RFAXSERVERHANDLE hServer,
 IN PFAXINFO_10 lpFI10,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI10	Pointer to the FAXINFO_10 structure.
sWhichKey	Specify the information used to determine ownership of the fax. 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID
hUser	If sWhichKey is 0, then specify the handle of the user ID, otherwise pass 0.
lpUserID	If sWhichKey is 1, then specify the user ID, otherwise pass 0.
lpRouteInfo	If sWhichKey is 2, then specify the routing information, otherwise pass 0.
lpDSNames	If sWhichKey is 3, then specify the Distinguished Name, otherwise pass 0.
ulRoutingCode	If sWhichKey is 4, then specify the Routing Code, otherwise pass 0.
ulSubscriberID	If sWhichKey is 5, then specify the Subscriber ID, otherwise pass 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxInitFax() on the previous page

RFaxInitFax3() on the next page

RFaxInitFax4() on page 93

RFaxInitFax5() on page 94

RFaxInitFax3()

Initializes a new fax based on one of a several possible keys. You can only select one key (specified with the sWhichKey argument) to pass to the RightFax server. The specified key will require a value for its argument, and the arguments for all other keys must be set to 0. If sFaxInfoLevel is set to 10, this function is identical to RFaxInitFax2().

C Prototype

```
DWORD RFAXAPI RFaxInitFax3(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI	Pointer to the FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel.
sFaxInfoLevel	Specify the level of information to return from the RightFax server. This value must be 10 for FAXINFO_10 or 11 for FAXINFO_11.
sWhichKey	Specify the information used to determine ownership of the fax. 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID.
hUser	If sWhichKey is 0, then specify the handle of the user ID, otherwise pass 0.
lpUserID	If sWhichKey is 1, then specify the user ID, otherwise pass 0.
lpRouteInfo	If sWhichKey is 2, then specify the routing information, otherwise pass 0.
lpDSName	If sWhichKey is 3, then specify the Distinguished Name, otherwise pass 0.
ulRoutingCode	If sWhichKey is 4, then specify the Routing Code, otherwise pass 0.
ulSubscriberID	If sWhichKey is 5, then specify the Subscriber ID, otherwise pass 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxInitFax() on page 90

RFaxInitFax2() on the previous page

RFaxInitFax4() on the next page

RFaxInitFax5() on page 94

RFaxInitFax4()

Initializes a new fax based on one of a several possible keys. You can only select one key (specified with the sWhichKey argument) to pass to the RightFax server. The specified key will require a value for its argument, and the arguments for all other keys must be set to 0. If sFaxInfoLevel is set to 10, this function is identical to RFaxInitFax2().

C Prototype

```
DWORD RFAXAPI RFaxInitFax4(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI	Pointer to the FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel.
sFaxInfoLevel	Specify the level of information to return from the RightFax server. This value must be 10 for FAXINFO_10 or 11 for FAXINFO_11.
sWhichKey	Specify the information used to determine ownership of the fax. 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID.
hUser	If sWhichKey is 0, then specify the handle of the user ID, otherwise pass 0.
lpUserID	If sWhichKey is 1, then specify the user ID, otherwise pass 0.
lpRouteInfo	If sWhichKey is 2, then specify the routing information, otherwise pass 0.
lpDSName	If sWhichKey is 3, then specify the Distinguished Name, otherwise pass 0.
ulRoutingCode	If sWhichKey is 4, then specify the Routing Code, otherwise pass 0.
ulSubscriberID	If sWhichKey is 5, then specify the Subscriber ID, otherwise pass 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxInitFax() on page 90

RFaxInitFax2() on page 91

RFaxInitFax3() on the previous page

RFaxInitFax5() on the next page

RFaxInitFax5()

Initializes a new fax based on one of a several possible keys. You can only select one key (specified with the sWhichKey argument) to pass to the RightFax server. The specified key will require a value for its argument, and the arguments for all other keys must be set to 0. If sFaxInfoLevel is set to 10, this function is identical to RFaxInitFax2().

C Prototype

```
DWORD RFAXAPI RFaxInitFax5(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL INT64 biSubscriberID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI	Pointer to the FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel.
sFaxInfoLevel	Specify the level of information to return from the RightFax server. This value must be 10 for FAXINFO_10 or 11 for FAXINFO_11.
sWhichKey	Specify the information used to determine ownership of the fax. 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID.
hUser	If sWhichKey is 0, then specify the handle of the user ID, otherwise pass 0.
lpUserID	If sWhichKey is 1, then specify the user ID, otherwise pass 0.
lpRouteInfo	If sWhichKey is 2, then specify the routing information, otherwise pass 0.
lpDSName	If sWhichKey is 3, then specify the Distinguished Name, otherwise pass 0.
ulRoutingCode	If sWhichKey is 4, then specify the Routing Code, otherwise pass 0.
biSubscriberID	If sWhichKey is 5, then specify the big Subscriber ID, otherwise pass 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxInitFax() on page 90

RFaxInitFax2() on page 91

RFaxInitFax3() on page 92

RFaxInitFax4() on the previous page

RFaxKickFax()

Causes outbound faxes in the “Error-Dropped” (ED:) state to be re-sent (“kicked”). For faxes in other states, this function forces the server to immediately update the fax status.

C Prototype

```
DWORD RFAXAPI RFaxKickFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN OPTIONAL RFAXCSTR lpUserID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to kick.
lpUserID	The ID of the user owning the fax referenced by hFax. If the user ID is not known, pass a Null or an empty string.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

“Kicking” a fax for which a send attempt has previously been made and where the status of the fax is FAXSTAT_DONE_ERROR will cause the fax server to attempt to send the fax one more time. If the fax fails on this subsequent attempt, then another Complete event or Archive event may be generated.

See Also

RFaxGetArchiveEvent() on page 125

RFaxGetCompletionEvent() on page 126

RFaxKickFax2()

Causes outbound faxes in the “Error-Dropped” (ED:) state to be re-sent (“kicked”). For faxes in other states, this function forces the server to immediately update the fax status.

C Prototype

```
DWORD RFAXAPI RFaxKickFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN OPTIONAL RFAXCSTR lpUserID
 IN ULONG ulFlags
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to kick.
lpUserID	The ID of the user owning the fax referenced by hFax. If the user ID is not known, pass a Null or an empty string.
ulFlags	Reserved.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

“Kicking” a fax for which a send attempt has previously been made and where the status of the fax is FAXSTAT_DONE_ERROR will cause the fax server to attempt to send the fax one more time. If the

fax fails on this subsequent attempt, then another Complete event or Archive event may be generated.

See Also

RFaxGetArchiveEvent() on page 125

RFaxGetCompletionEvent() on page 126

RFaxLoadFax()

Loads information about a fax from the fax handle and can copy the associated files to the local hard drive where they can be viewed.

C Prototype

```
DWORD RFAXAPI RFaxLoadFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN OPTIONAL RFAXSTR pszLocalDir,
 IN LPVOID lpReserved1,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of RightFax server created from RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to get information about.
pszLocalDir	If this optional argument is specified, RightFax will copy the file(s) to the local folder specified. The fax will be in RightFax format (.301, .302, etc.) Pass a Null if you do not want to copy the fax.
lpReserved1	Reserved argument. Must be Null.
lpFI10	Returns a pointer to the FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function has two uses. The first is to retrieve information about the fax. You can then display this information to the user or write it to a file. The other use is to copy the files locally so that they can be viewed.

See Also

RFaxLoadFax2() below

RFaxLoadFax3() on the next page

RFaxLoadFax4() on page 98

RFaxLoadFax2()

Loads information about a fax from the fax handle and can copy parts of the fax to the local hard drive where they can be viewed.

C Prototype

```
DWORD RFAXAPI RFaxLoadFax2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN OPTIONAL RFAXCSTR pszLocalDir,
 IN OPTIONAL USHORT usLoadImageOpts,
 IN LPVOID lpReserved1,
 OUT PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of RightFax server created from RFaxCreateServerHandle() or RFaxCreateServerHandle2().

hFax	Handle of the fax to get information about.
pszLocalDir	If this optional argument is specified, RightFax will copy the file(s) to the local folder specified. The fax will be in RightFax format (.301, .302, etc.) Pass a Null if you do not want to copy the fax.
usLoadImageOpts	Use the LOADFAX_??? constants to specify the fax elements to copy (e.g., body and/or cover sheet).
lpReserved1	Reserved argument. Must be Null.
lpFI10	Returns a pointer to the FAXINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to RFaxLoadFax() except that you can specify the LOADFAX_??? constant.

See Also

RFaxLoadFax() on the previous page

RFaxLoadFax3() below

RFaxLoadFax4() on the next page

RFaxLoadFax3()

Loads information about a fax from the fax handle and can copy parts of the fax to a specific location where they can be viewed.

C Prototype

```
DWORD RFAXAPI RFaxLoadFax3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN OPTIONAL RFAXCSTR pszLocalDir,
 IN OPTIONAL USHORT usLoadImageOpts,
 IN ULONG ulReserved,
 OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel
);
```

Arguments

hServer	Handle of the RightFax server created from RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to get information about.
pszLocalDir	If this optional argument is specified, RightFax will copy the file(s) to the local folder specified. The fax will be in RightFax format (.301, .302, etc.) Pass a Null if you do not want to copy the fax.
usLoadImageOpts	Use the LOADFAX_??? constants to specify the fax elements to copy (e.g., body and/or cover sheet).
ulReserved	Reserved argument. Must be Null.
lpFI	Specify a FAXINFO_10 or FAXINFO_11 structure dependent upon the sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to RFaxLoadFax() and RFaxLoadFax2() except you can specify to use either a FAXINFO_10 or FAXINFO_11structure.

See Also

RFaxCopyImages2() on page 62

RFaxLoadFax() on page 96

RFaxLoadFax2() on page 96

RFaxLoadFax4() below

RFaxLoadFax4()

Loads information about a fax from the fax handle and can copy parts of the fax to a specific location where they can be viewed.

C Prototype

```
DWORD RFAXAPI RFaxLoadFax4(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN OPTIONAL RFAXCSTR pszLocalDir,
 IN OPTIONAL RFAXCSTR pszLocalFileRoot,
 IN OPTIONAL USHORT usLoadImageOpts,
 IN OPTIONAL ULONG ulReserved,
 OUT RFAXPVOID lpFI,
 IN short sFaxInfoLevel
);
```

Arguments

hServer	Handle of the RightFax server created from RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the fax to get information about.
pszLocalDir	If this optional argument is specified, RightFax will copy the file(s) to the local folder specified. The fax will be in RightFax format (.301, .302, etc.) Pass a Null if you do not want to copy the fax.
pszLocalFileRoot	Specify the destination base file name for the images without an extension or a dot (.). If not specified, the source image names are used.
usLoadImageOpts	Use the LOADFAX_??? constants to specify the fax elements to copy (e.g., body and/or cover sheet).
ulReserved	Reserved argument. Must be Null.
lpFI	Specify a FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is identical to RFaxLoadFax3(), and you can specify a base file name.

See Also

RFaxCopyImages2() on page 62

RFaxLoadFax() on page 96

RFaxLoadFax2() on page 96

RFaxLoadFax3() on the previous page

RFaxLoadFaxNotes()

Loads the fax notes of a specified fax.

C Prototype

```
DWORD RFAXAPI RFaxLoadFaxNotes(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hNotes,
 OUT PNOTEINFO_10 lpNI10
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hNotes	Handle of the notes you want to load. The handle to the notes for a particular fax is identified by either the FAXINFO_10.faxnote_dba or FAXINFO_11.faxnote_dba member.
lpNI10	Returns the NOTEINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveFaxNotes() on page 110

RFaxLoadFilter()

Loads a filter for a filtered list of faxes.

C Prototype

```
DWORD RFAXAPI RFaxLoadFilter(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhich,
 IN OPTIONAL FAXHANDLE handle,
 IN OPTIONAL RFAXCSTR lpFilterName,
 OUT PFAXFILTERINFO_10 lpFAFI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhich	0 = load by handle (set handle to filter handle). 1 = load by NAME (set handle to owner handle and lpFilterName to filter name).
handle	Handle to filter or owner as specified in sWhich.
lpFilterName	Name of filter as specified in sWhich.
lpFAFI	If successful, returns *FAXFILTERINFO_10* on page 318.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxDeleteFilter() on page 72

RFaxGetFiltersList() on page 85

RFaxSaveFilter() on page 110

RFaxLockFax()

Locks a fax for performing an operation that can only be performed by one user at a time.

C Prototype

```
DWORD RFAXAPI RFaxLockFax(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usLockOptions,
 IN FAXHANDLE hFax,
 OUT RFAXSTR pszUserId,
 IN DWORD dwUserIdLen
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLockOptions	See *LOCKFAXOPTION_???* on page 449.
hFax	Handle of the fax to lock.
pszUserId	If return = RFAXERR_FAXLOCKED, indicates the ID of the user who has the document locked.
dwUserIdLen	Number of characters that can be written to pszUserId.

Return Values

Return	Result
RFAX_SUCCESS	Fax can be locked.
RFAXERR_FAXLOCKED	Fax is locked by another user.

Remarks

See Also

RFaxUnlockFax() on page 124

RFaxMarkFax()

Marks a fax with a flag, such as viewed and printed.

C Prototype

```
DWORD RFAXAPI RFaxMarkFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usMarkCmd
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to mark.
usMarkCmd	Specify one of the MARKFAX_??? constants. Several arguments can be combined together to mark several settings at one time.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxLoadFax() does not mark the fax as viewed. If you are writing an application that is used to view faxes, use RFaxMarkFax() to mark them as viewed.

See Also

RFaxMarkFax2() below

RFaxApproveFax() on page 58

RFaxMarkFax2()

Marks a fax with a flag, e.g., viewed, printed, etc.

C Prototype

```
DWORD RFAXAPI RFaxMarkFax2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usMarkCmd
 IN RFAXCSTR lpUserID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to mark.
usMarkCmd	Specify one of the MARKFAX_??? constants. Several arguments can be combined together to mark several settings at one time.
lpUserID	User ID of the person making the mark.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxLoadFax() does not mark the fax as viewed. If you are writing an application that is used to view faxes, use RFaxMarkFax() to mark them as viewed.

See Also

RFaxApproveFax() on page 58

RFaxMarkFax3()

Marks a fax with a flag, e.g., viewed, printed, etc.

C Prototype

```
DWORD RFAXAPI RFaxMarkFax3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN USHORT usMarkCmd
 IN | OUT OPTIONAL FAXHANDLE phandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to mark.
usMarkCmd	Specify one of the MARKFAX_??? constants. Several arguments can be combined together to mark several settings at one time.
phandle	In: Handle returned by prior call if augmenting a mark. Out: Handle resulting from mark, if any.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxLoadFax() does not mark the fax as viewed. If you are writing an application that is used to view faxes, use RFaxMarkFax() to mark them as viewed.

See Also

RFaxApproveFax() on page 58

RFaxMoveFax()

Moves a fax between the owner's folders.

C Prototype

```
DWORD RFAXAPI RFaxMoveFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFaxHandle,
 IN USHORT usFolder
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFaxHandle	Handle of the fax to move.
usFolder	ID of the destination folder.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function cannot move faxes between users. Use RFaxForwardFax() to move faxes between users.

See Also

RFaxMoveFax2() below

RFaxMoveFaxes() below

RFaxMoveFax2()

Moves a fax to a folder with the specified name under the specified parent. If a folder with the name does not yet exist, it is created.

C Prototype

```
DWORD RFAXAPI RFaxMoveFax2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE usParentFolder,
 IN RFAXCSTR pszFolderName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFaxHandle	Handle of the fax to move.
usParentFolder	ID of parent of destination folder.
pszFolderName	Name of the destination folder.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function cannot move faxes between users. Use RFaxForwardFax() to move faxes between users.

See Also

RFaxMoveFax() on the previous page

RFaxMoveFaxes() below

RFaxMoveFaxes()

Moves the faxes from one folder to another.

C Prototype

```
DWORD RFAXAPI RFaxMoveFaxes(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usSourceFolder,
 IN USHORT usTargetFolder
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Handle of the owner of the faxes to move.
usSourceFolder	Source folder.
usTargetFolder	Destination folder.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function cannot move faxes between users. Use RFaxForwardFax() to move faxes between users.

See Also

RFaxMoveFax() on page 101

RFaxOCRFax()

Performs optical character recognition (OCR) on a specified fax.

C Prototype

```
DWORD RFAXAPI RFaxOCRFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
| IN SHORT sStartPage,
 IN SHORT sEndPage,
 IN SHORT sLayout,
 IN SHORT sFormat,
 IN RFAXSTR pszOutputFile,
 IN RFAXSTR pszUserID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to OCR.
sStartPage	Specify page to begin OCR. First page is 1.
sEndPage	Specify last page to OCR. Use 0 to indicate all pages.
sLayout	Specify a single OCRLAYOUT_??? flag.
sFormat	Specify a single OCRFORMAT_??? flag.
pszOutputFile	File name specified in 8.3 format. Resulting file is stored in BFT folder on server. This argument is required.
pszUserID	ID of user performing OCR. This argument is required.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadFax() on page 96

RFaxPrintFax()

Prints a specified fax.

C Prototype

```
DWORD RFAXAPI RFaxPrintFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN AUTOPRINTREQ *pPrintInfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to print.
pPrintInfo	A pointer to an AUTOPRINTREQ structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadFax() on page 96

RFaxQueryFaxesToCSV()

Creates a text file containing comma-separated values for the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxQueryFaxesToCSV(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpCSVFileName,
 IN PRFAXQUERYINFO lpQueryInfo,
 OUT long *lpRecordsWritten
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpCSVFileName	Specify the name of the file that will be generated by the fax server.
lpQueryInfo	Specify the RFAXQUERYINFO structure.
lpRecordsWritten	Return the number of records written.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function creates a text file containing comma-separated values (CSV). Integer values are not bounded by quote marks, although all string values are (because string values can contain commas)

RFaxQueryFaxesToCSV() Text File Fields

Field	Type	Max Length	Notes
owner_id	string	21	User's RightFax ID
to_faxnum	string	31	To fax number
to_contactnum	string	31	To voice number
to_name	string	59	To name
to_company	string	59	To company name
to_citystate	string	59	To city/state
from_name	string	59	From name
from_phonenum	string	31	From voice number
billinfo1	string	15	Bill info 1
billinfo2	string	15	Bill info 2
unique_id	string	15	Unique ID of the fax
faxdidnum	string	31	From fax number
operatornum	string	31	From company voice number

Field	Type	Max Length	Notes
generalfaxnum	string	31	From company fax number
remoteid	string	21	CSID of remote fax machine
send_time	int	N/A	Total elapsed time on fax card, in seconds
fax_status	int	N/A	Status of the fax (see FAXSTAT_???)
fax_error_code	int	N/A	Error code of the fax
numpages	int	N/A	Includes optional cover sheet, if any
finemode	int	N/A	0 = normal, 1 = fine
received	int	N/A	0 = sent, 1 = received
faxdate	string	8	Date fax record last modified (MM/DD/YYYY)
faxtime	string	5	Time fax record last modified (HH:MM)

RFaxQueryFaxesToCSV2() can put a status window during the query if a parent window is specified.

RFaxQueryFaxesToCSV2()

Creates a text file containing comma-separated values for the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxQueryFaxesToCSV2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpCSVFileName,
 IN PRFAXQUERYINFO lpQueryInfo,
 IN OPTIONAL HWND hWndParent,
 OUT long *lpRecordsWritten
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpCSVFileName	Specify the name of the file that will be generated by the fax server.
lpQueryInfo	Specify the RFAXQUERYINFO structure.
hWndParent	Handle to the window that will be used as a parent to a progress message dialog box. Such a status message will appear if a valid window handle is passed in this argument. A value of 0 or Null will ensure that no message will appear.
lpRecordsWritten	Return the number of records written.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function creates a text file containing comma-separated values (CSV). Integer values are not bounded by quote marks, though all string values are (because string values can contain commas)

RFaxQueryFaxesToCSV2() Text File Fields

Field	Type	Max Length	Notes
owner_id	string	21	User's RightFax ID
to_faxnum	string	31	To fax number
to_contactnum	string	31	To voice number
to_name	string	59	To name

Field	Type	Max Length	Notes
to_company	string	59	To company name
to_citystate	string	59	To city/state
from_name	string	59	From name
from_phonenum	string	31	From voice number
billinfo1	string	15	Bill info 1
billinfo2	string	15	Bill info 2
unique_id	string	15	Unique ID of the fax
faxdidnum	string	31	From personal fax number
operatornum	string	31	From company voice number
generalfaxnum	string	31	From company fax number
remoteid	string	21	CSID of remote fax machine
send_time	int	N/A	Total elapsed time on fax card, in seconds
fax_status	int	N/A	Status of the fax (see FAXSTAT_???)
fax_error_code	int	N/A	Error code of the fax
numpages	int	N/A	Includes optional cover sheet, if any
finemode	int	N/A	0 = normal, 1 = fine
received	int	N/A	0 = sent, 1 = received
faxdate	string	8	Date fax record last modified (MM/DD/YYYY)
faxtime	string	5	Time fax record last modified (HH:MM)

See Also

RFaxQueryFaxesToCSV() on page 104

RFaxQueryFaxLock()

Query whether a fax is locked.

C Prototype

```
DWORD RFAXAPI RFaxQueryFaxLock(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usReserved,
 IN FAXHANDLE hFax,
 OUT RFAXSTR pszUserId,
 IN DWORD dwUserIdLen
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usReserved	Reserved for future use. Specify 0.
hFax	Handle of the fax.
pszUserId	If return = RFAXERR_FAXLOCKED, indicates the ID of the user who has the document locked.
dwUserIdLen	Number of characters that can be written to pszUserId.

Return Values

Return	Result
RFAX_SUCCESS	Fax is locked.

Remarks

See Also

RFaxUnlockFax() on page 124

RFaxLockFax() on page 99

RFaxSaveFax()

Saves a new fax to be sent or processed by the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveFax(
 IN RFAXSERVERHANDLE hServer,
 IN PFAXINFO_10 lpFI10,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 OUT FAXHANDLE *phNewFaxHandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI10	Stores the FAXINFO_10 structure in a new fax.
lpNI10	Stores the NOTEINFO_10 structure in a new fax. If not specified, then the fax will not contain any notes.
phNewFaxHandle	Returns the handle of the new fax.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function can force the saved fax to send. Initiating a fax, however, is more easily accomplished with other functions such as RFaxSendCVL() or by sending data directly to the HPFAX queue on the fax server.

This function can be used to update received faxes in the fax database. Be careful not to save faxes that are in a conversion or sending status. This can cause communication difficulties between the API application and the fax server.

See Also

RFaxInitFax() on page 90

RFaxSaveFax2() below

RFaxSaveFax3() on the next page

RFaxSaveFax4() on the next page

RFaxSaveFax5() on page 109

RFaxSaveFax2()

Saves a new fax to be sent or processed by the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveFax2(
 IN RFAXSERVERHANDLE hServer,
 IN PFAXINFO_10 lpFI10,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 OUT FAXHANDLE *phNewFaxHandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI10	Stores the FAXINFO_10 structure in a new fax.
lpNI10	Stores the NOTEINFO_10 structure in a new fax. If not specified, then the fax will not contain any notes.
phNewFaxHandle	Returns the handle of the new fax.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveFax() on the previous page

RFaxSaveFax3() on the next page

RFaxSaveFax4() below

RFaxSaveFax5() on the next page

RFaxSaveFax3()

Saves a new fax to be sent or processed by the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveFax3(
 IN RFAXSERVERHANDLE hServer,
 IN PFAXINFO_10 lpFI10,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 OUT FAXHANDLE *phNewFaxHandle,
 IN OPTIONAL USHORT usSaveOpt1,
 IN OPTIONAL USHORT usSaveOpt2,
 IN OPTIONAL USHORT usSaveOpt3,
 IN OPTIONAL USHORT usSaveOpt4,
 IN OPTIONAL USHORT usSaveOpt5
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI10	Stores the FAXINFO_10 structure in a new fax.
lpNI10	Stores the NOTEINFO_10 structure in a new fax. If not specified, then the fax will not contain any notes.
phNewFaxHandle	Returns the handle of the new fax.
usSaveOpt1	Use the SAVEFAX_??? constants to specify save options.
usSaveOpt2	Use the SAVEFAX_??? constants to specify the information to save.
usSaveOpt3	Use the SAVEFAX_??? constants to specify save options.
usSaveOpt4	Reserved. Set to 0.
usSaveOpt5	Reserved. Set to 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveFax() on page 106

RFaxSaveFax2() on the previous page

RFaxSaveFax3() above

RFaxSaveFax4() below

RFaxSaveFax5() on the next page

RFaxInitFax() on page 90

RFaxSendCVL() on page 111

RFaxSaveFax4()

Saves a new fax to be sent or processed by the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveFax4(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 OUT FAXHANDLE *phNewFaxHandle,
 IN OPTIONAL USHORT usSaveOpt1,
 IN OPTIONAL USHORT usSaveOpt2,
 IN OPTIONAL USHORT usSaveOpt3,
 IN OPTIONAL USHORT usSaveOpt5
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI	Specify a FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
lpNI10	Stores the NOTEINFO_10 structure in a new fax. If it is not specified, then the fax will not contain any notes.
phNewFaxHandle	Returns the handle of the new fax.
usSaveOpt1	Use the SAVEFAX_??? constants to specify save options.
usSaveOpt2	Use the SAVEFAX_??? constants to specify the information to save.
usSaveOpt3	Use the SAVEFAX_??? constants to specify save options.
usSaveOpt5	Reserved. Set to 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveFax() on page 106

RFaxSaveFax2() on page 107

RFaxSaveFax3() on the previous page

RFaxSaveFax5() below

RFaxInitFax() on page 90

RFaxSendCVL() on page 111

RFaxSaveFax5()

Saves a new fax to be sent or processed by the RightFax sever.

C Prototype

```
DWORD RFAXAPI RFaxSaveFax5(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 IN OPTIONAL FAXHANDLE hForwardedFax
 OUT FAXHANDLE *phNewFaxHandle,
 IN OPTIONAL USHORT usSaveOpt1,
 IN OPTIONAL USHORT usSaveOpt2,
 IN OPTIONAL USHORT usSaveOpt3,
 IN OPTIONAL USHORT usSaveOpt5
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpFI	Specify a FAXINFO_10 or FAXINFO_11 structure dependent upon sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
lpNI10	Stores the NOTEINFO_10 structure in a new fax. If it is not specified, then the fax will not contain any notes.
hForwardedFax	Returns the handle of the fax if it was forwarded from another fax.
phNewFaxHandle	Returns the handle of the new fax.
usSaveOpt1	Use the SAVEFAX_??? constants to specify save options.
usSaveOpt2	Use the SAVEFAX_??? constants to specify the information to save.

usSaveOpt3	Use the SAVEFAX_??? constants to specify save options.
usSaveOpt5	Reserved. Set to 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveFax() on page 106

RFaxSaveFax2() on page 107

RFaxSaveFax3() on page 108

RFaxSaveFax4() on page 108

RFaxSaveFax5() on the previous page

RFaxInitFax() on page 90

RFaxSendCVL() on the next page

RFaxSaveFaxNotes()

Saves the fax notes of a specified fax.

C Prototype

```
DWORD RFAXAPI RFaxSaveFaxNotes(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewList,
 IN PNOTEINFO_10 lpNI10,
 OUT OPTIONAL FAXHANDLE *lpNewHandle
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewList	Set to True to create a new note record to store the notes.
lpNI10	Specify the NOTEINFO_10 structure including the fax handle and the notes to be saved.
lpNewHandle	An optional pointer to a fax handle to be filled with the handle to the notes.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadFaxNotes() on page 98

RFaxSaveFilter()

Saves a filter for a filtered list of faxes.

C Prototype

```
DWORD RFAXAPI RFaxSaveFilter(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewFAFI,
 IN PFAXFILTERINFO_10 lpFAFI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

fNewFAFI	
lpFAFI	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxDeleteFilter() on page 72

RFaxGetFiltersList() on page 85

RFaxLoadFilter() on page 99

RFaxSendSecureDocSingleDest()

Sends Secure Docs to a single destination.

C Prototype

```
DWORD RFAXAPI RFaxSendSecureDocSingleDest(
 IN RFAXSERVERHANDLE hServer,
 IN FAXINFO_11 pFaxInfo,
 IN DWORD flagsSecureDocs
 IN RFAXSTR pszFrom,
 IN SD_ATTACHITEM* pSDAttachments,
 IN DWORD dwNumAttachments,
 IN OPTIONAL RFAXSTR pszPassword,
 IN OPTIONAL RFAXSTR pszSubject,
 OUT FAXHANDLE phFax
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pFaxInfo	Fax record. See FAXINFO_11
flagsSecureDocs	See SECURE_DOCS_???
pszFrom	Email address of the sender.
pSDAttachments	An array of SD_ATTACHITEM.
dwNumAttachments	Number of items in psDAttachments.
pszPassword	Password used to encrypt files.
pszSubject	Subject of the new email message.
phFax	Handle to the new fax record.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxCreateServerHandle() on page 26

RFaxCreateServerHandle2() on page 27

RFaxSendCVL()

Sends a conversion list to the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSendCVL(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpCVLFileName,
 IN PFAXINFO_10 lpFI10,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 IN OPTIONAL RFAXSTR lpFCSFileName,
 OUT FAXHANDLE *phNewFaxHandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpCVLFileName	Name of the CVL file from which RightFax will process and generate a fax.
lpFI10	Pass the FAXINFO_10 structure with appropriate information filled out. Note that FFRFLAGS_NEEDS_PRESCAN cannot be used.
lpNI10	If you want notes to appear on the cover sheet, include a pointer to a NOTEINFO_10 structure here.
lpFCSFileName	If you want a cover sheet to appear on the fax, include the name of the cover sheet here in the form of FILENAME.PCL. To use the user's default cover sheet, send an empty string and make sure the FAXFLAG_NOCOVER is not set. To not send a cover sheet, send a value of Null.
phNewFaxHandle	Returns the handle to the new fax created.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The lpFCSFileName must be Null for no cover sheet, or it must point to a file name in the form FILE.PCL. The file name must always end with the extension .pcl, and the file specified should already exist in the \RightFax\FCS folder. If you are not sure, use FCS.pcl, which is the default cover sheet name. A user's specific cover sheet is stored in the USERINFO_10.fcsmodelname field. You can get a USERINFO_10 structure for a specified user ID or handle by using the RFaxLoadUser() function.

CVL Files

CVL files are text files with a single command per line. Each line is terminated with a carriage return/line feed pair. Each line contains one of the following commands:

- PCL *SymbolicDir FileName*
- PS *SymbolicDir FileName*
- DEL *SymbolicDir FileName*
- COPY *SymbolicDir FileName*

The PCL or PS command will cause the fax server to convert a PCL file, text file, or PostScript file into a RightFax group-3 fax page. The DEL command will remove the specified file (only if the file is not marked as read-only). The COPY command is similar to the PCL command except that it will handle a variety of file types: PCL, text, ASCII, PostScript, PCX, DCX, RightFax group-3, and TIFF-G3.

The *SymbolicDir* argument can be one of three possible values:

- Out
- Image
- None

The Out value will cause the server to look for FileName in the RightFax\Outgoing folder. You do not have to specify the exact location of this folder. The value Image is similar except it indicates that the file named by FileName is to be found in the RightFax\Image folder. In both these cases, the FileName argument is assumed to be in the form “FILE.EXT.” No drive or path is specified. If the target FileName is not in the RightFax\Outgoing or RightFax\Image directories, then the None value should be used for the *SymbolicDir* and the FileName argument should be a complete UNC path identifying the target file and its location.

Example

```
PCL OUT PCL255.TMP
DEL OUT PCL255.TMP
COPY NONE \\server3\ibex\images\29949.TIF
```

FAXINFO_10 fields

All of these fields must be completed. Initialize structure to all zeros.

Field	Description
owner_id	RightFax user ID to which to assign this fax.
papernum	Form ID if form overlay is desired (-1 if no form desired).
finemode	Set to 0 for normal fax resolution (200 x 100) or 1 for fine fax resolution (200 x 200).
fr_flags	Initialize to 0L and flip on one or more of these flags: FAXFLAG_AUTODELETE FAXFLAG_AUTODELETEALL FAXFLAG_FINECOVER
to_faxnum	Phone number to which to send fax.
to_contactnum	Info only, goes on cover sheet.
to_name	Info only, goes on cover sheet.
to_company	Info only, goes on cover sheet.
to_citystate	Info only, goes on cover sheet.
from_name	Info only, goes on cover sheet.
from_phonenum	Info only, goes on cover sheet.
billinfo1	Info only.
billinfo2	Info only.
faxdidnum	Info only, goes on cover sheet.
operatornum	Info only, goes on cover sheet.
generalfaxnum	Info only, goes on cover sheet.
call_back	Info only, goes on cover sheet.
hold_for_preview	Set to 1 to force fax to be held for preview, else set to 0.
unique_id	Info only.
ucPriority	Set to: 0 = Normal 1 = Low 2 = High
usSendChannel	Set to a specific channel number, or set to 0 for any channel.

See Also

RFaxSendCVL2() on the next page

RFaxInitFax() on page 90

RFaxSaveFax() on page 106

RFaxSendCVL2()

Sends a conversion list to the RightFax server. This function lets you use the FAXINFO_11 information level for specification of an expiration value.

C Prototype

```
DWORD RFAXAPI RFaxSendCVL2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpCVLFileName,
 IN RFAXPVOID lpFI,
 IN short sFaxInfoLevel,
 IN OPTIONAL PNOTEINFO_10 lpNI10,
 IN OPTIONAL RFAXSTR lpFCSFileName,
 OUT FAXHANDLE *phNewFaxHandle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpCVLFileName	Name of the CVL file from which RightFax will process and generate a fax. For a description of CVL files used with this function, see CVL Files.
lpFI	For a list of fields which must be filled in, see FAXINFO_10.
sFaxInfoLevel	The level of data structure pointed to by lpFI (must be either 10 or 11)
lpNI10	If you want notes to appear on the cover sheet, include a pointer to a NOTEINFO_10 structure here.
lpFCSFileName	If you want a cover sheet to appear on the fax, include the name of the cover sheet here in the form of FILENAME.PCL. To use the user's default cover sheet, send an empty string and make sure the FAXFLAG_NOCOVER is not set. To not send a cover sheet, send a value of Null.
phNewFaxHandle	If function returns 0 (zero), this argument returns the handle to the new fax created.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The lpFCSFileName must be Null for no cover sheet, or it must point to a file name in the form FILE.PCL. The file name must always end with the extension .pcl, and the file specified should already exist in the \RightFax\FCS folder. If you are not sure, use FCS.pcl, which is the default cover sheet name. A user's specific cover sheet is stored in the USERINFO_10.fcsmodelname field. You can get a USERINFO_10 structure for a specified user ID or handle by using the RFaxLoadUser() function.

See Also

RFaxSendCVL() on page 111

RFaxInitFax() on page 90

RFaxSaveFax() on page 106

RFaxStringTrxStatus()

Returns a string version of the fax transmission status from a history list element.

C Prototype

```
DWORD RFAXAPI RFaxStringTrxStatus(
 IN PHISTLISTELEMENT0 pHist,
 OUT RFAXSTR pszBuffer,
 IN USHORT usBuffLen
);
```

Arguments

pHist	Specify HISTLISTELEMENT0 structure.
pszBuffer	Returns the string version of the transmission history.
usBuffLen	Specify the length of the buffer.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Lets you quickly get a string version of the history without having to break apart the HISTLISTELEMENT0. RFaxStringTrxStatus2() uses the information in the specified FAXLISTELEMENT0 or FAXLISTELEMENT1 to get more detailed status information.

See Also

RFaxStringTrxStatus2() below

RFaxStringTrxStatus3() on the next page

RFaxStringTrxStatus4() on the next page

RFaxStringTrxStatus2()

Returns a string version of the fax transmission status from a history list element.

C Prototype

```
DWORD RFAXAPI RFaxStringTrxStatus2(
 IN PHISTLISTELEMENT0 pHist,
 IN RFAXPVOID pFaxListElement,
 IN short sFaxInfoLevel,
 OUT RFAXSTR pszBuffer,
 IN USHORT usBuffLen
);
```

Arguments

pHist	Specify HISTLISTELEMENT0 structure.
pFaxListElement	Specify a FAXLISTELEMENT0 or FAXLISTELEMENT1 structure dependent upon the sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
pszBuffer/sBuffer	Returns the string version of the transmission history.
usBuffLen	Specify the length of the buffer for the string.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Lets you quickly get a string version of the history without having to break apart the HISTLISTELEMENT0. Uses the information in the specified FAXLISTELEMENT0 or FAXLISTELEMENT1 to get more detailed status information.

See Also

RFaxGetHistoryList() on page 139

RFaxStringFaxStatus() on the next page

RFaxStringTrxStatus() on the previous page

RFaxStringTrxStatus3() below

RFaxStringTrxStatus4() below

RFaxStringTrxStatus3()

Returns a string version of the fax transmission status from a history list element.

C Prototype

```
DWORD RFAXAPI RFaxStringTrxStatus3(
 IN RFAXSERVERHANDLE hServer,
 IN PHISTLISTELEMENT0 pHist,
 IN RFAXPVOID pFaxListElement,
 IN short sFaxInfoLevel,
 IN BOOL fOutputUTF8,
 OUT RFAXSTR pszBuffer,
 IN USHORT usBuffLen
);
```

Arguments

hServer	
pHist	Specify HISTLISTELEMENT0 structure.
pFaxListElement	Specify a FAXLISTELEMENT0 or FAXLISTELEMENT1 structure dependent upon the sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
fOutputUTF8	
pszBuffer/sBuffer	Returns the string version of the transmission history.
usBuffLen	Specify the length of the buffer for the string.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Lets you quickly get a string version of the history without having to break apart the HISTLISTELEMENT0. Uses the information in the specified FAXLISTELEMENT0 or FAXLISTELEMENT1 to get more detailed status information.

See Also

RFaxGetHistoryList() on page 139

RFaxStringFaxStatus() on the next page

RFaxStringTrxStatus() on the previous page

RFaxStringTrxStatus2() on the previous page

RFaxStringTrxStatus4() below

RFaxStringTrxStatus4()

Returns a string version of the fax transmission status from a history list element.

C Prototype

```
DWORD RFAXAPI RFaxStringTrxStatus4(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXPVOID pHistListElem,
 IN short sHistElemLevel,
 IN RFAXPVOID pFaxListElementOrInfo,
 IN short sFaxElemOrInfoLevel,
 IN BOOL fOutputUTF8,
 OUT RFAXSTR pszBuffer,
 IN USHORT usBuffLen
);
```

Arguments

hServer	
pHistListElem	Pointer to history element of type HISTLISTELEMENT0 or HISTLISTELEMENT2
sHistElemLevel	0 or 2 for the level of data structure pointed to by pHistListElem.
pFaxListElementOrInfo	Specify a FAXLISTELEMENT0 or FAXLISTELEMENT1, FAXLISTELEMENT2, FAXLISTELEMENT3*FAXINFO_10* on page 318, or *FAXINFO_11* on page 322.
sFaxElemOrInfoLevel	0, 1, 2, 3, 10, or 11.
fOutputUTF8	
pszBuffer/sBuffer	Returns the string version of the transmission history.
usBuffLen	Specify the length of the buffer for the string.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Lets you quickly get a string version of the history without having to break apart the HISTLISTELEMENT0. Uses the information in the specified FAXLISTELEMENT0 or FAXLISTELEMENT1 to get more detailed status information.

See Also

RFaxGetHistoryList() on page 139

RFaxStringFaxStatus() below

RFaxStringTrxStatus2() on page 115

RFaxStringTrxStatus3() on the previous page

RFaxStringFaxStatus()

Returns a string version of the fax status from the fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus(
 IN PFAXLISTELEMENT0 pFax,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen,
 OUT OPTIONAL USHORT *pusStatusType
);
```

Arguments

pFax	Specify the FAXLISTELEMENT0 structure.
pszBuff	Returns a string version of the current status of the fax.
usBuffLen	Specify the length of the buffer.
pusStatusType	Returns one of the STATUS_TYPE_??? constants.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxStringFaxStatus2() on the next page

RFaxStringFaxStatus3() on the next page

RFaxStringFaxStatus4() on page 119

RFaxStringFaxStatus5() on page 120

RFaxStringFaxStatus6() on page 121

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxInfo10Status() on page 123

RFaxStringFaxStatus2()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus2(
 IN USHORT usFaxStatus,
 IN USHORT usFaxErrorCode,
 IN ULONG ulFaxFlags,
 IN ULONG ulFaxSendDateTime,
 IN USHORT usPagesInFront,
 OUT OPTIONAL RFAXSTR pszBuff,
 IN USHORT usBuffLen,
 OUT OPTIONAL USHORT *pusStatusType
);
```

Arguments

usFaxStatus	Fax status codes (see FAXSTAT_???).
usFaxErrorCode	Fax error code flags (see FAXERROR_???).
ulFaxFlags	Fax record flags (see FAXFLAG_???).
ulFaxSendDateTime	Time the fax should be sent (delay send) — ctime type (GMT0).
usPagesInFront	Pages ahead of this fax to be sent.
pszBuff	Returns a string version of the current status of the fax.
usBuffLen	Specify the length of the buffer.
pusStatusType	Returns one of the STATUS_TYPE_??? constants.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxStringFaxStatus2 lets you specify details without having to create a FAXLISTELEMENT0 or FAXLISTELEMENT1. If you already have a FAXLISTELEMENT0 or FAXLISTELEMENT1, you can use the abbreviated RFaxStringFaxStatus3().

See Also

RFaxStringFaxStatus() on the previous page)

RFaxStringFaxStatus3() below

RFaxStringFaxStatus4() on the next page

RFaxStringFaxStatus5() on page 120

RFaxStringFaxStatus6() on page 121

RFaxStringFaxInfo10Status() on page 123

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxStatus3()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus3(
 IN RFAXPVOID pFaxListElement,
 IN short sFaxInfoLevel,
 IN DWORD dwOptions,
 OUT OPTIONAL RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

pFaxListElement	Specify a FAXLISTELEMENT0 or FAXLISTELEMENT1 structure dependent upon the sFaxInfoLevel argument.

sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
dwOptions	Specify a combination of the FAXSTRINGSTAT_??? flags by bitwise ORing them together, or specify 0 for none.
pszBuff	Returns a string version of the current status of the fax.
usBuffLen	Specify the length of the buffer.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxStringFaxStatus2() lets you specify details without having to create a FAXLISTELEMENT0 or FAXLISTELEMENT1. If you already have a FAXLISTELEMENT0 or FAXLISTELEMENT1, you can use the abbreviated RFaxStringFaxStatus3.

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on the previous page

RFaxStringFaxStatus4() below

RFaxStringFaxStatus5() on the next page

RFaxStringFaxStatus6() on page 121

RFaxStringFaxInfo10Status() on page 123

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxStatus4()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus4(
 IN USHORT usFaxStatus,
 IN USHORT usFaxErrorCode,
 IN ULONG ulFaxFlags,
 IN ULONG ulFaxSendDateTime,
 IN USHORT usPagesInFront,
 IN ULONG ulFaxFlags2,
 IN RFAXCSTR pszToName,
 OUT OPTIONAL RFAXCSTR pszStatus,
 IN USHORT usStatusLen,
 OUT OPTIONAL USHORT *pusStatusType,
);
```

Arguments

usFaxStatus	Fax status codes (see FAXSTAT_???)
usFaxErrorCode	Fax error code flags (see FAXERROR_???)
ulFaxFlags	Fax record flags (see FAXFLAG_???)
ulFaxSendDateTime	Time the fax should be sent (delay send) — ctime type (GMT0).
usPagesInFront	Pages ahead of this fax to be sent.
ulFaxFlags2	Fax record flags (see FAXFLAG2_???)
pszToName	Character buffer receiving To: name.
pszStatus	Character buffer receiving status string.
usStatusLen	Length of pszStatus string buffer being passed in.
pusStatusType	Returns one of the STATUS_TYPE_??? constants.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxStringFaxStatus2() lets you specify details without having to create a FAXLISTELEMENT0 or FAXLISTELEMENT1. If you already have a FAXLISTELEMENT0 or FAXLISTELEMENT1, you can use the abbreviated RFaxStringFaxStatus3.

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on page 118

RFaxStringFaxStatus3() on page 118

RFaxStringFaxStatus5() below

RFaxStringFaxStatus6() on the next page

RFaxStringFaxInfo10Status() on page 123

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxStatus5()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus5(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXPVOID pFaxListElement,
 IN short sFaxInfoLevel,
 IN DWORD dwOptions,
 OUT OPTIONAL RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

hServer	Handle of theRightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pFaxListElement	Specify a FAXLISTELEMENT0 or FAXLISTELEMENT1 structure dependent upon the sFaxInfoLevel argument.
sFaxInfoLevel	Specify the level of information. Must be 10 or 11.
dwOptions	Specify a combination of the FAXSTRINGSTAT_??? flags by bitwise ORing them together, or specify 0 for none.
pszBuff	Returns a string version of the current status of the fax.
usBuffLen	Specify the length of the buffer.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxStringFaxStatus2() lets you specify details without having to create a FAXLISTELEMENT0 or FAXLISTELEMENT1. If you already have a FAXLISTELEMENT0 or FAXLISTELEMENT1, you can use the abbreviated RFaxStringFaxStatus3.

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on page 118

RFaxStringFaxStatus3() on page 118

RFaxStringFaxStatus4() on page 119

RFaxStringFaxInfo10Status() on page 123

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxStatus6()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus6(
 IN RFAXCSTR pszLanguage,
 IN USHORT usFaxStatus,
 IN USHORT usFaxErrorCode,
 IN ULONG ulFaxFlags,
 IN ULONG ulFaxSendDateTime,
 IN USHORT usPagesInFront,
 IN ULONG ulFaxFlags2,
 IN RFAXCSTR pszToName,
 OUT RFAXSTR pszStatus,
 IN USHORT usStatusLen,
 OUT USHORT *pusStatusType
);
```

Arguments

pszLanguage	Language choice (Spanish, German, English, French, French-Canadian, Italian, Japanese, Korean, Portuguese, Chinese, Arabic, Russian).
usFaxStatus	
usFaxErrorCode	
ulFaxFlags	
ulFaxSendDateTime	
usPagesInFront	
ulFaxFlags2	
pszToName	
pszStatus	
usStatusLen	
*pusStatusType	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

.

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on page 118

RFaxStringFaxStatus3() on page 118

RFaxStringFaxStatus4() on page 119

RFaxStringFaxStatus5() on the previous page

RFaxStringFaxInfo10Status() on page 123

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxStatus7()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus7(
 IN RFAXCSTR pszLanguage,
 IN USHORT usFaxStatus,
```

```
    IN USHORT usFaxErrorCode,
    IN ULONG ulFaxFlags,
    IN ULONG ulFaxSendDateTime,
    IN USHORT usPagesInFront,
    IN ULONG ulFaxFlags2,
    IN RFAXCSTR pszToName,
    OUT OPTIONAL RFAXSTR pszStatus,
    IN USHORT usStatusLen,
    OUT OPTIONAL USHORT *pusStatusType
    IN DWORD dwOptions,
    OUT OPTIONAL USHORT *pusStatusID
);
```

Arguments

pszLanguage	Language choice (Spanish, German, English, French, French-Canadian, Italian, Japanese, Korean, Portuguese, Chinese, Arabic, Russian).
usFaxStatus	
usFaxErrorCode	
ulFaxFlags	
ulFaxSendDateTime	
usPagesInFront	
ulFaxFlags2	
pszToName	
pszStatus	
usStatusLen	
*pusStatusType	
dwOptions	See *FAXSTRINGSTAT_??? on page 424* options.
*pusStatusID	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

.

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on page 118

RFaxStringFaxStatus3() on page 118

RFaxStringFaxStatus4() on page 119

RFaxStringFaxStatus5() on page 120

RFaxStringFaxStatus5() on page 120

RFaxStringFaxStatus5() on page 120

RFaxStringFaxInfo10Status() on the next page

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxStatus8()

Returns a string version of the fax status from the specified fields or fax list element.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxStatus8(
 IN RFAXCSTR pszLanguage,
 IN PFAXLISTELEMENT4 faxElem,
 OUT OPTIONAL RFAXSTR pszStatus,
 IN USHORT usStatusLen,
 OUT OPTIONAL USHORT *pusStatusType,
 IN DWORD dwOptions,
 OUT OPTIONAL USHORT *pusStatusID
);
```

Arguments

pszLanguage	Language choice (Spanish, German, English, French, French-Canadian, Italian, Japanese, Korean, Portuguese, Chinese, Arabic, Russian).
faxElem	
pszStatus	
usStatusLen	
*pusStatusType	
dwOptions	See *FAXSTRINGSTAT_???* on page 424 options.
*pusStatusID	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

.

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on page 118

RFaxStringFaxStatus3() on page 118

RFaxStringFaxStatus4() on page 119

RFaxStringFaxStatus5() on page 120

RFaxStringFaxStatus5() on page 120

RFaxStringFaxStatus5() on page 120

RFaxStringFaxInfo10Status() below

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxStringFaxInfo10Status()

Returns a string version of the fax status from the specified fax info structure.

C Prototype

```
DWORD RFAXAPI RFaxStringFaxInfo10Status(
 IN PFAXINFO_10 pFax,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen,
 OUT OPTIONAL USHORT *pusStatusType
);
```

Arguments

pFax	Specify the FAXINFO_10 structure.
pszBuff	Returns a string version of the current status of the fax.
usBuffLen	Specify the length of the buffer.
pusStatusType	Returns one of the STATUS_TYPE_??? constants.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxStringFaxStatus() on page 117

RFaxStringFaxStatus2() on page 118

RFaxStringFaxStatus3() on page 118

RFaxStringFaxStatus4() on page 119

RFaxStringFaxStatus5() on page 120

RFaxGetFaxList() on page 81

RFaxStringTrxStatus() on page 115

RFaxUnlockFax()

Unlocks a fax for performing an operation that can only be performed by one user at a time.

C Prototype

```
DWORD RFAXAPI RFaxUnlockFax(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usReserved,
 IN FAXHANDLE hFax,
 OUT RFAXSTR pszUserId,
 IN DWORD dwUserIdLen
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usLockOptions	See *LOCKFAXOPTION_???* on page 449.
hFax	Handle of the fax to unlock.
pszUserId	If return = RFAXERR_FAXLOCKED, indicates the ID of the user who has the document locked.
dwUserIdLen	Number of characters that can be written to pszUserId.

Return Values

Return	Result
RFAX_SUCCESS	Fax can be unlocked.
RFAXERR_FAXLOCKED	Fax is locked by another user.

Remarks

See Also

RFaxLockFax() on page 99

Chapter 8: Event functions

The event functions let you dequeue various server events. Most of these functions will have little practical application for API developers. The RFaxGetCompletionEvent functions are, however, useful for getting information about faxes that have transmitted successfully.

RFaxGetArchiveEvent()

This function gets archive events from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetArchiveEvent(
 IN RFAXSERVERHANDLE hServer,
 OUT PARCHIVEREQUEST lpArcReq
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpArcReq	Returns the ARCHIVEREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

In order to generate an archive event on the fax server with this function, “Archive Sent Faxes” must be enabled in individual user accounts. Also, verify that none of the RightFax WorkServer Modules on the RightFax server are performing the archive function. If they are, your application and the WorkServer will compete for archive events. An archive event is generated when one of the following occurs:

- The fax is successfully sent.
- The fax fails for the last time. This is denoted in FaxUtil by the status of ED (Error Done).

The fax has failed for the last time and the user “kicks” the fax (reschedules it).

Note that, when the fax has failed for the last time and the user “kicks” (or reschedules) the fax, the process restarts and the fax will generate an additional completion event when it is again complete. Therefore, this can happen repeatedly for faxes that keep failing and keep getting “kicked.” An archive event will not be generated when the fax fails, but the server will retry it. This is denoted in FaxUtil by the status ER (Error Retry). A common use for the RFaxGetArchiveEvent API call is to get information about sent faxes when they are successfully sent or failed during a large broadcast. You can then use the information from the archive event to update your databases. Completion events may be more appropriate in many cases.

See the notes for the SVCINFO_10 structure for more details.

See Also

RFaxGetCompletionEvent() below

RFaxGetCompletionEvent()

This function retrieves a completion event from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetCompletionEvent(
 IN RFAXSERVERHANDLE hServer,
 OUT PCOMPLETEREQUEST lpCompReq
);
```

Arguments

hServer	Handle of the RightFax server creating using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpCompReq	Returns a pointer to the COMPLETEREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This functions is similar to RFaxGetArchiveEvent(), except that it does not depend on archiving being enabled for the user. Instead, the FAXFLAG_COMPLETEEVENT flag should be enabled when the fax is generated. If this flag is enabled, the fax will generate a completion event.

A completion event is generated when of these things happens:

- The fax is successfully sent.
- The fax encounters an error during its conversion or building process (ED:Conversion Failed).

The fax fails for the last time. This is denoted in FaxUtil by the status of ED (Error Done).

Note that when the fax has failed for the last time and the user “kicks” (or reschedules) the fax, this process restarts and the fax will generate an additional completion event when it is again complete. Therefore, this can happen repeatedly for faxes that keep failing and keep getting “kicked.”

Notice that a completion event is similar to an archive event, except that a completion event will be generated when a fax is not going to be sent at all, usually due to some sort of conversion failure. For example, a fax needs a PDF to G3 conversion prior to being sent, but the PDF conversion fails, so the fax never even gets scheduled for a phone line. In this example, a completion event will be generated, but an archive event will not.

It is possible to have both an archive event and a completion event generated for a single fax. In this case, both events are posted asynchronously so care must be taken when one of the event handlers attempts to delete the fax while handling the event, i.e., one of the handlers can delete the fax prior to the other handler getting a chance to load the fax and process it.

Completion events are provided for the application programmer and are not used internally by RightFax client or server components. Archive events are, in some cases, used by the RightFax server. This makes completion events the preferred notification to the application programmer that a fax has completed.

RFaxGetCompletionEvent2() and RFaxGetCompletionEvent3() provide the added ability to get completion events for a particular user by user ID. These functions require the newer

COMPLETEREQUEST2 structure. RFaxGetCompletionEvent3() also provides the added ability of filtering on notification channel.

See Also

RFaxGetCompletionEvent2()

RFaxGetCompletionEvent3()

RFaxCreateServerHandle()

RFaxGetArchiveEvent()

RFaxGetCompletionEvent2()

Retrieves a completion event from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetCompletionEvent2(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL RFAXSTR lpUserID,
 OUT PCOMPLETEREQUEST2 lpCompReq
);
```

Arguments

hServer	Handle of the RightFax server creating using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Pass in the ID of the user for whom to get completion events.
lpCompReq	Returns a pointer to COMPLETEREQUEST2 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is identical to RFaxGetCompletionEvent() except that RFaxGetCompletionEvent2 provides the added ability to get completion events for a particular user by user ID. This function requires the newer COMPLETEREQUEST2 structure.

There is a limitation with RFaxGetCompletionEvent2. When using this function via any copy of the RightFax API built on or after May 1, 2000, the function will fail for 6.x and 7.x servers that use Faxdb.exe built before this date.

Contact RightFax Technical Support for service packs to upgrade your 6.x or 7.x Faxdb.exe to an appropriate version. Service packs and the API are available online.

See Also

RFaxGetCompletionEvent()

RFaxGetCompletionEvent3()

RFaxGetCompletionEvent3()

This function retrieves a completion event from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetCompletionEvent3(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL RFAXSTR lpUserID,
 IN OPTIONAL BOOL bFilterOnNotifyChannel,
 IN OPTIONAL unsigned short
usNotificationChannels,
 OUT PCOMPLETEREQUEST2 lpCompReq
);
```

Arguments

hServer	Handle of the RightFax server creating using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

lpUserID	Pass in the ID of the user for whom to get completion events.
bFilterOnNotifyChannel	When set to True, usNotificationChannels is used as a channel filter.
usNotificationChannels	When bFilterOnNotifyChannel is True, this field is used as a channel filter. Set this to include the channels for which to get completion events.
lpCompReq	Returns a pointer to the COMPLETEREQUEST2 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is identical to RFaxGetCompletionEvent2() except that RFaxGetCompletionEvent3 also provides the added ability to filter on notification channels.

There is a limitation with RFaxGetCompletionEvent3. When using this function via any copy of the RightFax API built on or after May 1, 2000, the function will fail for 6.x and 7.x servers that use Faxdb.exe built before this date.

Contact RightFax Technical Support for service packs to upgrade your 6.x or 7.x Faxdb.exe to an appropriate version. Service packs and the API are available online.

See Also

RFaxGetCompletionEvent()

RFaxGetCompletionEvent2()

RFaxGetEventList()

This function retrieves a list of all the events stored on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetEventList(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLH	Returns a handle to a list of SVCINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used to retrieve all of the events stored on the RightFax server at one time. This function should be used with caution because there could be tens or hundreds of thousands of events on a server requiring a long time to execute. This API, unlike the RFaxGetGenericEvent() and other API calls, does not remove the requests from the server work queue. It is also likely, and normal, that the list is immediately out of date upon retrieval, because the WorkServers, Gateways, and other processes on the fax server continuously dequeue events.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetGenericEvent()

This function retrieves a variety of generic events from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetGenericEvent(
 IN RFAXSERVERHANDLE hServer,
 IN DWORD dwMessageTypes,
 OUT PWORKREQUEST lpWorkReq
);
```

Arguments

hServer	Handle of the RightFax server creating by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
dwMessageTypes	Type of requests to get from the RightFax server. Specify one or more of the REQUEST_??? constants.
lpWorkReq	Returns the WORKREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is useful for getting a variety of work requests from the RightFax server. When retrieving specific requests, verify that the RightFax WorkServer modules are not also fulfilling those requests, because you will end up competing with it. See WORKREQUEST for more details.

RFaxGetGenericMailEvent()

This function retrieves generic mail events (but not cc:Mail, MS-Mail, or GroupWise).

C Prototype

```
DWORD RFAXAPI RFaxGetGenericMailEvent(
 IN RFAXSERVERHANDLE hServer,
 IN DWORD ulRequestMask,
 IN USHORT usMailType,
 OUT PWORKREQUEST lpWorkReq
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulRequestMask	Optionally, specify a mask of *REQUEST_??* on page 465 or 0 for default mail mask.
usMailType	Type of mail requests to receive. Specify one of the GENMAILTYPE_??? constants.
lpWorkReq	Returns the WORKREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

When retrieving email messages, verify that an email gateway module is not also retrieving the requests. Otherwise, the email gateway module may retrieve the requests before your application.

RFaxGetMessageEvent()

This function can be used to dequeue notification events from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetMessageEvent(
 IN RFAXSERFVERHANDLE hServer
```

```
    IN USHORT usNotifyType
    OUT PMESSAGEREQUEST lpMsgReq
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usNotifyType	Type of notifications to receive. Specify one of the NOTIFY_TYPE_??? constants. NOTIFY_TYPE_ANY can be used to specify that any type of notification or message event should be dequeued.
lpMsgReq	Returns the MESSAGEREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function can be used to get custom notification types from the RightFax server, allowing you to create a custom notification.

If you are trying to get one of the notification types that are normally handle by the RightFax server (e.g., network broadcasts), verify that none of the WorkServers have network notifications enabled. If the WorkServers do have network notifications enabled, then you will be “fighting” with the WorkServers for requests. Also, verify that the RightFax Server Module configuration program in Windows Control Panel has Allow Notifications enabled. If you do not have Allow Notifications enabled, then the RightFax server will not generate notification requests.

See the notes for the WORKREQUEST structure for more details.

RFaxGetPrintEvent()

This function retrieves a variety of generic events from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetPrintEvent(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXSTR aszEligiblePrinters,
 OUT PWORKREQUEST lpWorkReq
);
```

Arguments

hServer	Handle of the RightFax server creating by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
aszEligiblePrinters	A character array[25][10] of printers to check.
lpWorkReq	Returns the WORKREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See WORKREQUEST for more details.

Compatibility

Compatible with RightFax versions.

RFaxGetValidateEvent()

This function gets a validation event from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetValidateEvent(
 IN RFAXSERVERHANDLE hServer
 OUT PVALIDATEREQUEST lpValidateReq
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpValidateReq	Returns a pointer to the VALIDATEREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to get validation events from the RightFax server. To use this feature, billing code verification must enabled in Enterprise Fax Manager. Set billing codes to external validation in the RightFax Server configuration program in Windows Control Panel.

Once a validation request is received, you can view the document and approve or disapprove the fax using RFaxSaveStatus().

RFaxPurgeEvents()

Purges all current work requests that match the specified mask.

C Prototype

```
DWORD RFAXAPI RFaxPurgeEvents(
 IN RFAXSERVERHANDLE hServer,
 IN DWORD dwMessageTypes,
 OUT OPTIONAL ULONG *pulCount
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
dwMessageTypes	Composed of REQUEST_??? constants bitwise ORed together.
pulCount	The count of the number of requests purged.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Purges all of the work requests that match the specified message types. To use this function, the caller must be logged in as an NT account that is linked to a RightFax account with RightFax administrative permission.

See Also

WORKREQUEST on page 393

RFaxSaveEvent()

This function stores an event in RightFax that can be retrieved either by another RFaxGetGenericEvent() or by the RightFax WorkServer or Gateway modules.

C Prototype

```
DWORD RFAXAPI RFaxSaveEvent(
 IN RFAXSERVERHANDLE hServer,
 IN PWORKREQUEST lpWorkReq
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpWorkReq	Pointer to the WORKREQUEST structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Stores requests that need to be retrieved from another application (such as a queue server or RightFax). For example, you can submit requests to the RightFax server to perform a PCL-5 conversion job. To retrieve a request you can use RFaxGetGenericEvent().

RFaxSaveEvent2()

This function stores an event in RightFax that can be retrieved either by another RFaxGetGenericEvent() or by the RightFax WorkServer or Gateway modules.

C Prototype

```
DWORD RFAXAPI RFaxSaveEvent2(
 IN RFAXSERVERHANDLE hServer,
 IN PWORKREQUEST lpWorkReq
 IN.USHORT. usPriority
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpWorkReq	Pointer to the WORKREQUEST structure.
usPriority	One of PRIORITY_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Stores requests that need to be retrieved from another application (such as a queue server or RightFax). For example, you can submit requests to the RightFax server to perform a PCL-5 conversion job. To retrieve a request you can use RFaxGetGenericEvent().

RFaxSaveEvent3()

This function stores an event in RightFax that can be retrieved either by another RFaxGetGenericEvent() or by the RightFax WorkServer or Gateway modules.

C Prototype

```
DWORD RFAXAPI RFaxSaveEvent3(
 IN RFAXSERVERHANDLE hServer,
 IN PWORKREQUEST lpWorkReq
 IN.USHORT. usPriority
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpWorkReq	Pointer to the WORKREQUEST structure.
usPriority	One of PRIORITY_???.
hService	Target service when required. Otherwise, specify NULL.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Stores requests that need to be retrieved from another application (such as a queue server or RightFax). For example, you can submit requests to the RightFax server to perform a PCL-5 conversion job. To retrieve a request you can use RFaxGetGenericEvent().

RFaxSaveStatus()

This function can store or change the status of a fax.

C Prototype

```
DWORD RFAXAPI RFaxSaveStatus(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hObject,
 IN short sStatus
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hObject	Handle of the fax to set the status of.
sStatus	Sets the status of the fax. Zero indicates a success, and non-zero indicates a failure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to change the status of a fax that is in validation.

See Also

RFaxApproveFax() on page 58

RFaxGetValidateEvent() on page 130

RFaxSaveStatus2()

This function can store or change the status of a fax.

C Prototype

```
DWORD RFAXAPI RFaxSaveStatus2(
 IN RFAXSERVERHANDLE hServer,
 IN PREQUESTSTATUS prs
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
prs	Pointer to the REQUESTSTATUS structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to change the status of a fax that is in validation.

Chapter 9: Certified Delivery Users

These functions allow you to configure and setup Certified Delivery Users.

RFaxVerifyCertifiedDeliveryUser()

This function verifies a Certified Delivery User on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxVerifyCertifiedDeliveryUser(
IN RFAXSERVERHANDLE hServer,
IN RFAXCSTR pszEmail,
IN RFAXCSTR lpPassword,
IN ULONG loginType // One of VERIFYUSER_???
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszEmail	Specify the email address of the user to verify
lpPassword	Specify the password of the user specified in lpUserID to verify.
loginType	One of VERIFYUSER_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxLoadCertifiedDeliveryUser()

This function loads the Certified Delivery User from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxLoadCertifiedDeliveryUser(
IN RFAXSERVERHANDLE hServer,
IN RFAXCSTR pszEmail, // Email of certified
delivery user to load.
IN USHORT usInfoLevel, // Information level.
Must match desired CERTIFIEDDELIVERYUSERINFO_#.
IN ULONG ulReserved, // Reserved for future
use. Pass 0.
OUT RFAXPVOID pinfo //
CERTIFIEDDELIVERYUSERINFO_10.
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszEmail	String containing email address.
usInfoLevel	Specify the information level to load. This must be set to 10
ulReserved	Reserved - currently must be zero.
pinfo	CERTIFIEDDELIVERYUSERINFO_

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveCertifiedDeliveryUser()

This function saves the details of a Certified Delivery User to the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveCertifiedDeliveryUser(
IN RFAXSERVERHANDLE hServer,
IN USHORT usInfoLevel, // Information level.
Must match desired CERTIFIEDDELIVERYUSERINFO_#.
IN RFAXPVOID pinfo, //
CERTIFIEDDELIVERYUSERINFO_10.
IN ULONG ulReserved // Reserved for future use.
Pass 0.
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Specify the information level to save. This must be set to 10
pinfo	CERTIFIEDDELIVERYUSERINFO_
ulReserved	Reserved - currently must be zero.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxDeleteCertifiedDeliveryUser()

This function deletes a Certified Delivery User from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxDeleteCertifiedDeliveryUser(
IN RFAXSERVERHANDLE hServer,
IN RFAXCSTR pszEmail, // Email of certified
delivery user to delete.
IN ULONG ulReserved // Reserved for future use.
Pass 0.
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszEmail	String containing email address.
ulReserved	Reserved - currently must be zero.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetCertifiedDeliveryUserList()

This function gets the Certified Delivery User list from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetCertifiedDeliveryUserList
(
IN RFAXSERVERHANDLE hServer,
IN USHORT usInfoLevel, // Information level.
Must match desired CERTIFIEDDELIVERYUSERINFO_#.
IN ULONG ulReserved, // Reserved for future
use. Pass 0.
OUT RFAXLISTHANDLE FAR* lpLH // If return == 0,
returns a handle to list of
CERTIFIEDDELIVERYUSERINFO_# structures
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Specify the information level to load. This must be set to 10.
ulReserved	Reserved - currently must be zero.
lpLH	If return value is 0, returns a handle to list of CERTIFIEDDELIVERYUSERINFO_ structures depending upon usInfoLevel parameter.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxChangeCertifiedDeliveryUserPassword ()

This function is used to change the password for a Certified Delivery User.

C Prototype

```
DWORD RFAXAPI
RFaxChangeCertifiedDeliveryUserPassword(
IN RFAXSERVERHANDLE hServer,
IN RFAXCSTR pszEmail, // Email of user whose
password should be changed.
IN RFAXCSTR lpOldPassword, // Current password
IN RFAXCSTR lpNewPassword, // New password
IN USHORT usReserved // Reserved for future
use. Pass 0.
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszEmail	String containing email address.
lpOldPassword	Specify the current password.
lpNewPassword	Specify the new password.
usReserved	Reserved - currently must be zero.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxForgotCertifiedDeliveryUserPassword()

This function resets a forgotten Certified Delivery User password on the RightFax server.

C Prototype

```
DWORD RFAXAPI
RFaxForgotCertifiedDeliveryUserPassword(
IN RFAXSERVERHANDLE hServer,
IN RFAXCSTR pszEmail, // Email of user whose
password was forgotten.
IN USHORT usReserved // Reserved for future
use. Pass 0.
);
```

Arguments

hServer	Handle of the RightFax server created by RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszEmail	String containing email address.
usReserved	Reserved - currently must be zero.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 10: Fax history functions

The fax history functions load and save history information for specified faxes.

RFaxGetHistoryElementDescription ()

This function gets a detailed description for a history list element.

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryElementDescription
(
 IN RFAXSERVERHANDLE hServer,
 IN void* pHistoryElement,
 IN USHORT usHistoryInfoLevel,
 IN void* pFax,
 IN USHORT usFaxInfoLevel,
 IN USHORT usOptions,
 OUT LPTSTR pszDescription,
 IN DWORD dwSize
);
```

Arguments

hServer	Server handle
pHistoryElement	HISTLISTELEMENT0, HISTLISTELEMENT1, or HISTLISTELEMENT2.
usHistoryInfoLevel	Level of HISTLISTELEMENT: 0, 1, or 2.
pFax	Specify a FAXLISTELEMENT0, FAXLISTELEMENT1, FAXLISTELEMENT2, or FAXLISTELEMENT3 structure or FAXINFO_10 or FAXINFO_11.
usFaxInfoLevel	Level of FAXLISTELEMENT: 0-3 or FAXINFO_: 10-11
usOptions	See HISTDESC_???.
pszDescription	Description for pHistoryElement.
dwSize	Maximum characters that can be stored at pszDescription.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Gets the detailed description for a history list element..

RFaxGetHistoryItemDetails()

This function gets the detailed description for a history item. The specified history item is assumed to have been returned by a successfull call to RFaxGetHistoryList() or RFaxGetHistoryList2().

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryItemDetails(
 IN PHISTLISTELEMENT0 pHist,
 OUT RFAXSTR pszDesc,
 IN LONG lDescBufSize

);
```

Arguments

pHist	History item to get details for
pszDesc	Description
lDescBufSize	Size of buffer pointed to by lpDesc

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Gets the detailed description for a history item.

RFaxGetHistoryItemDetails2()

This function Introduces the ability to specify multiple types of HISTLISTELEMENT.

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryItemDetails2(
 IN RFAXPVOID pHist,
 IN USHORT usElemLevel,
 OUT RFAXSTR pszDesc,
 IN LONG lDescBufSize

);
```

Arguments

pHist	History item of type HISTLISTELEMENT0 or HISTLISTELEMENT2.
usElemLevel	Level of data at pHist (0 or 2).
pszDesc	Description
lDescBufSize	Size of buffer pointed to by lpDesc

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Gets the detailed description for a history item.

RFaxGetHistoryList()

This function obtains a list of history information about a particular fax.

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to retrieve the history list from.
lpLH	Returns a handle to a list of HISTLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves a list of history elements that can then be displayed to the user or exported to a database file.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetHistoryList2() below

RFaxGetHistoryList3() on the next page

RFaxGetHistoryList4() on the next page

RFaxGetHistoryList2()

This function obtains a list of history information about a particular fax.

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryList2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to retrieve the history list from.
lpLH	Returns a handle to a list of HISTLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves a list of history elements that can then be displayed to the user or exported to a database file.

See Also

RFaxGetHistoryList2Close() below

RFaxGetHistoryList() on the previous page

RFaxGetHistoryList3() on the next page

RFaxGetHistoryList4() on the next page

RFaxGetHistoryList2Close()

This function closes RFaxGetHistoryList2().

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryList2Close((
 IN RFAXLISTHANDLE hList,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to retrieve the history list from.
lpLH	Returns a handle to a list of HISTLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success

Return	Result
RFAXERR_???	Failure

Remarks

This function closes RFaxGetHistoryList2().

RFaxGetHistoryList3()

This function obtains a list of history information about a particular fax.

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryList3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Fax to get history records for.
usInfoLevel	Must be 0, 1, or 2..
lpLH	Dependent on usInfoLevel, returns a handle to a list of HISTLISTELEMENT0, HISTLISTELEMENT1 or HISTLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves a list of history elements that can then be displayed to the user or exported to a database file.

See Also

RFaxGetHistoryList() on page 139

RFaxGetHistoryList2() on the previous page

RFaxGetHistoryList4() below

RFaxGetHistoryList4()

This function obtains a list of history information about a particular fax.

C Prototype

```
DWORD RFAXAPI RFaxGetHistoryList4(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN USHORT usInfoLevel,
 IN USHORT usOptions,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Fax for which to get history records.
usInfoLevel	Must be 0, 1, or 2.
usOptions	See *GETHISTORYLISTOPT_???* on page 435.
lpLH	If return = 0, returns a handle to list of *HISTLISTELEMENT0* on page 341, *HISTLISTELEMENT1* on page 344, or *HISTLISTELEMENT2* on page 345 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves a list of history elements that can then be displayed to the user or exported to a database file.

See Also

RFaxGetHistoryList() on page 139

RFaxGetHistoryList2() on page 140

RFaxGetHistoryList3() on the previous page

RFaxInitHistoryItem()

This function gets the detailed description for a history item. The specified history item is assumed to have been returned by a successful call to RFaxGetHistoryList() or RFaxGetHistoryList2().

C Prototype

```
DWORD RFAXAPI RFaxInitHistoryItem(
 IN RFAXSERVERHANDLE hServer,
 IN OUT PHISTLISTELEMENT0 pHist,
 IN DWORD dwGenHistType
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pHist	History item to initialize.
dwGenHistType	History type. See GENHISTTYPE_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the given history element for the specified fax. If the fNew argument is True, a new history element will be saved for the fax. If the fNew argument is False, the “handle” field of the HISTLISTELEMENT0 structure must be a valid handle.

RFaxInitHistoryItem1()

This function gets the detailed description for a history item. The specified history item is assumed to have been returned by a successful call to RFaxGetHistoryList() or RFaxGetHistoryList2().

C Prototype

```
DWORD RFAXAPI RFaxInitHistoryItem(
 IN RFAXSERVERHANDLE hServer,
 IN OUT PHISTLISTELEMENT1 pHist,
 IN DWORD dwGenHistType
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pHist	History item to initialize.
dwGenHistType	History type. See GENHISTTYPE_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the given history element for the specified fax. If the fNew argument is True, a new history element will be saved for the fax. If the fNew argument is False, the “handle” field of the HISTLISTELEMENT0 structure must be a valid handle.

RFaxInitHistoryItem2()

This function gets the detailed description for a history item. The specified history item is assumed to have been returned by a successful call to RFaxGetHistoryList() or RFaxGetHistoryList2().

C Prototype

```
DWORD RFAXAPI RFaxInitHistoryItem(
 IN RFAXSERVERHANDLE hServer,
 IN OUT PHISTLISTELEMENT2 pHist,
 IN DWORD dwGenHistType
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pHist	History item to initialize.
dwGenHistType	History type. See GENHISTTYPE_???.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the given history element for the specified fax. If the fNew argument is True, a new history element will be saved for the fax. If the fNew argument is False, the “handle” field of the HISTLISTELEMENT0 structure must be a valid handle.

RFaxSaveHistory()

This function saves the history for a specified fax.

C Prototype

```
DWORD RFAXAPI RFaxSaveHistory(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN SHORT fNew,
 IN PHISTLISTELEMENT0 lpHE0
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax for which to save the history.
fNew	Specify True to save the specified information as a new history element. Specify False to replace an existing element.
lpHE0	A HISTLISTELEMENT0 structure containing history information to be saved. If the fNew argument is False, the “handle” field of the HISTLISTELEMENT0 structure must be a valid handle.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the given history element for the specified fax. If the fNew argument is True, a new history element will be saved for the fax. If the fNew argument is False, the “handle” field of the HISTLISTELEMENT0 structure must be a valid handle.

See Also

RFaxGetHistoryList() on page 139

RFaxSaveHistory2()

This function saves the history for a specified fax.

C Prototype

```
DWORD RFAXAPI RFaxSaveHistory2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN SHORT fNew,
 IN USHORT usInfoLevel,
 IN RFAXPVOID lpHE
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax for which to save the history.
fNew	Specify True to save the specified information as a new history element. Specify False to replace an existing element.
usInfoLevel	Specifythe level of information to return from the RightFaxserver (0, 1, or 2).
lpHE	Pointer to history record of type HISTLISTELEMENT0, HISTLISTELEMENT1, or HISTLISTELEMENT2.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveHistory()

Chapter 11: User attribute functions

The user attribute functions dynamically associate attributes with a user.

RFaxDeleteUserTag()

Deletes an attribute associated with the specified user.

C Prototype

```
DWORD RFAXAPI RFaxDeleteUserTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN RFAXCSTR pszName
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the user whose attribute will be deleted.
pszName	Name of the attribute to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetUserTagList()

Get attributes associated with the specified user.

C Prototype

```
DWORD RFAXAPI RFaxGetUserTagList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the user whose attributes will be listed.
lpLH	0 or a handle to a list oTAGINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadUserTag()

Loads an attribute associated with the specified user.

C Prototype

```
DWORD RFAXAPI RFaxLoadUserTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN RFAXCSTR pszName,
 OUT PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of user to load attribute from.
pszName	Name of the attribute to load.
lpTI10	Details of the loaded attribute. Returns the TAGINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveUserTag()

Save an attribute associated with the specified user.

C Prototype

```
DWORD RFAXAPI RFaxSaveUserTag(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNew,
 IN PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNew	TRUE if the attribute is new to the user referred to in lpTI10 hOwner.
lpTI10	Details of the attribute to save. Returns the TAGINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 12: User functions

The user functions opens a specific user or a list of users, creates and deletes users, and manipulates user accounts on the RightFax server.

RFaxChangePassword()

This function changes the password of a user ID.

C Prototype

```
DWORD RFAXAPI RFaxChangePassword(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpOldPassword,
 IN RFAXCSTR lpNewPassword,
 IN BOOL bVerify
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID of the user for whom to change the password.
lpOldPassword	Specify the current password if bVerify is set to True. Set to Null if bVerify is set to False (allows you to change the current password without knowing the old password.)
lpNewPassword	Specify the new password.
bVerify	Specify True to verify the password (lpOldPassword must be set to the current password for the function to succeed). Specify False to change the password without verifying.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use extreme caution in setting bVerify to False.

See Also

RFaxLoadUser() on page 155

RFaxChangeUserFlags()

This function changes the flags for the specified user.

C Prototype

```
DWORD RFAXAPI RFaxChangeUserFlags(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN ULONG ulFlags,
 IN USHORT usFlagGroup,
 IN BOOL bUnset,
```

```
    IN OPTIONAL ULONG *pulNewFlags
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle to the user whose flags are to be changed.
ulFlags	Flags to set or remove (dependent upon the bUnset and usFlagGroup arguments). A group of USERFLAGS_??? or USERFLAGS2_??? flags combined by bitwise ORing them together.
usFlagGroup	Specify 0 to modify userflags field with a combination of USERFLAGS_??? value or 1 to modify ulUserFlags2 field with a combination of USERFLAGS2_??? value.
bUnset	Set to True to remove the specified flags for the given user. Set to False to add the specified flags to the given user.
pulNewFlags	Optional resulting flags returned on success.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Sets and returns the flags for the specified user. You must pass the appropriate user flag constants for the flag group you are modifying. When 0 is passed for usFlagGroup, use the USERFLAGS_??? constants. When 1 is passed for usFlagGroup, use the USERFLAGS2_??? constants.

See Also

RFaxLoadUser() on page 155

RFaxDeleteUser()

This function deletes a user, including all their faxes and phonebook items, from RightFax.

C Prototype

```
DWORD RFAXAPI RFaxDeleteUser(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN HWND hWndParent
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user to delete.
hWndParent	Handle to windows that will be used as a parent to a progress dialog box. If the RFaxDeleteUser function must empty a user (delete all faxes and phonebook entries owned by the user) it can display a progress dialog box. Such a status message will appear if a valid window handle is passed in this argument. A 0 or Null value will ensure that no dialog box will appear.
lReserved	Reserved value. Must be zero (0).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

When a user is deleted with this function, all the faxes and phonebook entries of that user also are deleted. Note that the process of deleting all a user's faxes and phonebook entries can be lengthy, and thus you

may want to execute it on a separate thread or pass a window handle to open a progress dialog box.

RFaxDeleteUser will delete a user and all of the user's faxes and phonebook entries without confirmation.

See Also

RFaxDeleteUser2() below

RFaxDeleteUser2()

This function deletes a user, including all their faxes and phonebook items, from RightFax.

C Prototype

```
DWORD RFAXAPI RFaxDeleteUser2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN DWORD dwFlags,
 IN HWND hWndParent
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user to delete.
dwFlags	One or more of the DELETEUSER_FLAGS_???.
hWndParent	Handle to windows that will be used as a parent to a progress dialog box. If the RFaxDeleteUser function must empty a user (delete all faxes and phonebook entries owned by the user) it can display a progress dialog box. Such a status message will appear if a valid window handle is passed in this argument. A 0 or Null value will ensure that no dialog box will appear.
lReserved	Reserved value. Must be zero (0).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to the RFaxDeleteUser() except that it allows control over whether or not a user is emptied.

RFaxGetNewFaxCount()

This function gets statistics for all of a specified user's faxes.

C Prototype

```
DWORD RFAXAPI RFaxGetNewFaxCount(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN USHORT usFolder,
 OUT GETNEWFAXCOUNT *lpGNFC
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to load the new fax count.
usFolder	Specify the FAXFOLDER_??? to load. Specify zero to load all folders.
lpGNFC	Returns the GETNEWFAXCOUNT structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Gets statistics for all of a specified user's faxes in a GETNEWFAXCOUNT structure.

RFaxGetPagedUserList()

This function loads a subset of users that match the query.

C Prototype

```
DWORD RFAXAPI RFaxGetPagedUserList(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL USHORT usGetOptions,
 IN OPTIONAL LPCSTR searchWords,
 IN USHORT usOrder,
 IN BOOL fAscending,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG* plTotal,
 IN short sInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, 1, or 2 depending on whether the caller wants a list of USERLISTELEMENT0, USERLISTELEMENT1, USERLISTELEMENT2 structures.
usGetOptions	0 or combination of *GETUSERS_???* on page 436 flags.
searchWords	Words to search for in the list.
usOrder	Order in which to load the paged records. See *USERORDER_???* on page 491.
fAscending	TRUE for ascending; FALSE for descending.
lPageIndex	0-based index of page to return.
usPageSize	Number of users per page.
plTotal	When not null, receives total records.
lpLH	Returns a handle to a list of USERLISTELEMENT0 or USERLISTELEMENT1 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used to get a detailed list of all RightFax users.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetUserList() below

RFaxGetUserList2() on the next page

RFaxGetUserList3() on page 152

RFaxGetUserList4() on page 152

RFaxGetUserList5() on page 153

RFaxGetUserList()

This function loads the list of users.

C Prototype

```
DWORD RFAXAPI RFaxGetUserList(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, 1, or 2 depending on whether the caller wants a list of USERLISTELEMENT0, USERLISTELEMENT1, USERLISTELEMENT2 structures.
lpLH	Returns a handle to a list of USERLISTELEMENT0 or USERLISTELEMENT1 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used to get a detailed list of all RightFax users.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetUserList2() below

RFaxGetUserList3() on the next page

RFaxGetUserList4() on the next page

RFaxGetUserList5() on page 153

RFaxGetUserList2()

This function loads the list of users.

C Prototype

```
DWORD RFAXAPI RFaxGetUserList2(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN SHORT sSortIndex,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, 1, or 2 depending on whether the caller wants a list of USERLISTELEMENT0, USERLISTELEMENT1, USERLISTELEMENT2 structures.
sSortIndex	One of USERORDER_???.
lpLH	Returns a handle to a list of USERLISTELEMENT0 or USERLISTELEMENT1, or USERLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used to get a detailed list of all RightFax users.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetUserList() on page 150

RFaxGetUserList3() below

RFaxGetUserList4() below

RFaxGetUserList5() on the next page

RFaxGetUserList3()

This function loads the list of users.

C Prototype

```
DWORD RFAXAPI RFaxGetUserList3(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN SHORT sSortIndex,
 IN OPTIONAL RFAXCSTR lpSearchString,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, 1, or 2 depending on whether the caller wants a list of USERLISTELEMENT0, USERLISTELEMENT1, USERLISTELEMENT2 structures.
sSortIndex	One of USERORDER_???.
lpSearchString	Return only users whose name contains the specified search string.
lpLH	Returns a handle to a list of USERLISTELEMENT0 or USERLISTELEMENT1, or USERLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used to get a detailed list of all RightFax users.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetUserList() on page 150

RFaxGetUserList2() on the previous page

RFaxGetUserList4() below

RFaxGetUserList5() on the next page

RFaxGetUserList4()

This function loads the list of users.

C Prototype

```
DWORD RFAXAPI RFaxGetUserList4(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN SHORT sSortIndex,
 IN OPTIONAL RFAXCSTR lpSearchString,
 IN OPTIONAL USHORT usGetOptions,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

sInfoLevel	Specify the information level. Must be 0, 1, or 2 depending on whether the caller wants a list of USERLISTELEMENT0, USERLISTELEMENT1, USERLISTELEMENT2 structures.
sSortIndex	One of USERORDER_???.
lpSearchString	Return only users whose name contains the specified search string.
usGetOptions	0 or a combination of GETUSERS_??? flags.
lpLH	Returns a handle to a list of USERLISTELEMENT0 or USERLISTELEMENT1, or USERLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is used to get a detailed list of all RightFax users.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetUserList() on page 150

RFaxGetUserList2() on page 151

RFaxGetUserList3() on the previous page

RFaxGetUserList5() below

RFaxGetUserList5()

This function loads the list of users.

C Prototype

```
DWORD RFAXAPI RFaxGetUserList5(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN short sSortIndex,
 IN OPTIONAL RFAXCSTR lpSearchString,
 IN OPTIONAL USHORT usGetOptions,
 IN OPTIONAL USHORT usSearchOptions,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, 1, or 2 depending on whether the caller wants a list of USERLISTELEMENT0, USERLISTELEMENT1, USERLISTELEMENT2 structures.
sSortIndex	One of USERORDER_???.
lpSearchString	Return only users whose name contains the specified search string.
usGetOptions	0 or a combination of GETUSERS_??? flags.
usSearchOptions	0 or combination of *SEARCHUSEROPTS_???* on page 477 flags. Defaults to SEARCHUSEROPTS_USERNAME if a lpSearchString is passed in.
lpLH	Returns a handle to a list of USERLISTELEMENT0 or USERLISTELEMENT1, or USERLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success

Return	Result
RFAXERR_???	Failure

Remarks

This function is used to get a detailed list of all RightFax users.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetUserList() on page 150

RFaxGetUserList2() on page 151

RFaxGetUserList3() on page 152

RFaxGetUserList4() on page 152

RFaxGetVerifiedUser()

This function gets the user ID of the user that was most recently verified via *RFaxVerifyUser()* on page 165, *RFaxVerifyUser2()* on page 165, or *RFaxVerifyVoiceUser()* on page 167.

C Prototype

```
DWORD RFAXAPI RFaxGetVerifiedUser(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXSTR pszUserID,
 IN DWORD dwUserIDSize,
 IN RFAXCSTR lpPassword,
 OUT OPTIONAL LOGINFLAGS*
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszUserID	Buffer for logged in user's ID.
dwUserIDSize	Number of characters that can be written to pszUserID.
pulLoginFlags	Logged in user's *LOGINFLAGS_???* on page 450, or Null if not needed.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_BADKEY	The specified user cannot be found.

Remarks

RFaxGetVerifiedUserToken()

This function gets the user token of the user that was most recently verified via *RFaxVerifyUser()* on page 165, *RFaxVerifyUser2()* on page 165, or *RFaxVerifyVoiceUser()* on page 167.

C Prototype

```
DWORD RFAXAPI RFaxGetVerifiedUserToken(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usReserved,
 OUT RFAXSTR pszVerificationToken,
 IN DWORD dwVerificationTokenSize,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usReserved	Reserved.
pszVerificationToken	Buffer for verification token.
dwVerificationTokenSize	Number of characters that can be written to pszVerificationToken.

Return Values

Some of the most common return values are listed in the following table.

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.

Remarks

RFaxLoadUser()

This function loads a user record in the USERINFO_10 structure.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 OUT PUSERINFO_10 lpUI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user to load. Specify 0 if this argument is not used.
lpUserID	User ID of the user to load. Specify an empty string if this argument is not used.
lpRouteInfo	Routing information of the user to load. Specify an empty string if this argument is not used.
lpDSName	Distinguished Name of the user to load. Specify an empty string if this argument is not used.
lpUI10	Returns the USERINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

When the USERINFO_10 structure is loaded, the password field will be blank. This field is write only.

Only one of the arguments hUser, lpUserID, lpRouteInfo, or lpDSName should be used. If multiple arguments are specified, only the first valid argument will be used to search for the user.

See Also

RFaxLoadUser() above

RFaxLoadUser2() below

RFaxLoadUser3() on the next page

RFaxLoadUser4() on page 157

RFaxLoadUser5() on page 158

RFaxLoadUser6() on page 159

RFaxLoadUserByEmail() on page 160

RFaxLoadUserByLoadKey() on page 161

RFaxLoadUserByRouteInfo() on page 162

RFaxLoadUser2()

This function loads a user based on a variety of arguments.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser2(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID,
 OUT PUSERINFO_10 lpUI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify the argument to search on. 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID
hUser	Handle of the user to load.
lpUserID	Pointer to the ID of the user to load.
lpRouteInfo	Pointer to the routing information of the user to load.
lpDSName	Pointer to the Distinguished Name of the user to load.
ulRoutingCode	Routing code information of the user to load.
ulSubscriberID	Subscriber ID of the user to load.
lpUI10	Returns the USERINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

RFaxLoadUser() for comments.

See Also

RFaxLoadUser() on the previous page

RFaxLoadUser2() on the previous page

RFaxLoadUser3() below

RFaxLoadUser4() on the next page

RFaxLoadUser5() on page 158

RFaxLoadUser6() on page 159

RFaxLoadUserByEmail() on page 160

RFaxLoadUserByLoadKey() on page 161

RFaxLoadUserByRouteInfo() on page 162

RFaxLoadUser3()

This function loads a user based on a variety of arguments.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser3(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID,
 IN short sUserInfoLevel,
```

```
    OUT RFAXPVOID lpUI
  );
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify the argument to search on: 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID
hUser	Handle of the user to load.
lpUserID	Pointer to the ID of the user to load.
lpRouteInfo	Pointer to the routing information of the user to load.
lpDSName	Pointer to the Distinguished Name of the user to load.
ulRoutingCode	Routing code information of the user to load.
ulSubscriberID	Subscriber ID of the user to load.
sUserInfoLevel	Specify the information level. This argument dictates which structure was passed in as lpUI. Must be 10 or 11.
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument) to receive the information on the specified user.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

RFaxLoadUser() for comments.

See Also

See Also

RFaxLoadUser() on page 155

RFaxLoadUser2() on page 155

RFaxLoadUser4() below

RFaxLoadUser5() on the next page

RFaxLoadUser6() on page 159

RFaxLoadUserByEmail() on page 160

RFaxLoadUserByLoadKey() on page 161

RFaxLoadUserByRouteInfo() on page 162

RFaxLoadUser4()

This function loads a user based on a variety of arguments.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser4(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID,
 IN short sUserInfoLevel,
 OUT RFAXPVOID lpUI
 IN OPTIONAL USHORT.usLoadOpt
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify the argument to search on: 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID
hUser	Handle of the user to load.
lpUserID	Pointer to the ID of the user to load.
lpRouteInfo	Pointer to the routing information of the user to load.
lpDSName	Pointer to the Distinguished Name of the user to load.
ulRoutingCode	Routing code information of the user to load.
ulSubscriberID	Subscriber ID of the user to load.
sUserInfoLevel	Specify the information level. This argument dictates which structure was passed in as lpUI. Must be 10 or 11.
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument) to receive the information on the specified user.
usLoadOpt	0 or LOADUSER_??? flag.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

RFaxLoadUser() for comments.

See Also

RFaxLoadUser() on page 155

RFaxLoadUser2() on page 155

RFaxLoadUser3() on page 156

RFaxLoadUser4() on the previous page

RFaxLoadUser5() below

RFaxLoadUser6() on the next page

RFaxLoadUserByEmail() on page 160

RFaxLoadUserByLoadKey() on page 161

RFaxLoadUserByRouteInfo() on page 162

RFaxLoadUser5()

This function loads a user based on a variety of arguments.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser4(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL ULONG ulSubscriberID,
 IN short sUserInfoLevel,
 OUT RFAXPVOID lpUI
 IN OPTIONAL USHORT.usLoadOpt
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify the argument to search on: 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = ulSubscriberID
hUser	Handle of the user to load.
lpUserID	Pointer to the ID of the user to load.
lpRouteInfo	Pointer to the routing information of the user to load.
lpDSName	Pointer to the Distinguished Name of the user to load.
ulRoutingCode	Routing code information of the user to load.
ulSubscriberID	Subscriber ID of the user to load.
sUserInfoLevel	Specify the information level. This argument dictates which structure was passed in as lpUI. Must be 10 or 11.
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument) to receive the information on the specified user.
usLoadOpt	0 or LOADUSER_??? flag.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxLoadUser() for comments.

See Also

RFaxLoadUser() on page 155

RFaxLoadUser2() on page 155

RFaxLoadUser3() on page 156

RFaxLoadUser4() on page 157

RFaxLoadUser6() below

RFaxLoadUserByEmail() on the next page

RFaxLoadUserByLoadKey() on page 161

RFaxLoadUserByRouteInfo() on page 162

RFaxLoadUser6()

This function changes the Subscriber ID parameter from 32 bits (in RFaxLoadUser5) to 64 bits.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser4(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpRouteInfo,
 IN OPTIONAL RFAXCSTR lpDSName,
 IN OPTIONAL ULONG ulRoutingCode,
 IN OPTIONAL INT64 biSubscriberID,
 IN short sUserInfoLevel,
 OUT RFAXPVOID lpUI
 IN OPTIONAL USHORT.usLoadOpt
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify the argument to search on: 0 = hUser 1 = lpUserID 2 = lpRouteInfo 3 = lpDSName 4 = ulRoutingCode 5 = biSubscriberID
hUser	Handle of the user to load.
lpUserID	Pointer to the ID of the user to load.
lpRouteInfo	Pointer to the routing information of the user to load.
lpDSName	Pointer to the Distinguished Name of the user to load.
ulRoutingCode	Routing code information of the user to load.
biSubscriberID	Big Subscriber ID of the user to load.
sUserInfoLevel	Specify the information level. This argument dictates which structure was passed in as lpUI. Must be 10 or 11.
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument) to receive the information on the specified user.
usLoadOpt	0 or LOADUSER_??? flag.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxLoadUser() for comments.

See Also

RFaxLoadUser() on page 155

RFaxLoadUser2() on page 155

RFaxLoadUser3() on page 156

RFaxLoadUser4() on page 157

RFaxLoadUser5() on page 158

RFaxLoadUser6() on the previous page

RFaxLoadUserByEmail() below

RFaxLoadUserByLoadKey() on the next page

RFaxLoadUserByRouteInfo() on page 162

RFaxLoadUserByEmail()

This function loads a user record in the USERINFO_11 structure.

C Prototype

```
DWORD RFAXAPI RFaxLoadUser(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpEmailAddress,
 OUT PUSERINFO_11 lpUI11
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpEmailAddress	Email address of the user to load.
lpUI11	Returns the USERINFO_11 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

When the USERINFO_11 structure is loaded, the password field will be blank. This field is write only.

Only one of the arguments hUser or lpEmailAddress should be used. If multiple arguments are specified, only the first valid argument will be used to search for the user.

See Also

RFaxLoadUser() on page 155

RFaxLoadUser2() on page 155

RFaxLoadUser3() on page 156

RFaxLoadUser4() on page 157

RFaxLoadUser5() on page 158

RFaxLoadUser6() on page 159

RFaxLoadUserByLoadKey() below

RFaxLoadUserByRouteInfo() on the next page

RFaxLoadUserByLoadKey()

Deprecated.

This function loads a user record in the USERINFO_11 structure.

C Prototype

```
DWORD RFAXAPI RFaxLoadUserByLoadKey(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpKey,
 IN USHORT loadkey,
 OUT PUSERINFO_11 lpUI11
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpKey	
loadkey	Loadkey.
lpUI11	Returns the USERINFO_11 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

When the USERINFO_11 structure is loaded, the password field will be blank. This field is write only.

See Also

RFaxLoadUser() on page 155

RFaxLoadUser2() on page 155

RFaxLoadUser3() on page 156

RFaxLoadUser4() on page 157

RFaxLoadUser5() on page 158

RFaxLoadUser6() on page 159

RFaxLoadUserByEmail() on page 160

RFaxLoadUserByRouteInfo() below

RFaxLoadUserByRouteInfo()

This function loads a user record in the USERINFO_11 structure.

C Prototype

```
DWORD RFAXAPI RFaxLoadUserByRouteInfo(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpRouteInfo,
 OUT PUSERINFO_11 lpUI11
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpRouteInfo	Routing type for the user.
lpUI11	Returns the USERINFO_11 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

When the USERINFO_11 structure is loaded, the password field will be blank. This field is write only.

See Also

RFaxCreateServerHandle()
RFaxCreateServerHandle2()
RFaxLoadUser2()
RFaxLoadUser3()

RFaxLoadUserProperties()

This function loads the properties of a user.

C Prototype

```
DWORD RFAXAPI RFaxLoadUserProperties(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 OUT PUSERPROPERTIES lpUD
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user whose properties to load. Specify 0 if this argument is not used.
lpUserID	User ID of the user whose properties to load. Specify an empty string if this argument is not used.
lpUD	Pointer to the USERPROPERTIES structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

RFaxSaveUser()

This function saves user information.

C Prototype

```
DWORD RFAXAPI RFaxSaveUser(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewUser,
 IN PUSERINFO_10 lpUI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewUser	Specify if this record is a new user or if you are updating an existing user. 0 = Updating existing user 1 = Saving a new user
lpUI10	Pointer to the USERINFO_10 structure with the information to save.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The password field of the USERINFO_10 object pointer by the pUI10 will be ignored, thus password changes are not allowed. Calls where the USERINFO_11.userid field is changed can be lengthy, because the server will adjust ownership of all the user's phonebook entries. If a user object is loaded with RFaxLoadUser(), modified, and saved back using RFaxSaveUser, care must be taken to preserve the value of the USERINFO_10.handle field. Changing this field can cause the save user operation to fail or, worse, to corrupt another user object.

See Also

RFaxSaveUser2() below

RFaxSaveUser2()

This function saves user information.

C Prototype

```
DWORD RFAXAPI RFaxSaveUser2(
IN RFAXSERVERHANDLE hServer,
IN SHORT fNewUser,
IN short sUserInfoLevel,
IN RFAXPVOID lpUI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewUser	Specify if this record is a new user or if you are updating an existing user. 0 = Updating existing user 1 = Saving a new user
sUserInfoLevel	Specify the information level. This argument dictates which structure was passed in as lpUI. Must be 10 or 11.
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument) to receive the information on the specified user.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The password field of the USERINFO_10 object pointer by the pUI10 will be ignored, thus password changes are not allowed. Calls where

the USERINFO_10.userid field is changed can be lengthy because the server will adjust ownership of all the user's phonebook entries. If a user object is loaded with RFaxLoadUser(), modified, and saved back using RFaxSaveUser(), care must be taken to preserve the value of the USERINFO_10.handle field. Changing this field can cause the save user operation to fail or, worse, to corrupt another user object.

See Also

RFaxSaveUser() on page 162

RFaxSaveUserProperties()

This function saves the properties of a user.

C Prototype

```
DWORD RFAXAPI RFaxSaveUserProperties(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hUser,
 IN OPTIONAL RFAXCSTR lpUserID,
 OUT PUSERPROPERTIES lpUD
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Specify the handle of the user whose properties you want to save. Specify 0 if this argument is not used.
lpUserID	Specify the User ID of the user whose properties you want to save. Specify an empty string if this argument is not used.
lpUD	Pointer to the USERPROPERTIES structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

RFaxUnVerifyUser()

This function signs a user off .

C Prototype

```
DWORD RFAXAPI RFaxUnVerifyUser(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to verify.

Return Values

Return	Result
RFAX_SUCCESS	Success.

RFaxVerifyPassword()

This function verifies a password.

C Prototype

```
DWORD RFAXAPI RFaxVerifyUser(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpPassword
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpPassword	The password to verify.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_BADPASSWORD	The specified password is invalid.
RFAXERR_BADKEY	The user specified cannot be found.
RFAXERR_ACCESSDENIED	The user is not an administrator, and the bAdmin flag was set to True.

Remarks

This function is useful for verifying a login to your application. This function only uses RightFax authentication.

See Also

RFaxVerifyUser2() below

RFaxVerifyVoiceUser() on page 167

RFaxVerifyUser()

This function verifies a user's password and administrative access.

C Prototype

```
DWORD RFAXAPI RFaxVerifyUser(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN RFAXCSTR lpPassword,
 IN int loginType
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to verify.
lpPassword	Specify the password of the user specified in lpUserID to verify.
loginType	One of VERIFYUSER_??? (used to be and still compatible with BOOL bAdmin).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_BADPASSWORD	The specified password is invalid.
RFAXERR_BADKEY	The user specified cannot be found.
RFAXERR_ACCESSDENIED	The user is not an administrator, and the bAdmin flag was set to True.

Remarks

This function is useful for verifying a login to your application. This function only uses RightFax authentication. To use Windows NT authentication, use RFaxVerifyUser2().

See Also

RFaxVerifyVoiceUser() on page 167

RFaxVerifyUser2()

Verifies a user's password and administrative access.

C Prototype

```
DWORD RFAXAPI RFaxVerifyUser2(
IN RFAXSERVERHANDLE hServer,
IN RFAXCSTR lpUserID,
IN RFAXCSTR lpPassword,
IN int loginType,
OUT PLOGININFO pLoginInfo,
IN BOOL bForceNTAuthentication
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to verify.
lpPassword	Specify the password of the user specified in lpUserID to verify.
loginType	One of VERIFYUSER_??? (used to be and still compatible with BOOL bAdmin).
pLoginInfo	If bForceNTAuthentication is True, this argument will return a LOGININFO structure containing login information for the specified user on success. If bForceNTAuthentication is False, this argument is not used.
bForceNTAuthentication	Set to True to use NT authentication. Set to False to use RightFax security.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_BADPASSWORD	The specified password is invalid.
RFAXERR_BADKEY	The specified user cannot be found.
RFAXERR_ACCESSDENIED	The user is not an administrator, and the bAdmin flag was set to True.
RFAXERR_AUTHENTICATION_REQ	The user is set with USERFLAGS2_REQNTAUTHEN, but lpUserID was given (should be blank).
RFAXERR_AUTHENTICATE_FAILURE	There was an error discovering the client's identity at the server.
RFAXERR_INVALID_PARAMETER	The lpUserID argument is Null. and this is a pre-7.0 server.
RFAXERR_INVALID_HANDLE	The hServer argument is invalid.
RFAXERR_NOTSUPPORTED	The bForceNTAuthentication argument is True, but the server is pre-7.0.

Remarks

This function is used to verify a user login to your application using NT authentication (although it can be used for RightFax authentication when bForceNTAuthentication is False.)

When a RightFax user is verified using NT authentication, the API creates a secure connection to the RightFax server, which the server uses to get a security identifier (SID). The SID is then compared to the RightFax user database to identify the RightFax user. This user is returned in the pLoginInfo argument. If bForceNTAuthentication is True, the lpUserID argument is not used for authentication.

If lpUserID is Null or an empty string, the server will use NT authentication, even if bForceNTAuthentication is False.

See Also

RFaxVerifyUser() on the previous page

RFaxVerifyVoiceUser() on the next page

RFaxVerifyVoiceUser()

For voice users using TeleConnect, this function verifies a user's password.

C Prototype

```
DWORD RFAXAPI RFaxVerifyVoiceUser(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpUserID,
 IN RFAXCSTR lpPassword
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUserID	Specify the user ID to verify.
lpPassword	Specify the password of the user specified in lpUserID to verify. Because voice users can only enter digits on their telephone, a fuzzy comparison will be performed based upon the letters printed on telephone keys. "2" will match 2, A, B, or C. "3" will match 3, C, D, or E. "Q" and "Z" will be matched by any digit.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_BADPASSWORD	The specified password is invalid.
RFAXERR_BADKEY	The specified user cannot be found.

Remarks

This function is useful to verify a login from a telephone into you application.

See Also

RFaxVerifyUser() on page 165

Chapter 13: Delegate functions

The delegate functions open a specific delegate or a list of delegates, and add and delete delegates on the RightFax server.

RFaxCombineUserDelegatePermissions()

To simplify permissions checking, this function combines administrative, user, and delegate information into a single delegation. This function neither stores nor loads data. It merely combines permissions from the given sources. This function is consistent with the behavior of FaxUtil. That is, there are no operational guidelines for the behavior of this function other than it combines permissions in a manner that will be consistent with FaxUtil.

C Prototype

```
DWORD RFAXAPI
RFaxCombineUserDelegatePermissions(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXPVOID puserActing,
 IN RFAXPVOID puserDelegator,
 IN SHORT sDelegatorUserInfoLevel,
 IN OPTIONAL RFAXPVOID pdelegRelation,
 IN SHORT sDelegationInfoLevel,
 IN OPTIONAL RFAXPVOID pfax,
 IN SHORT sFaxInfoLevel,
 OUT PDELEGATELISTELEMENT0 pdelegComputed)
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
puserActing	USERINFO or USERLISTELEMENT of the delegated/acting user performing the action. Usually the logged in user.
sActingUserInfoLevel	User structure type at puserActing.
puserDelegator	USERINFO or USERLISTELEMENT of the delegating/owning user owning the resource. The one who delegated use of their faxes, phonebooks, etc.
sDelegatorUserInfoLevel	User structure type at puserDelegator.
pdelegRelation	DELEGATEINFO or DELEGATELISTELEMENT of delegate relation if any.
sDelegationInfoLevel	Delegation structure type at pdelegRelation.
pfax	FAXINFO or FAXLISTELEMENT of fax, if any. If the query is in regard to viewing or other fax-specific operation, the fax should be provided.
sFaxInfoLevel	Fax structure type at pfax.
pdelegComputed	The combination of all other parameters in a virtual delegate relation.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxDeleteDelegate()

This function deletes the specified delegate.

C Prototype

```
DWORD RFAXAPI RFaxDeleteDelegate(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hDelegate
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hDelegate	Handle of the delegate to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes the specified delegate. A handle to a delegate is stored in the DELEGATELISTELEMENT0 structure, which can be obtained by traversing the list returned by RFaxGetDelegateList().

RFaxDeleteDelegateTemplate()

This function deletes the specified delegate template.

C Prototype

```
DWORD RFAXAPI RFaxDeleteDelegateTemplate(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hTemplate
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hTemplate	Handle of the template to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes the specified delegate template. A handle to a template is stored in the DELEGATETEMPLATELISTELEMENT0 structure, which can be obtained by traversing the list returned by RFaxGetDelegateTemplateList().

RFaxGetDelegateList()

This function gets a handle to a list of delegates.

C Prototype

```
DWORD RFAXAPI RFaxGetDelegateList(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL RFAXCSTR pszDelegateeID,
 IN OPTIONAL RFAXCSTR pszDelegatorID,
 IN USHORT usInfoLevel,
 IN BOOL fGroup,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszDelegateeID	Name of a delegate. See Remarks for details.
pszDelegatorID	Name of a delegator. See Remarks for details.
usInfoLevel	Information level. Must be 0, which specifies a list of DELEGATELISTELEMENT0 structures.
fGroup	Specify True if pszDelegateeID is a user group ID.
lpLH	On success, returns a handle to a list of DELEGATELISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets a handle to a list of delegates. Any combination of the optional arguments pszDelegateeID and pszDelegatorID can be specified. The following scenarios are defined:

- Only delegate is specified – all delegate records referring to the specified user as a delegate are returned.
- Only delegator is specified – all delegate records referring to the specified user as a delegator are returned.
- Both are specified – all delegate records referring to both the specified users in their respective role are returned.
- Neither is specified – all delegate records for all users are returned.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetDelegateTemplateList()

This function gets a handle to a list of delegate templates.

C Prototype

```
DWORD RFAXAPI RFaxGetDelegateTemplateList(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL RFAXCSTR pszUserID,
 IN USHORT usInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszUserID	Name of a user. See Remarks for details.
usInfoLevel	Reserved. Must be 0.
lpLH	On success, returns a handle to a list of DELEGATETEMPLATELISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets a handle to a list of delegate templates.

See Also

RFaxCloseListHandle() on page 258

RFaxLoadDelegate()

Loads the specified delegate and stores the information in the provided structure.

C Prototype

```
DWORD RFAXAPI RFaxLoadDelegate(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 OUT PDELEGATEINFO_10 lpDI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the delegate to load.
lpDI10	Pointer to a DELEGATEINFO_10 structure to receive information on the delegate to load.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadDelegate2() below

RFaxSaveDelegate() on the next page

RFaxLoadDelegate2()

Loads the specified delegate.

C Prototype

```
DWORD RFAXAPI RFaxLoadDelegate2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszDelegateeID,
 IN RFAXCSTR pszDelegatorID,
 IN USHORT usInfoLevel,
 IN FAXBOOL fGroup,
 OUT RFAXPVOID lpDI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszDelegateeID	Required delegatee. The user ID or group ID of the user who has been delegated. See fGroup parameter.
pszDelegatorID	Required delegator. User ID of the user who is delegating.
usInfoLevel	Level of data at lpDI. Must be zero or 10.
fGroup	True indicates the delegatee is a group ID. False indicate the delegatee is a user ID.
lpDI	Pointer to a DELEGATEINFO_10 structure to receive information on the delegate to load.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadDelegate() on the previous page

RFaxSaveDelegate() on the next page

RFaxLoadDelegateTemplate()

Loads the specified delegate template and stores the information in the provided structure.

C Prototype

```
DWORD RFAXAPI RFaxLoadDelegateTemplate(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 OUT PDELEGATETEMPLATELISTELEMENT0 lpDI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the template to load.
lpDI10	Pointer to a DELEGATETEMPLATELISTELEMENT0 structure to receive information on the template to load.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxCreateServerHandle() on page 26
RFaxCreateServerHandle2() on page 27
RFaxSaveDelegate() below

RFaxSaveDelegate()

This function saves the specified delegate.

C Prototype

```
DWORD RFAXAPI RFaxSaveDelegate(
 IN RFAXSERVERHANDLE hServer,
 IN PDELEGATEINFO_10 lpDI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpDI	Pointer to a DELEGATEINFO_10 structure containing information on the delegate to save. To modify an existing delegate, the “handle” field should be set to the delegate’s handle. To save a new delegate, the “handle” field should be Null.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function saves the specified delegate. If the delegate information structure contains a valid delegate handle, that delegate will be modified. If the handle is 0, a new delegate will be saved.

See Also

RFaxLoadDelegate() on page 170
RFaxGetDelegateList() on page 169

RFaxSaveDelegateTemplate()

This function saves the specified delegate.

C Prototype

```
DWORD RFAXAPI RFaxSaveDelegateTemplate(
 IN RFAXSERVERHANDLE hServer,
 IN PDELEGATETEMPLATELISTELEMENT0 lpDI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpDl	Pointer to a DELEGATETEMPLATELISTELEMENT0 structure containing information on the delegate template to save. To modify an existing template, the "handle" field should be set to the template's handle. To save a new template, the "handle" field should be Null.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function saves the specified delegate template. If the delegate information structure contains a valid template handle, that delegate will be modified. If the handle is 0, a new delegate will be saved.

See Also

RFaxLoadDelegateTemplate() on page 171*RFaxGetDelegateTemplateList()* on page 170

Chapter 14: Group functions

The group functions open a specific group of users or a list of user groups and add, edit, and delete groups on the RightFax server.

RFaxDeleteGroup()

This function deletes a group.

C Prototype

```
DWORD RFAXAPI RFaxDeleteGroup(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hGroup
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of the group to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

You cannot delete a group that has any members. If you attempt to do so, RFaxDeleteGroup will return RFAXERR_GROUPNOTEMPTY.

See Also

RFaxLoadGroup() on page 180

RFaxDeleteGroupFcsExcluded()

This function deletes fax cover sheet information for a group of users.

C Prototype

```
DWORD RFAXAPI RFaxDeleteGroupFcsExcluded(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sWhichKey
 IN FAXHANDLE hGroup,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0= hGroup
hGroup	Handle of the group to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxDeleteGroupLibDoc()

This function deletes library document information for a group of users.

C Prototype

```
DWORD RFAXAPI RFaxDeleteGroupLibDoc(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sWhichKey
 IN FAXHANDLE hGroup,
 IN FAXHANDLE hLibDoc,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0= hGroup 1=hLibDoc
hGroup	Handle of the group.
hLibDoc	Handle of the library document.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxDeleteGroupPhonebooks()

This function deletes phonebook information for a group of users.

C Prototype

```
DWORD RFAXAPI RFaxDeleteGroupPhonebooks(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sWhichKey
 IN OPTIONAL FAXHANDLE hGroup,
 IN OPTIONAL FAXHANDLE hPhonebook,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0= hGroup 1=hPhonebook
hGroup	Handle of the group.
hPhonebook	Handle of the phonebook.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxFcsDlg1ToReqFieldIndex()

This function finds the REQFIELD_IDX_??? equivalent to the given GRP_FCSDLGCTRL1_???. If there is no relation, REQFIELD_IDX_NONE is returned.

C Prototype

```
DWORD RFAXAPI RFaxFcsDlg1ToReqFieldIndex(
 IN GRP_FCSDLGCTRL1 fcsDlgCtrl,
 OUT int* preqFieldIdx
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of the group to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

REQFIELD_??? on page 464

GRP_FCSDLGCTRL1_??? on page 438

RFaxFcsDlg2ToReqFieldIndex()

This function finds the REQFIELD_IDX_??? equivalent to the given GRP_FCSDLGCTRL2_???. If there is no relation, REQFIELD_IDX_NONE is returned.

C Prototype

```
DWORD RFAXAPI RFaxFcsDlg2ToReqFieldIndex(
 IN GRP_FCSDLGCTRL2 fcsDlgCtrl,
 OUT int* preqFieldIdx
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of the group to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

REQFIELD_??? on page 464

GRP_FCSDLGCTRL2_??? on page 440

RFaxGetGroupExcludedLibDocList()

This function gets a list of Lib docs that a specified group does not have access to.

C Prototype

```
DWORD RFAXAPI RFaxGetGroupExcludedLibDocList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN FAXHANDLE nGroupHandle,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	1 = return a list of LIBDOCLISTELEMENT1. 2 =return a list of LIBDOCLISTELEMENT2.
nGroupHandle	Handle of the group for which to list lib docs the group does not have access to.
lpLH	On success, returns a handle to a list of LIBDOCLISTELEMENT1, orLIBDOCLISTELEMENT2 elements.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetGroupFilteredCoversList()

This function retrieves a list of the fax cover sheet (FCS) documents the group can access.

C Prototype

```
DWORD RFAXAPI RFaxGetGroupFilteredCoversList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN FAXHANDLE nGroupHandle,
 IN LPCTSTR pszFileMask,
 IN BOOL bAllowed,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Specify the information level to load. Set to 0 or 10.
nGroupHandle	Handle of the group.
pszFileMask	Mask for files to list, such as *.*, *.pcl.
bAllowed	Set to True to get the list of files the group is allowed to access Set to False to get the list of files the group is not allowed to access.
lpLH	Returns a handle to a list of COVERSHEETINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that the current user has access to.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetGroupList()

This function loads a list of user groups.

C Prototype

```
DWORD RFAXAPI RFaxGetGroupList(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 OUT RFAXLISTHANDLE *lpLH,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	This argument must be 10.
lpLH	Returns a handle to a list of GROUPINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Loads all of the user groups on a particular RightFax server. This function does not load phonebook groups.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetGroupList2() below

RFaxGetGroupList3() below

RFaxLoadGroup() on page 180

RFaxCloseListHandle() on page 258

RFaxGetGroupList2()

This function loads a list of user groups.

C Prototype

```
DWORD RFAXAPI RFaxGetGroupList2(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL USHORT usGetOptions
 IN short sInfoLevel,
 OUT RFAXLISTHANDLE *lpLH,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usGetOptions	0 or GETGROUPS_??? flag.
sInfoLevel	This argument must be 10.
lpLH	Returns a handle to a list of GROUPINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Loads all of the user groups on a particular RightFax server. This function does not load phonebook groups.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetGroupList() on the previous page

RFaxGetGroupList3() below

RFaxLoadGroup() on page 180

RFaxGetGroupList3()

This function loads a list of user groups.

C Prototype

```
DWORD RFAXAPI RFaxGetGroupList3(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL USHORT usGetOptions
 IN short sInfoLevel,
 IN OPTIONAL RFAXCSTR lpSearchString
 in RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usGetOptions	0 or GETGROUPS_??? flag.
sInfoLevel	This argument must be 10.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Loads all of the user groups on a particular RightFax server. This function does not load phonebook groups.

See Also

RFaxGetGroupList() on page 177

RFaxGetGroupList2() on the previous page

RFaxLoadGroup() on the next page

RFaxGetGroupUserList()

This function loads a list of the users in a group.

C Prototype

```
DWORD RFAXAPI RFaxGetGroupUserList(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszGroupID,
 IN short sInfoLevel,
 IN short sSortIndex,
 IN RFAXCSTR lpSearchString,
 IN USHORT usGetOptions,
 IN USHORT usSearchOptions,
 OUT RFAXLISTHANDLE *lpLH,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszGroupID	ID of the group whose members are to be returned.
sInfoLevel	Must be 0, 1, or 2 (2 requires version 7.0 or above).
sSortIndex	One of USERORDER_???.
lpSearchString	Optional. Search and return only users whose name contains the specified search string.
usGetOptions	Optional. 0 or combination of GETUSERS_??? flags.
usSearchOptions	Optional. 0 or combination of SEARCHUSEROPTS_??? flags. Defaults to SEARCHUSEROPTS_USERNAME if a lpSearchString is passed in.
lpLH	If return == 0, returns a handle to list of USERLISTELEMENT# structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetPagedGroupList()

This function loads a detailed page of RightFax user groups.

C Prototype

```
DWORD RFAXAPI RFaxGetPagedGroupList(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sInfoLevel,
 IN USHORT usGetOptions,
 IN LPCSTR searchWords,
 IN USHORT usOrder,
 IN BOOL fAscending,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
```

```
    OUT LONG plTotal,
    OUT RFAXLISTHANDLE lpLH
  );
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Information level must be 10.
usGetOptions	0 or a combination of *GETGROUPS_???* on page 435 flags.
searchWords	Words to search for the within the list.
usOrder	Order in which to load the paged records. See the *GROUPORDER_???* on page 437 constants.
fAscending	TRUE for ascending; FALSE for descending.
lPageIndex	0-based index of page to return.
usPageSize	Number of groups per page.
plTotal	When not null, receives total records.
lpLH	If return value is 0, returns a handle to list of *GROUPINFO_10* on page 339 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetGroupList() on page 177

RFaxGetGroupList2() on page 178

RFaxGetGroupList3() on page 178

RFaxLoadGroup()

This function loads a group of users.

C Prototype

```
  DWORD RFAXAPI RFaxLoadGroup(
    IN RFAXSERVERHANDLE hServer,
    IN OPTIONAL RFAXCSTR lpGroupID,
    OUT PGROUPINFO_10 lpGI10
  );
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpGroupID	Specify the Group ID that you want to load. If the argument is Null, it will load the group EVERYONE.
lpGI10	Returns the GROUPINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Returns only information on a specified group of users.

See Also

RFaxLoadGroup2() below

RFaxLoadGroup3() on the next page

RFaxLoadGroup2()

This function loads a group of users.

C Prototype

```
DWORD RFAXAPI RFaxLoadGroup2(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL RFAXCSTR lpGroupID,
 IN OPTIONAL USHORT usLoadOpt,
 OUT PGROUPINFO_10 lpGI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpGroupID	Specify the Group ID that you want to load. If the argument is Null, it will load the group EVERYONE.
usLoadOpt	0 or LOADGROUP_??? flags.
lpGI10	Returns the GROUPINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

Returns only information on a specified group of users.

See Also

RFaxLoadGroup() on the previous page

RFaxLoadGroup3() below

RFaxLoadGroup3()

This function loads information for a user group.

C Prototype

```
DWORD RFAXAPI RFaxLoadGroup3(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sWhichKey,
 IN FAXHANDLE hGroup,
 IN RFAXCSTR lpGroupID,
 IN RFAXCSTR lpRoutingCode,
 IN USHORT usLoadOpt,
 OUT PGROUPINFO_10 lpGI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0=hGroup 1=lpGroupID 2=lpRoutingCode
hGroup	Handle of the group.
lpGroupID	Specify the group ID that you want to load. If the argument is NULL, it will load the group EVERYONE.
lpRoutingCode	Specify the routing code of the group to load with sWhichKey = 2; otherwise, NULL.
usLoadOpt	0 or LOADGROUP_??? flags.
lpGI10	Returns the GROUPINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

.

See Also

RFaxLoadGroup() on the previous page

RFaxLoadGroup2() on page 180

RFaxLoadGroupTag()

Load an attribute associated with the specified group.

C Prototype

```
DWORD RFAXAPI RFaxLoadGroupTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hGroup,
 IN RFAXCSTR pszName,
 OUT PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of group to load attribute from.
pszName	Name of the attribute to load.
lpTI10	Returns the details of the loaded attribute.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

RFaxSaveGroup()

This function saves user group information.

C Prototype

```
DWORD RFAXAPI RFaxSaveGroup(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewGroup,
 IN PGROUPINFO_10 lpGI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewGroup	Specify True to add a new group. Specify False to update an existing group.
lpGI10	Pointer to the GROUPINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an old group, verify that fNewGroup is set to False and that the handle member of the GROUPINFO_10 structure is preserved. Otherwise, the function will fail.

See Also

RFaxLoadGroup() on page 180

RFaxSaveGroupFcsExcluded()

This function saves fax cover sheet information for a group of users.

C Prototype

```
DWORD RFAXAPI RFaxSaveGroupFcsExcluded(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hGroup,
```

```
  IN FAXHANDLE hCover,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of the group.
hCover	Handle of the library fax cover sheet.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveGroupLibDoc()

This function saves library document information for a group of users.

C Prototype

```
DWORD RFAXAPI RFaxSaveGroupLibDoc(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hGroup,
 IN FAXHANDLE hLibDoc
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of the group.
hLibDoc	Handle of the library document.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveGroupPhonebooks()

This function saves phonebook information for a group of users.

C Prototype

```
DWORD RFAXAPI RFaxSaveGroupPhonebooks(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hGroup,
 IN FAXHANDLE hPhonebook
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hGroup	Handle of the group.
hLibDoc	Handle of the phonebook.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveGroupTag()

Save an attribute associated with the specified group.

C Prototype

```
DWORD RFAXAPI RFaxSaveGroupTag(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNew,
```

```
    IN PTAGINFO_10 lpTI10
  );
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNew	TRUE if the attribute is new to the group referred to in lpTI10->hOwner.
lpTI10	Returns the details of the attribute to save.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Chapter 15: Icon functions

The icon functions open a list of icon image files and add, edit, and delete icon images on the RightFax server. Icon functions supersede the stamp functions.

RFaxDeleteIcon()

This function deletes an icon definition.

C Prototype

```
DWORD RFAXAPI RFaxDeleteIcon(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hIcon,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hIcon	Specify the handle of the icon to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxSaveIcon() on page 187

RFaxLoadIcon() on the next page

RFaxGetDefaultIcon()

This function loads a default icon image from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetIconList(
 IN RFAXSERVERHANDLE hServer,
 IN USERINFO_11* puserinfo,
 IN USERLISTELEMENT1* puserelem,
 IN RFAXCSTR pszFileName,
 IN USHORT usOptions
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
puserinfo	User information. Either puserinfo or puserelem must be specified.

puserelem	User element 1 or 2. You must specify puserinfo or puserelem.
pszFileName	Full path to the file where icon data should be written
usOptions	Reserved for future use.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxGetIconList() below

RFaxGetIconList()

This function loads a list of icon images from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetIconList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usOptions,
 OUT RFAXLISTHANDLE *lpLH,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Handle of owning user or NULL for unowned.
usOptions	Reserved for future use. Specify 0.
lpLH	If return = 0, returns a handle to a list of ICONINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Returns a list all of the icons on the RightFax server.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetDefaultIcon() on the previous page

RFaxLoadIcon()

This function loads icon information.

C Prototype

```
DWORD RFAXAPI RFaxLoadIcon(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 OUT PICONINFO_10 piconinfo,
 IN RFAXCSTR pszFileName,
 IN USHORT usOptions
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of icon or icon owner. See LOADICONOPT_???.
piconinfo	Optional pointer to ICONINFO_10 to fill with icon information or NULL to not load icon information.

pszFileName	Optional full path to the file where icon data should be written or NULL to not download icon data.
usOptions	Icon loading options. See LOADICONOPT_???.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

See Also

RFaxGetDefaultIcon() on page 185

RFaxGetIconList() on the previous page

RFaxSaveIcon()

This function saves a new icon image definition or updates an existing one.

C Prototype

```
DWORD RFAXAPI RFaxSaveIcon(
 IN RFAXSERVERHANDLE hServer,
 IN PICONINFO_10 pinfo,
 IN RFAXCSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pinfo	Icon information. See ICONINFO_10.
pszFileName	Optional local file containing icon data or NULL if data does not need to be updated.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadIcon() on the previous page

RFaxDeleteIcon() on page 185

Chapter 16: Signature functions

The signature functions open a list of signature files and add, edit, and delete signatures on the RightFax server.

RFaxDeleteSig()

This function deletes the specified signature.

C Prototype

```
DWORD RFAXAPI RFaxDeleteSig(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hSig,
 IN BOOL fRemoveImageFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hSig	Handle of the signature to delete.
fRemoveImageFile	Specify False to remove the signature from the database. Specify True to delete the image file as well.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Deletes the specified signature. A handle to a signature is stored in the SERVERINFO structure that can be obtained by traversing the list returned by RFaxGetSigList().

RFaxGetSigList()

This function gets a handle to a list of signatures.

C Prototype

```
DWORD RFAXAPI RFaxGetSigList(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Information level desired. Must be 0, which specifies a list of SERVERINFO structures.
lpLH	On success, returns a handle to a list of SERVERINFO structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets a handle to a list of signatures.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadSig()

This function loads the specified signature.

C Prototype

```
DWORD RFAXAPI RFaxLoadSig(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhich,
 IN OPTIONAL FAXHANDLE hSig,
 IN OPTIONAL RFAXCSTR lpSignID,
 OUT PSIGINFO_10 lpSI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhich	Specify 0 to identify the signature to load by hSig. Specify 1 to identify the signature to load by lpSignID.
hSig	If sWhich is 0, specify the handle of the signature to load. Otherwise, specify 0.
lpSignID	If sWhich is 1, specify the ID of the signature to load. Otherwise, specify 0.
lpSI10	Pointer to a SIGINFO_10 structure to receive information on the signature to load.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

Loads the signature. The signature can be identified by either a handle passed as the hSig argument or an identifier of a signature passed as the lpSignID argument.

See Also

RFaxSaveSig() below

RFaxSaveSig()

This function saves the specified signature.

C Prototype

```
DWORD RFAXAPI RFaxSaveSig(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewSig,
 IN PSIGINFO_10 lpSI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewSig	Specify True if this is a new signature. If False is specified, the “handle” member of the SIGINFO_10 structure must be a valid signature handle.
lpSI10	Pointer to a SIGINFO_10 structure containing information on the signature to save. See Remarks for details.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the specified signature.

The “signid” and “signowner” members of the signature info structure must be set, or the function will fail.

See Also

RFaxLoadSig() on the previous page

RFaxGetSigList() on page 188

Chapter 17: Stamp functions

The stamp functions are deprecated. The functions are superseded by the icon functions. See *Icon functions* on page 185.

The stamp functions open a list of stamp files and add, edit, and delete stamps on the RightFax server.

RFaxDeleteStamp()

Deprecated.

This function deletes a stamp definition.

C Prototype

```
DWORD RFAXAPI RFaxDeleteStamp(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hStamp,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hStamp	Specify the handle of the stamp to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxLoadStamp() on the next page

RFaxSaveForm() on page 196

RFaxGetStampList()

Deprecated.

This function loads a list of stamps from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetStampList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usOpt,
 OUT RFAXLISTHANDLE *lpLH,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Handle of owning user or NULL for unowned.
usOpt	Reserved for future use. Specify 0.

lpLH	Returns a handle to a list of STAMPINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Returns a list all of the stamps on the RightFax server.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadStamp()

Deprecated.

This function loads stamp information.

C Prototype

```
DWORD RFAXAPI RFaxLoadStamp(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 OUT OPTIONAL PSTAMPINFO_10 pstampinfo,
 IN OPTIONAL RFAXCSTR pszFileName,
 IN USHORT usOptions
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of stamp or stamp owner. See LOADSTAMPOPT_OWNER.
pstampinfo	Optional pointer to STAMPINFO_10 to fill with stamp information or NULL to not load stamp information.
pszFileName	Optional full path to file where stamp data should be written or NULL to not download stamp.
usOptions	Stamp loading options. See LOADSTAMPOPT_???.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

RFaxSaveStamp()

Deprecated.

This function saves a new stamp definition or updates an existing one.

C Prototype

```
DWORD RFAXAPI RFaxSaveStamp(
 IN PSTAMPINFO_10 pinfo,
 IN OPTIONAL RFAXCSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pinfo	Stamp information. See STAMPINFO_10.
pszFileName	Optional local file containing stamp data or NULL if data does not need to be updated.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadStamp() on the previous page

RFaxDeleteStamp() on page 191

Chapter 18: Form functions

The form functions open a specific overlay form or a list of overlay forms and add, edit, and delete forms on the RightFax server.

RFaxDeleteForm()

This function deletes a form definition.

C Prototype

```
DWORD RFAXAPI RFaxDeleteForm(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hForm,
 IN BOOL fRemoveImageFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hForm	Specify the handle of the form to delete.
fRemoveImageFile	Specify 1 if the server should remove the form image along with the form definition. Specify 0 if the form definition should be deleted from the fax server database, yet the image should be preserved. Passing 0 is useful when a new form definition will be created later using the same image file.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Multiple form definitions can use the same image file. For example, form X1 can use image B0001234.301 and form X2 also uses B0001234.301. If the X1 form definition is deleted and fRemoveImageFile is 1, then the server will also delete the B0001234.301 form image. The form X2 definition is now incomplete because the image file B0001234.301 has been removed. The caller to RFaxDeleteForm() is responsible for dealing with this issue; the server does not track the usage count for form image files.

See Also

RFaxLoadForm() on the next page

RFaxSaveForm() on page 196

RFaxGetFormList()

This function loads a list of forms from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetFormList(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXLISTHANDLE *lpLH,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLH	Returns a handle to a list of FORMLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Returns a list all of the forms on the RightFax server.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadForm()

Loads a form definition from the server.

C Prototype

```
DWORD RFAXAPI RFaxLoadForm(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhich,
 IN OPTIONAL FAXHANDLE hForm,
 IN OPTIONAL RFAXCSTR lpFormCode,
 IN OPTIONAL short sFormNum,
 OUT PFORMINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhich	Specify the information to load: 0 = hForm 1 = lpFormCode/sFormCode 2 = sFormNum/iFormNum
hForm	Specify the handle of the form to load (when sWhich = 0). When sWhich is not 0, then pass a value of 0.
lpFormCode	Specify the form ID of the form to load (when sWhich = 1). When sWhich is not 1, then pass a Null pointer or an empty string "".
sFormNum	Specify the form number of the form to load (when sWhich = 2). When sWhich is not 2, then pass a zero for this argument.
lpFI10	Returns the FORMINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Loading a form definition does not load the form image as well. If the caller needs the form image, it should load the image separately from the RightFax Papers folder using the RFaxFileRead() function.

See Also

RFaxSaveForm() on the next page

RFaxDeleteForm() on the previous page

RFaxFileRead() on page 35

RFaxSaveForm()

This function saves a new form definition or updates an existing one.

C Prototype

```
DWORD RFAXAPI RFaxSaveForm(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewForm,
 IN PFORMINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewForm	Specify True if you are adding a new form definition. Specify False if you are updating an existing form definition.
lpFI10	Pointer to the FORMINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an existing form definition, verify that fNewForm is set to False and that the handle member of the See "FORMINFO_10" structure is preserved. Otherwise the function will fail.

Saving or updating a form definition does not save the actual form image. The form image file must already exist in the RightFax Papers folder or must be placed there by the caller. RFaxFileWrite() can be used to save or update a form image.

See Also

RFaxLoadForm() on the previous page

RFaxDeleteForm() on page 194

Chapter 19: Printer functions

The printer functions open a specific printer or a list of printers and add, edit, and delete printers on the RightFax server.

The device functions load MFP devices and add, edit, and delete them on the RightFax server.

RFaxDeletePrinter()

This function deletes a printer.

C Prototype

```
DWORD RFAXAPI RFaxDeletePrinter(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hPrinter
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hPrinter	Specify the handle of the printer to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Deletes the printer specified by the hPrinter field.

See Also

RFaxLoadPrinter() on the next page

RFaxGetPrinterList() below

RFaxSavePrinter() on the next page

RFaxGetPrinterList()

This function retrieves a list of printers on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetPrinterList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Specify the information level to load. This must be set to 0.
lpLH	Returns a handle to a list of PRINTERLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use RFaxLoadPrinter() to get information about a particular printer.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadPrinter()

This function loads printer information.

C Prototype

```
DWORD RFAXAPI RFaxLoadPrinter(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hPrinter,
 IN OPTIONAL RFAXCSTR lpPrinterID,
 OUT PPRINTERINFO_10 lpPI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hPrinter	Specify the handle of the printer that to load. Optional.
lpPrinterID	Specify the ID of the printer to load. Optional.
lpPI10	Returns a PRINTERINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

You must specify one and only one of the hPrinter or lpPrinterID arguments. The unused argument must be set to zero (0) or Null.

RFaxSavePrinter()

This function updates or saves a new printer object.

C Prototype

```
DWORD RFAXAPI RFaxSavePrinter(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewPrinter,
 IN PPRINTERINFO_10 lpPI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewPrinter	Specify True if you are adding a new printer. Specify False if you are updating an existing printer.
lpPI10	Pointer to a PRINTERINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an existing printer, verify that fNewPrinter is set to False and that the handle member of the PRINTERINFO_10 structure is preserved. Otherwise, the function will fail.

See Also

RFaxLoadPrinter() above

Chapter 20: Billing code functions

The billing code functions let you open a specific billing code or a list of billing codes and add, edit, and delete billing codes on the RightFax server.

RFaxDeleteBillCode()

Deletes a billing code entry.

C Prototype

```
DWORD RFAXAPI RFaxDeleteBillCode(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhich,
 IN FAXHANDLE hCode,
 IN OPTIONAL RFAXCSTR lpBI1,
 IN OPTIONAL RFAXCSTR lpBI2
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
sWhich	Specify which information you want to load with: 0 = hCode 1 = lpBI1 + lpBI2
hCode	Specify the handle of the billing code that you wish to delete (when sWhich = 0).
lpBI1	First billing code ID (when sWhich = 1). Note that lpBI2 must also be supplied when sWhich = 1. An empty string, "", may be passed for lpBI2 if there is no second billing code ID.
lpBI2	Second billing code ID (when sWhich = 1). Note that lpBI1 must also be supplied when sWhich = 1. An empty string, "", may be passed for lpBI1 if there is no first billing code ID.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadBillCode() on page 201

RFaxSaveBillCode() on page 201

RFaxGetBillCodeList()

Loads a list of billing codes.

C Prototype

```
DWORD RFAXAPI RFaxGetBillCodeList(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpSearchString,
 IN unsigned short usKeyField,
```

```
    IN short sInfoLevel,
    IN unsigned long ulMaxRead,
    OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
lpSearchString	Specify a string to have RightFax filter the billing codes on. Specify Null or "" to load all billing codes. See Remarks for more details.
usKeyField	Specify one of the BILLSEARCHKEY_??? constants. This field specifies which billing code fields lpSearchString should be compared to during search.
sInfoLevel	Requested information level. Must be set to zero (0)
ulMaxRead	Specify the maximum number of codes to return.
lpLH	Depending on the value of usInfoLevel, returns a handle to a list of FAXLISTELEMENT0, FAXLISTELEMENT1, or FAXLISTELEMENT2structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function can be used to filter a billing code list, by specifying a search string in the lpSearchString argument. Each code returned will begin with the specified string. For example, if "00" is specified, "00132" would be considered a match.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetBillCodeList2() below

RFaxGetBillCodeList2()

Retrieves a billing code list from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetBillCodeList2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpSearchString,
 IN unsigned short usKeyField,
 IN short sInfoLevel,
 IN unsigned long ulMaxRead,
 IN unsigned short usDirection,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
lpSearchString	String to filter first code to return by. Specify Null or "" to load all billing codes. See Remarks for details.
usKeyField	Specify one of the BILLSEARCHKEY_??? constants. This field specifies which billing code fields lpSearchString should be compared to during search.
sInfoLevel	Requested information level. This must be 0.
ulMaxRead	Maximum number of codes to return.
usDirection	Specify 0 to go forward or 1 to go reverse. See Remarks for details.
lpLH	Returns a handle to a list of BILLLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to RFaxGetBillCodeList() except, once a match is found, that match and the next (or previous dependent upon usDirection) ulMaxRead codes are returned whether or not they match lpSearchString. The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetBillCodeList() on page 199

RFaxLoadBillCode()

Loads a billing code description entry from the server.

C Prototype

```
DWORD RFAXAPI RFaxLoadBillCode(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhich,
 IN OPTIONAL FAXHANDLE hCode,
 IN OPTIONAL RFAXCSTR lpBI1,
 IN OPTIONAL RFAXCSTR lpBI2,
 OUT PBILLINFO_10 lpBI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhich	Specify which information you want to load with. See BILLCODE_LOAD_???.
hCode	Specify the handle of the billing code that you wish to load (when sWhich = 0).
lpBI1	First billing code ID (when sWhich = 1). lpBI2 must also be supplied when sWhich = 1. An empty string, “”, may be passed for lpBI2 if there is no second billing code ID.
lpBI2	Second billing code ID (when sWhich = 1). lpBI1 must be supplied when sWhich = 1. An empty string, “”, may be passed for lpBI1 if there is no first billing code ID.
lpBI10	Returns a BILLINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveBillCode()

Allows you to update or save a new billing code entry.

C Prototype

```
DWORD RFAXAPI RFaxSaveBillCode(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewCode,
 IN PBILLINFO_10 lpBI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
fNewCode	Specify True if you are adding a new billing code, otherwise specify False if you are updating an existing billing code.

lpBI10	Pointer to a BILLINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an existing billing code then make sure fNewCode is set to False and that the handle member of the BILLINFO_10 structure is preserved, otherwise the function will fail.

See Also

RFaxLoadBillCode() on the previous page

RFaxDeleteBillCode() on page 199

RFaxSaveOrUpdateBillCode()

Allows you to update or save a new billing code entry.

C Prototype

```
DWORD RFAXAPI RFaxSaveOrUpdateBillCode(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewCode,
 IN SHORT fUpdateIfExist,
 IN PBILLINFO_10 lpBI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
fNewCode	Specify True if you are adding a new billing code, otherwise specify False if you are updating an existing billing code.
fUpdateIfExist	If the billing code exists, then update it.
lpBI10	Pointer to a BILLINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an existing billing code then make sure fNewCode is set to False and that the handle member of the BILLINFO_10 structure is preserved, otherwise the function will fail.

See Also

RFaxLoadBillCode() on the previous page

RFaxDeleteBillCode() on page 199

Chapter 21: Fax cover sheet functions

The fax cover sheet function opens a list of fax cover sheets on the RightFax server.

RFaxDeleteCoversheet()

This function deletes the coversheet.

C Prototype

```
DWORD RFaxDeleteCoversheet(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFCS,
 IN BOOL fRemoveImageFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFCS	Handle to fax cover sheet (for example. from FaxLoadCoversheet())
fRemoveImageFile	0=keep FCS image file 1=remove FCS image file

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetFCSList()

This function retrieves a list of fax cover sheet (FCS) documents on the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetFCSList(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLH	Returns a handle to a list of strings where each member is a coversheet.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that are located in the \RightFax\FCS folder.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFCSList2() below

RFaxGetFCSList3() below

RFaxGetFCSList2()

This function retrieves a list of fax cover sheet (FCS) documents that the current user has access to.

C Prototype

```
DWORD RFAXAPI RFaxGetFCSList2(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLH	Returns a handle to a list of strings where each member is a coversheet. See FILELISTELEMENT0

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that the current user has access to.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFCSList() on the previous page

RFaxGetFCSList3() below

RFaxGetFCSList3()

This function retrieves a list of fax cover sheet (FCS) documents for the specified group.

C Prototype

```
DWORD RFAXAPI RFaxGetFCSList3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE nGroupHandle
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
nGroupHandle	Handle of the group.
lpLH	Returns a handle to a list of strings where each member is a coversheet.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that the current user has access to.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFCSList() on page 203

RFaxGetFCSList2() on the previous page

FaxLoadCoversheet()

This function loads the coversheet.

C Prototype

```
DWORD RFaxLoadCoversheet(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhich,
 IN OPTIONAL FAXHANDLE hFCS,
 IN OPTIONAL RFAXCSTR lpFCSID,
 OUT PCOVERSHEETINFO_10 lpCI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhich	0 = load by handle 1 = load by ID 2 = load by filename 3 = load by short file name 4 = load system default
hFCS	Handle to coversheet to load if sWhich=0, else 0L
lpFCSID	Coversheet ID if sWhich=1 filename if sWhich = 2 short file name if sWhich = 3 else NULL
lpCI10	If successful, returns the COVERSHEETINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that are located in the \RightFax\FCS folder.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxSaveCoversheet()

This function saves the coversheet.

C Prototype

```
DWORD RFaxSaveCoversheet(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewFCS,
 IN PCOVERSHEETINFO_10 lpCI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewFCS	0=update existing cover sheet 1=new cover sheet
lpCI10	COVERSHEETINFO_10 data to create/update. On return contains FCS handle and short file name.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that are located in the \RightFax\FCS folder.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxSetDefaultCoversheet()

This function sets the default coversheet.

C Prototype

```
DWORD RFaxSetDefaultCoversheet(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFCS,
 IN USHORT usOptions
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFCS	Handle to coversheet,
usOptions	Reserved for future use; must pass 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The server returns a list of all cover sheet files that are located in the \RightFax\FCS folder.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

Chapter 22: Library document functions

The library document functions open a specific library document or a list of library documents, create new library documents from faxes, and manipulate library documents on the RightFax server.

RFaxAdjustLibDocUsage()

This function adjusts the usage amount of library documents

C Prototype

```
DWORD RFAXAPI RFaxAdjustLibDocUsage(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hLibDoc,
 IN OPTIONAL RFAXCSTR lpDocumentID,
 IN USHORT usUsageType,
 IN USHORT usUsageValue,
 OUT OPTIONAL PLIBDOCLISTELEMENT1 pLI11
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify 0 to set the library document using the handle or hLibDoc argument. Specify 1 to set the library document using the library document ID or lpDocumentID.
hLibDoc	If sWhichKey is set to 0, specify the handle of the library document. Otherwise, specify Null.
lpDocumentID	If sWhichKey is set to 1, specify the ID of the library document. Otherwise, specify Null.
usUsageType	Specify one of the LIBDOC_USAGE_??? constants.
usUsageValue	Specify 0 to reset the counter, otherwise it will increase by the specified amount.
pLI11	Returns the LIBDOCLISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use the function to increment the counters for library document usage. When a library document is used, call this function to update the RightFax server. Values can also be reset by specifying 0 in the usUsageType argument.

RFaxCombineLibDocs()

This function combines two or more library document into a new fax.

C Prototype

```
DWORD RFAXAPI RFaxCombineLibDocs(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulNumLibDocs,
 IN FAXHANDLE lpDocHandles,
 IN PFAXINFO_10 lpFI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulNumLibDocs	Specify the number of library documents that are to be combined. This is the number of handles at lpDocHandles.
lpDocHandles	Specify an array of handles of the library documents. There should be at least ulNumLibDocs handles in this array.
lpFI10	Specify the FAXINFO_10 structure to get the new data. See Remarks section below for details.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The number of library documents specified by ulNumLibDocs must be the same number of elements specified in lpDocHandles array. This function does not save the fax object and initiate the fax operation. If there is no “faxfilename” in the FAXINFO_10, a file name is acquired from the server and the “faxfilename” field is set to that name. Additionally, the “numpages” field is updated to reflect the pages added by this function. The images for the fax are created on the server. Use one of the RFaxSaveFax() functions to save the fax.

RFaxDeleteLibDoc()

This function deletes a library document.

C Prototype

```
DWORD RFAXAPI RFaxDeleteLibDoc(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hLibDoc,
 IN BOOL fRemoveImageFiles
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hLibDoc	Specify the handle of the library document to delete.
fRemoveImageFiles	Specify 1 if the server should remove the document image along with the document definition. Specify 0 if the document definition should be deleted from the fax server database, yet the image should be preserved. Passing 0 is useful when a new document definition will be created later using the same image file.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Multiple library documents can use the same image file. For example, document X1 can use image B0001234.301 and document X2 also uses B0001234.301. If the X1 document definition is deleted and

fRemoveImageFile is 1, then the server will also delete the B0001234.301 document image. The X2 document definition is now incomplete because the image file B0001234.301 has been removed. The caller to RFaxDeleteLibDoc is responsible for dealing with this issue; the server does not track the usage count for document image files.

See Also

RFaxLoadLibDoc() on page 213

RFaxLoadLibDoc2() on page 214

RFaxLoadLibDoc3() on page 214

RFaxLoadLibDoc4() on page 215

RFaxSaveLibDoc2() on page 216

RFaxSaveLibDoc3() on page 216

RFaxGetLibDocList()

This function retrieves a list of library documents from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxGetLibDocList(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLH	Returns a handle to a list of LIBDOCLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Returns a list of all the library documents in the system. Use RFaxGetLibDocList2() if you want to filter the library document list on a variety of arguments.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetLibDocList3() on the next page

RFaxGetLibDocList4() on page 211

RFaxGetLibDocList5() on page 211

RFaxGetLibDocList2()

This function loads a list of library documents with a choice of information levels and filter levels.

C Prototype

```
DWORD RFAXAPI RFaxGetLibDocList2(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulLoadMask,
 IN USHORT usInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

ulLoadMask	Specify one of the GFLAG_??? constants to filter the list of library documents.
usInfoLevel	Specify 0, 1, or 2 to specify the level of required information.
lpLH	Returns a handle to a list of LIBDOCLISTELEMENT0, LIBDOCLISTELEMENT1, or LIBDOCLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns a list of all the library documents in the system.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetLibDocList() on the previous page

RFaxGetLibDocList3() below

RFaxGetLibDocList4() on the next page

RFaxGetLibDocList5() on the next page

RFaxGetLibDocList3()

This function loads a list of library documents with a choice of information levels and a search on the library documents.

C Prototype

```
DWORD RFAXAPI RFaxGetLibDocList3(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulLoadMask,
 IN USHORT usInfoLevel,
 IN ULONG ulLibEnumFlags,
 IN RFAXCSTR lpSearchString,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulLoadMask	Specify one of the GFLAG_??? constants to filter the list of library documents.
usInfoLevel	Specify 1 or 2 to specify the level of required information.
ulLibEnumFlags	Specify one of the LIBENUMFLAGS_??? constants.
lpSearchString	Specify string to search on.
lpLH	Returns a handle to a list of LIBDOCLISTELEMENT1 or LIBDOCLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The argument in lpSearchString will search on the field specified in ulLibEnumFlags.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetLibDocList() on page 209

RFaxGetLibDocList2() on page 209

RFaxGetLibDocList4() below

RFaxGetLibDocList5() below

RFaxGetLibDocList4()

This function gets a list of library documents with a choice of information levels and a search on the library documents.

C Prototype

```
DWORD RFAXAPI RFaxGetLibDocList4(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulLoadMask,
 IN USHORT usInfoLevel,
 IN ULONG ulLibEnumFlags,
 IN USHORT usAdminFlags,
 IN OPTIONAL RFAXCSTR lpSearchString,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulLoadMask	Specify one of the GFLAG_??? constants to filter list of the library documents.
usInfoLevel	Specify 1 or 2 for LIBDOCLISTELEMENT1 or LIBDOCLISTELEMENT2, respectively, for the resultant list.
ulLibEnumFlags	Specify one of the LIBENUMFLAGS_??? constants.
usAdminFlags	Specify one of the LIBENUM_ADMINLOAD_??? constants.
lpSearchString	Specify the string to search on. This field corresponds to the field specified via the ulLibEnumFlags field.
lpLH	Returns a handle to a list of LIBDOCLISTELEMENT1 or LIBDOCLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The argument in lpSearchString will search on the field specified in ulLibEnumFlags. The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetLibDocList() on page 209

RFaxGetLibDocList2() on page 209

RFaxGetLibDocList3() on the previous page

RFaxGetLibDocList5() below

RFaxCloseListHandle() on page 258

RFaxGetLibDocList5()

This function gets a list of library documents with a choice of information levels and a search on the library documents.

C Prototype

```
DWORD RFAXAPI RFaxGetLibDocList5(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulLoadMask,
 IN USHORT usInfoLevel,
 IN ULONG ulLibEnumFlags,
 IN USHORT usAdminFlags,
```

```
    IN OPTIONAL RFAXCSTR lpSearchString,
    OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
ulLoadMask	Specify one of the GFLAG_??? constants to filter list of the library documents.
usInfoLevel	Specify 1 or 2 for LIBDOCLISTELEMENT1 or LIBDOCLISTELEMENT2, respectively, for the resultant list.
ulLibEnumFlags	Specify one of the LIBENUMFLAGS_??? constants.
usAdminFlags	Specify one of the LIBENUM_ADMINLOAD_??? constants.
lpSearchString	Specify the string to search on. This field corresponds to the field specified via the ulLibEnumFlags field.
lpLH	Returns a handle to a list of LIBDOCLISTELEMENT1 or LIBDOCLISTELEMENT2 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The argument in lpSearchString will search on the field specified in ulLibEnumFlags. The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetLibDocList() on page 209

RFaxGetLibDocList2() on page 209

RFaxGetLibDocList3() on page 210

RFaxGetLibDocList4() on the previous page

RFaxLibDocFromFax()

This function stores a fax as a library document.

C Prototype

```
DWORD RFAXAPI RFaxLibDocFromFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN RFAXCSTR pszCode,
 IN RFAXCSTR pszDesc,
 IN OPTIONAL FAXHANDLE hLibDoc
);
```

Arguments

hServer	Handle of RightFax server creating by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to create as a library document.
pszCode	String indicating the ID of the library document.
pszDesc	String indicating the description of the library document.
hLibDoc	To replace an existing library document with this new one, specify a valid handle to an existing library document. Otherwise specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is useful for storing faxes as library documents. This function will not store the cover sheet if this is a sent fax. It will store the cover sheet if this is a received document. If the received fax has a cover sheet, use RFaxCombineFaxes() to remove it.

See Also

RFaxLibDocFromFax2() below

RFaxLibDocFromFax2()

This function stores a fax as a library document.

C Prototype

```
DWORD RFAXAPI RFaxLibDocFromFax2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN PLIBDOCLISTELEMENT2 pLibdoc
);
```

Arguments

hServer	Handle of the RightFax server creating by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of the fax to create as a library document.
pLibdoc	Pointer to a LIBDOCLISTELEMENT2. Must have at least the "objectid" field filled in. If the "handle" field has a value, the corresponding library document will be replaced with this one.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Useful for storing faxes as library documents. This function will not store the cover sheet if this is a sent fax. It will store the cover sheet if this is a received document. If the received fax has a cover sheet, use RFaxCombineFaxes() to remove it.

See Also

RFaxLibDocFromFax() on the previous page

RFaxLoadLibDoc()

This function loads information about a library document.

C Prototype

```
DWORD RFAXAPI RFaxLoadLibDoc(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hLibDoc,
 OUT PLIBDOCLISTELEMENT0 lpLibDoc
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hLibDoc	Specify the handle of the library document.
lpLibDoc	Return a LIBDOCLISTELEMENT0 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

This function loads information about a particular library document. Use one of the RFaxGetLibDocList() functions to get a list of available library documents.

See Also

RFaxLoadLibDoc2() below

RFaxLoadLibDoc3() below

RFaxLoadLibDoc4() on the next page

RFaxLoadLibDoc2()

This function loads a library document from the handle of the library document or the ID of the library document.

C Prototype

```
DWORD RFAXAPI RFaxLoadLibDoc2(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hLibDoc,
 IN OPTIONAL RFAXCSTR lpDocumentID,
 OUT PLIBDOCLISTELEMENT0 lpLibDoc
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify 0 to load using the handle or hLibDoc argument. Specify 1 to load using the ID of the library document or lpDocumentID argument.
hLibDoc	If sWhichKey is set to 0, then specify the handle the handle of the library document. Otherwise, specify Null.
lpDocumentID	If sWhichKey is set to 1, then specify the document ID of the library document. Otherwise, specify Null.
lpLibDoc	Return a LIBDOCLISTELEMENT0 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxLoadLibDoc() except you can choose to load the library document by the document ID or the handle of the library document.

See Also

RFaxLoadLibDoc() on the previous page

RFaxLoadLibDoc3() below

RFaxLoadLibDoc4() on the next page

RFaxLoadLibDoc3()

This function loads a library document from the handle of the library document or the ID of the library document. You can specify intended use arguments.

C Prototype

```
DWORD RFAXAPI RFaxLoadLibDoc3(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL short sIntendedUse,
 IN OPTIONAL FAXHANDLE hLibDoc,
 IN OPTIONAL RFAXCSTR lpDocumentID,
 OUT PLIBDOCLISTELEMENT0 lpLibDoc
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

sWhichKey	Specify 0 to load using the handle or hLibDoc argument. Specify 1 to load using the ID of the library document or lpDocumentID argument.
sIntendedUse	Specify one of the LIBDOC_USAGE_??? constants.
hLibDoc	If sWhichKey is set to 0, then specify the handle the handle of the library document. Otherwise specify 0.
lpDocumentID	If sWhichKey is set to 1, then specify the document ID of the library document. Otherwise specify an empty string.
lpLibDoc	Returns a LIBDOCLISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxLoadLibDoc2(), except it returns a LIBDOCLISTELEMENT1 structure. You can specify the indented use in the sIntendedUse argument.

See Also

RFaxLoadLibDoc() on page 213

RFaxLoadLibDoc2() on the previous page

RFaxLoadLibDoc4() below

RFaxLoadLibDoc4()

This function loads a library document from the handle of the library document or the ID of the library document. You can specify intended use arguments.

C Prototype

```
DWORD RFAXAPI RFaxLoadLibDoc3(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN short sIntendedUse,
 IN FAXHANDLE hLibDoc,
 IN RFAXCSTR lpDocumentID,
 OUT PLIBDOCLISTELEMENT0 lpLibDoc
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	Specify 0 to load using the handle or hLibDoc argument. Specify 1 to load using the ID of the library document or lpDocumentID argument.
sIntendedUse	Specify one of the LIBDOC_USAGE_??? constants.
hLibDoc	If sWhichKey is set to 0, then specify the handle the handle of the library document. Otherwise specify 0.
lpDocumentID	If sWhichKey is set to 1, then specify the document ID of the library document. Otherwise specify an empty string.
usLoadOpt	See *LOADLIB_???* on page 447.
lpLibDoc	Returns a LIBDOCLISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

Similar to RFaxLoadLibDoc2(), except it returns a LIBDOCLISTELEMENT1 structure. You can specify the indented use in the sIntendedUse argument.

See Also

RFaxLoadLibDoc() on page 213

RFaxLoadLibDoc2() on page 214

RFaxLoadLibDoc3() on page 214

RFaxSaveLibDoc2()

This function saves the specified library document.

C Prototype

```
DWORD RFAXAPI RFaxSaveLibDoc2(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewDoc,
 IN PLIBDOCLISTELEMENT0 lpLE0
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewDoc	Specify True if this is a new library document. If False is specified, the "handle" member of the LIBDOCLISTELEMENT0 structure must be a valid signature handle.
lpLE0	Pointer to a LIBDOCLISTELEMENT0 structure containing information on the library document to save. See Remarks for details.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the specified library document. If this is a new document, specify True for fNewDoc. If False, the "handle" field must contain a valid handle of the library document to modify.

See Also

RFaxSaveLibDoc3() below

RFaxLoadLibDoc() on page 213

RFaxLoadLibDoc2() on page 214

RFaxLoadLibDoc3() on page 214

RFaxGetLibDocList() on page 209

RFaxSaveLibDoc3()

This function saves the specified library document.

C Prototype

```
DWORD RFAXAPI RFaxSaveLibDoc3(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewDoc,
 IN PLIBDOCLISTELEMENT1 lpLE1
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewDoc	Specify True if this is a new library document. If False is specified, the "handle" member of the LIBDOCLISTELEMENT1 structure must be a valid signature handle.
LpLE1	Pointer to a LIBDOCLISTELEMENT1 structure containing information on the library document to save. See Remarks for details.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the specified library document. If this is a new document, specify True for fNewDoc. If False, the “handle” field must contain a valid handle of the library document to modify.

See Also

RFaxSaveLibDoc2() on the previous page

RFaxLoadLibDoc() on page 213

RFaxLoadLibDoc2() on page 214

RFaxLoadLibDoc3() on page 214

RFaxGetLibDocList() on page 209

Chapter 23: SMS functions

The SMS functions open a list of SMS services and add, edit, and delete SMS services on the RightFax server.

RFaxDeleteSMSService()

This function deletes the specified SMS service.

C Prototype

```
DWORD RFAXAPI RFaxDeleteSMSService(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hService,
 IN OPTIONAL RFAXCSTR lpServiceID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hService	Optional SMS service handle. Pass 0 or Null if not using this argument. If this argument is not used, the lpServiceID argument must be used.
lpServiceID	Optional SMS service identifier. Pass 0 or Null if not using this argument. If this argument is not used, the hService argument must be used.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Deletes the specified SMS service. The service can be identified by either a handle to an SMS service passed as the hService argument or an identifier of an SMS service passed as the lpServiceID argument.

See Also

RFaxGetSMSServiceList() below

RFaxGetSMSServiceList()

This function gets a handle to a list of SMS services.

C Prototype

```
DWORD RFAXAPI RFaxGetSMSServiceList(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Information level. Must be 0, which specifies a list of SVCLISTELEMENT0 structures, or 10, which specifies a list of SVCINFO_10 structures.
lpLH	On success, returns a handle to a list of SVCINFO_10 or SVCLISTELEMENT0 structures dependent upon the sInfoLevel argument.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets a handle to a list of SMS services.

Note that the “handle” member of the structures in the list may be 0 due to the manner in which the SMS information is stored. If so, the remainder of the SMS functions can accept an SMS ID to identify a service. See the “szServiceID” field of the SVCINFO_10 or SVCLISTELEMENT0 structures.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadSMSService()

This function loads the specified SMS service.

C Prototype

```
DWORD RFAXAPI RFaxLoadSMSService(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hService,
 IN OPTIONAL RFAXCSTR lpServiceID,
 OUT PSVCINFO_10 lpSVC10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hService	Optional SMS service handle. Pass 0 or Null if not using this argument. If this argument is not used, the lpServiceID argument must be used.
lpServiceID	Optional SMS service identifier. Pass 0 or Null if not using this argument. If this argument is not used, the hService argument must be used.
lpSVC10	Pointer to a SVCINFO_10 structure to receive information on the SMS service to load.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

This function loads the specified SMS service. The service can be identified by either a handle to an SMS service passed as the hService argument or an identifier of an SMS service passed as the lpServiceID argument.

See Also

RFaxSaveSMSService() below

RFaxSaveSMSService()

This function saves the specified SMS service.

C Prototype

```
DWORD RFAXAPI RFaxSaveSMSService(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewService,
 IN PSVCINFO_10 lpSVC10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewService	Specify True if this is a new SMS service. This flag is currently ignored but should be specified correctly for future use.
lpSVC10	Pointer to a SVCINFO_10 structure containing information on the SMS service to save. The szServiceID field must contain the ID of the SMS service to save.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Saves the specified SMS service. If the service already exists, it will be modified. If it does not exist, it will be created.

The fNewService argument is currently ignored by the server, but it should be specified for compatibility with future servers.

See Also

RFaxLoadSMSService() on the previous page

RFaxGetSMSServiceList() on page 218

RFaxSubmitSMS()

This function submits a message to be sent by the RightFax Pager service.

C Prototype

```
DWORD RFAXAPI RFaxSubmitSMS(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszUserID,
 IN RFAXCSTR pszPagerServiceID,
 IN RFAXCSTR pszPagerDeviceID,
 IN RFAXCSTR pszMessage,
 IN FAXHANDLE hFax,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszUserID	ID of RightFax user sending message.
pszPagerServiceID	ID of SMS/Pager service to transport this message.
pszPagerDeviceID	ID of destination device.
pszMessage	The message to send.
hFax	Associated fax handle or NULL, if none.
ulReserved	Reserved for future use; must pass 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function sends a message to a specified SMS-capable device vie the RightFax SMS/Pager service.

Chapter 24: Fax number functions

The fax number functions provide information about fax numbers available on a RightFax server.

RFaxDeleteFaxNumber()

Deletes a fax number from the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFaxNumber(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFaxnumber
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFaxnumber	Handle of the fax number to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function call delete the specified fax number and remove the number from the list.

RFaxGetFaxNumberList()

Gets a list of fax numbers.

C Prototype

```
DWORD RFAXAPI RFaxGetFaxNumberList(
 IN RFAXSERVERHANDLE hServer,
 IN BOOL fAscending,
 IN USHORT usOrder,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG* plTotal,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fAscending	TRUE for ascending; FALSE for descending.
usOrder	Order in which to load the paged records. See FAXNUMBERORDER_???.
lPageIndex	0-based index of page to return.
usPageSize	Number of records per page.
plTotal	When not null, receives total records.

lpLH	Returns a handle to a list of FAXNUMBERLISTELEMENT structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_NOTFOUND	The faxnumber list is empty.
RFAXERR_INVALID_HANDLE	The hServer argument is invalid. Be sure to call RFaxCreateServerHandle() or RFaxCreateServerHandle2() to obtain a valid RightFax server handle.
RFAXERR_INVALID_PARAMETER	The lpLH argument is Null. Be sure to declare lpLH and pass its address using the address operator (&).

Remarks

This function loads a list of fax numbers in the order of usOrder.

This function returns a FAXNUMBERLISTELEMENT function.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadFaxNumber()

Loads the fax numbers on the specified server.

C Prototype

```
DWORD RFAXAPI RFaxLoadFaxNumber(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hFaxNumber,
 IN OPTIONAL RFAXCSTR lpFaxNumber,
 OUT PFAXNUMBERLISTELEMENT lpFN
);
```

Arguments

hServer	Handle of the RightFax server created by using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0 to load by record handle, hFaxNumbe. 1 to load by fax number string, lpFaxNumber.
hFaxNumber	Handle to the fax number record to load if sWhichKey is 0.
lpFaxNumber	Fax number to load if sWhichKey is 1.
lpFN	Returns the FAXNUMBERLISTELEMENT structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveFaxNumber()

Saves a new fax to be sent or processed by the RightFax server.

C Prototype

```
DWORD RFAXAPI RFaxSaveFaxNumber(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewEntry,
 IN PFAXNUMBERLISTELEMENT lpFN
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

fNewEntry	Adds the entry.
PFAXNUMBERLISTELEMENT	See FAXNUMBERLISTELEMENT..

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSaveFaxNumberExclusions()

For the given set of fax numbers, eliminates access for the specified group's members.

C Prototype

```
DWORD RFAXAPI RFaxSaveFaxNumberExclusions(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usOptions,
 IN FAXHANDLE hGroup,
 IN FAXHANDLE* pahFaxNumber,
 IN SHORT sFaxNumberCount);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usOptions	0 or one of SAVEFAXNUMBEREXCLUSIONS_???.
hGroup	The group to be excluded from use of the fax numbers.
pahFaxNumber	Pointer to an array of fax numbers the group will be excluded from.
sFaxNumberCount	Number of handles at pahFaxNumber.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 25: MFP functions

The MFP functions open a list of MFP devices and add, edit, and delete MFPs on the RightFax server.

RFaxDeleteDevice()

This function deletes an MFP device.

C Prototype

```
DWORD RFAXAPI RFaxDeleteDevice(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey
 IN OPTIONAL FAXHANDLE hDevice
 IN OPTIONAL RFAXCSTR lpUniqueIdentifier
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0=handle 1=network ID 2=uniqueID 3=printDriver
hDevice	Specify the handle of the device to delete.
lpUniqueIdentifier	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Deletes the MFP device specified by the handle field.

See Also

RFaxLoadDevice() on page 228

RFaxGetDeviceList() on the next page

RFaxSaveDevice() on page 230

RFaxDeleteDeviceTag()

This function deletes an attribute associated with the specified device.

C Prototype

```
DWORD RFAXAPI RFaxDeleteDeviceTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN RFAXCSTR pszName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user whose attribute will be deleted.
pszName	Name of the attribute to delete

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxLoadDeviceTag() on page 229

RFaxSaveDeviceTag() on page 231

RFaxGetDeviceTagList() below

RFaxGetDeviceList()

This function gets a handle to a list of MFPs.

C Prototype

```
DWORD RFAXAPI RFaxGetDeviceList(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL RFAXCSTR lpDeviceType,
 IN OPTIONAL RFAXCSTR lpIntegrationType,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpDeviceType	
lpIntegrationType	
lpLH	On success, returns a handle to a list of DEVICEINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetDeviceTagList()

This function gets attributes associated with the specified device.

C Prototype

```
DWORD RFAXAPI RFaxGetDeviceTagList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of the user whose attribute will be deleted.
*lpLH	If return value is 0, a handle to a list of *TAGINFO_10* on page 374 structures.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxLoadDeviceTag() on page 229

RFaxSaveDeviceTag() on page 231

RFaxDeleteDeviceTag() on page 225

RFaxGetPagedDeviceList()

Gets a paged list of MFPs.

C Prototype

```
DWORD RFAXAPI RFaxGetPagedDeviceList(
 IN RFAXSERVERHANDLE hServer,
 IN BOOL fAscending,
 IN USHORT usOrder,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG plTotalDevices,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fAscending	TRUE for ascending; FALSE for descending.
usOrder	Order in which to load the paged records. See the DEVICE_ORDER_??? constants.
lPageIndex	0-based index of page to return.
usPageSize	Number of device records per page.
plTotalDevices	When not null, receives total device records.
lpLH	Returns a handle to list of DEVICEINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_NOTFOUND	The fax list is empty.

See Also

RFaxGetDeviceList() on the previous page

RFaxGetPagedDeviceList2() below

RFaxGetPagedDeviceList2()

Gets a paged list of MFPs with a choice of information level.

C Prototype

```
DWORD RFAXAPI RFaxGetPagedDeviceList2(
 IN RFAXSERVERHANDLE hServer,
 IN BOOL fAscending,
 IN USHORT usOrder,
 IN LONG lPageIndex,
 IN USHORT usPageSize,
 OUT LONG plTotalDevices,
 IN USHORT usInfoLevel 10,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

fAscending	TRUE for ascending; FALSE for descending.
usOrder	Order in which to load the paged records. See the DEVICE_ORDER_??? constants.
lPageIndex	0-based index of page to return.
usPageSize	Number of device records per page.
plTotalDevices	When not null, receives total device records.
usInfoLevel	Specify the level of information to load, 10 or 11.
lpLH	If return = 0, returns a handle to list of DEVICEINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_NOTFOUND	The fax list is empty.

See Also

RFaxGetDeviceList() on page 226

RFaxGetPagedDeviceList() on the previous page

RFaxLoadDevice()

This function loads MFP device information.

C Prototype

```
DWORD RFAXAPI RFaxLoadDevice(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sWhichKey,
 IN OPTIONAL FAXHANDLE hDevice,
 IN OPTIONAL RFAXCSTR lpSearchValue,
 OUT PDEVICEINFO_10 lpDV10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0=hDevice 1=networkAddress 2=uniqueID 3=PrintDriver
hDevice	Handle of the device to load when sWhichKey is 0.
lpSearchValue	Text matching the device to load when sWhichKey is greater than 0.
lpDV10	Returns a DEVICEINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

You must specify one and only one of the hDevice or lpUniqueIdentifier arguments. The unused argument must be set to zero (0) or Null.

See Also

RFaxLoadDevice2() below

RFaxSaveDevice() on page 230

RFaxSaveDevice2() on page 230

RFaxLoadDevice2()

This function loads MFP device information.

C Prototype

```
DWORD RFAXAPI RFaxLoadDevice(
 IN RFAXSERVERHANDLE hServer,
```

```
    IN SHORT sWhichKey,
    IN FAXHANDLE hDevice,
    IN RFAXCSTR lpSearchValue,
    IN USHORT usInfoLevel,
    OUT PDEVICEINFO_10 lpDV10
   );
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichKey	0=hDevice 1=networkAddress 2=uniqueID 3=PrintDriver
hDevice	Handle of the device to load when sWhichKey is 0.
lpSearchValue	Text matching the device to load when sWhichKey is greater than 0.
usInfoLevel	Level of data structure pointed to by lpDV, 10 or 11.
lpDV10	Returns a DEVICEINFO_10 structure or DEVICEINFO_11 structure as indicated by the usInfoLevel parameter.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

You must specify one and only one of the hDevice or lpUniqueIdentifier arguments. The unused argument must be set to zero (0) or Null.

See Also

RFaxLoadDevice() on the previous page

RFaxSaveDevice() on the next page

RFaxSaveDevice2() on the next page

RFaxLoadDeviceTag()

This function loads an attribute associated with the specified device.

C Prototype

```
DWORD RFAXAPI RFaxLoadDeviceTag(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 IN RFAXCSTR pszName,
 OUT PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	Handle of user to load attribute from.
pszName	Name of the attribute to load.
lpTI10	Details of the loaded attribute.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxSaveDeviceTag() on page 231

RFaxSaveDeviceTag() on page 231

RFaxGetDeviceTagList() on page 226

RFaxSaveDevice()

This function updates or saves a new MFP device object.

C Prototype

```
DWORD RFAXAPI RFaxSaveDevice(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewDevice,
 IN PDEVICEINFO_10 lpDV10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewDevice	Specify True if you are adding a new printer. Specify False if you are updating an existing printer.
lpPI10	Pointer to a DEVICEINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an existing device, verify that fNewDevice is set to False and that the handle member of the DEVICEINFO_10 structure is preserved. Otherwise, the function will fail.

See Also

RFaxSaveDevice2() below

RFaxLoadDevice() on page 228

RFaxLoadDevice2() on page 228

RFaxSaveDevice2()

This function updates or saves a new MFP device object.

C Prototype

```
DWORD RFAXAPI RFaxSaveDevice2(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNewDevice,
 IN USHORT usInfoLevel 10,
 IN RFAXPVOID lpDV
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNewDevice	
usInfoLevel	Level of data structure pointed to by lpDV, 10 or 11.
lpDV	As indicated by usInfoLevel parameter, the DEVICEINFO_10 structure or *DEVICEINFO_11* on page 316 structure to save.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

If you are updating an existing device, verify that fNewDevice is set to False and that the handle member of the DEVICEINFO_10 structure is preserved. Otherwise, the function will fail.

See Also

RFaxSaveDevice() above

RFaxLoadDevice() on page 228

RFaxLoadDevice2() on page 228

RFaxSaveDeviceTag()

This function saves an attribute associated with the specified device.

C Prototype

```
DWORD RFAXAPI RFaxSaveDeviceTag(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT fNew,
 IN PTAGINFO_10 lpTI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
fNew	TRUE if the attribute is new to the user referred to in lpTI10->hOwner.
lpTI10	Details of the attribute to save.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

See Also

RFaxLoadDeviceTag() on page 229

RFaxGetDeviceTagList() on page 226

RFaxDeleteDeviceTag() on page 225

Chapter 26: Folder functions

The folder functions open one folder or a list of folders for a specified user. You can also create, re-name, end empty users' folders.

RFaxCreateFolder()

Deprecated.

This function creates a folder with the specified name for the given user.

C Prototype

```
DWORD RFAXAPI RFaxCreateFolder(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 pUI10,
 IN RFAXCSTR pszFolderName,
 IN BOOL bSaveUser,
 OUT OPTIONAL USHORT *pusNewFolderID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure with the information on the user for whom a folder is to be created. Such a structure can be obtained via a call to an RFaxLoadUser() function.
pszFolderName	Name for the new folder.
bSaveUser	Set to True to save the user record after performing the create. Otherwise, the change will not be reflected until the user record is saved via a call to RFaxSaveUser().
pusNewFolderID	ID of the newly created folder optionally returned. Pass 0 or Null if this value is not needed.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

This function creates the specified folder for the given user. This function can fail if there is not enough room for the folder name or if the folder limit has been reached. All folders created with this function create the folder directly under the root. To create a subfolder, use the RFaxCreateFolder2() function.

See Also

RFaxCreateFolder2() on the next page

RFaxGetFolderList() on page 238

RFaxSaveUser() on page 162

RFAXERR_NOFOLDERSPACE on page 468

RFAXERR_NOMOREFOLDERS on page 468

RFaxCreateFolder2()

Deprecated.

This function creates a folder or subfolder with the specified name for the given user.

C Prototype

```
DWORD RFAXAPI RFaxCreateFolder2(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 pUI10,
 IN USHORT usParentID,
 IN RFAXCSTR pszFolderName,
 IN BOOL bSaveUser,
 OUT OPTIONAL USHORT *pusNewFolderID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pUI10	A USERINFO_10 structure with the information on the user for whom a folder is to be created. Such a structure can be obtained via a call to an RFaxLoadUser() function.
usParentID	The name of the parent folder to use when creating subfolders. Set this argument to zero to create the folder under the root.
pszFolderName	Name for the new folder.
bSaveUser	Set to True to save the user record after performing the create. Otherwise, the change will not be reflected until the user record is saved via a call to RFaxSaveUser().
pusNewFolderID	ID of the newly created folder optionally returned. Pass 0 or Null if this value is not needed.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

This function creates the specified folder for the given user. This function can fail if there is not enough room for the folder name or if the folder limit has been reached.

See Also

RFaxCreateFolder() on the previous page

RFaxGetFolderList() on page 238

RFAXERR_NOFOLDERSPACE on page 468

RFAXERR_NOMOREFOLDERS on page 468

RFaxCreateFolder3()

This function creates a folder with the specified name for the given user.

C Prototype

```
DWORD RFAXAPI RFaxCreateFolder3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usParentID,
 IN RFAXCSTR pszFolderName,
 OUT USHORT *pusNewFolderID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Handle of owning user or NULL for unowned.

usParentID	The name of the parent folder to use when creating subfolders. Set this argument to zero to create the folder under the root.
pszFolderName	Name for the new folder.
pusNewFolderID	Optional. ID of the newly created folder optionally returned. Pass 0 or Null if this value is not needed.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

This function creates the specified folder for the given user. This function can fail if there is not enough room for the folder name or if the folder limit has been reached.

See Also

RFaxGetFolderList2() on page 238

RFaxGetFolderList3() on page 239

RFAXERR_NOFOLDERSPACE on page 468

RFAXERR_NOMOREFOLDERS on page 468

RFaxDeleteFolder()

Depcrecated.

This function deletes the given folder for the specified user.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFolder(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 pUI10,
 IN USHORT usFolderID,
 IN BOOL bCheckEmpty,
 IN BOOL bSaveUser
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure with the information on the user whose folder is to be deleted. Such a structure can be obtained via a call to an RFaxLoadUser() function.
usFolderID	Specify the ID of the folder to delete. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
bCheckEmpty	Set to True to verify that the folder is empty before deleting. If this is True and the folder is not empty, the function will not delete the folder and will return RFAXERR_FOLDERNOTEMPTY.
bSaveUser	Set to True to save the user record after performing the delete. Otherwise, the change will not be reflected until the user record is saved via a call to RFaxSaveUser().

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes the specified folder for the given user. The usFolderID can be retrieved from a FOLDERLISTELEMENT0 structure, which can be obtained by traversing the list returned by RFaxGetFolderList().

See Also

RFaxGetFolderList() on page 238

RFaxSaveUser() on page 162

RFaxDeleteFolder2()

This function deletes the given folder for the specified user.

C Prototype

```
DWORD RFAXAPI RFaxDeleteFolder2(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usFolderID,
 IN FAXBOOL bCheckEmpty
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	Handle of owning user or NULL for unowned.
usFolderID	Specify the ID of the folder to delete. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
bCheckEmpty	Set to True to verify that the folder is empty before deleting. If this is True and the folder is not empty, the function will not delete the folder and will return RFAXERR_FOLDERNOTEMPTY.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes the specified folder for the given user. The usFolderID can be retrieved from a FOLDERLISTELEMENT0 structure, which can be obtained by traversing the list returned by RFaxGetFolderList().

See Also

RFaxGetFolderList() on page 238

RFaxSaveUser() on page 162

RFaxEmptyFolder()

This function empties the specified folder by deleting all the faxes it contains.

C Prototype

```
DWORD RFAXAPI RFaxEmptyFolder(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszUserID,
 IN USHORT usFolderID
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszUserID	The ID of the user whose folder is to be emptied.
usFolderID	Specify the ID of the folder to empty. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes all the faxes contained in the specified folder.

See Also

RFaxGetFolderList() on page 238

RFaxSaveUser() on page 162

RFaxGetBuiltInFolderName()

This function gets the localized name for a permanent or built-in folder.

C Prototype

```
DWORD RFAXAPI RFaxCreateFolder2(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usFolderID,
 IN FAXBOOL fOutputUTF8,
 OUT RFAXSTR pszFoldername,
 IN ULONG ulMaxFoldername
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usFolderID	pecify the ID of the folder to delete. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
fOutputUTF8	
pszFolderName	Name for the new folder.
ulMaxFoldername	Maximum size of string at pszFoldername.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

See Also

RFaxGetFolderList() on page 238

RFAXERR_NOFOLDERSPACE on page 468

RFAXERR_NOMOREFOLDERS on page 468

RFaxGetChildFolders()

This function gets a list of child folders.

C Prototype

```
DWORD RFAXAPI RFaxGetChildFolders(
 IN RFAXLISTHANDLE hListHandle,
 IN USHORT usParentID,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hListHandle	Handle to list of folder list elements. SeeFOLDERLISTELEMENT0
usParentID	Parent ID of the folder whose children will be in the list..
lpLH	On success, returns a handle to list of FOLDERLISTELEMENT0 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success

Return	Result
RFAXERR_???	Failure

RFaxGetFolderCount()

Deprecated.

This function gets the number of user-created folders. To specify a fax server or to include subfolders, use the RFaxGetFolderCount2() function.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderCount(
 IN PUSERINFO_10 lpUI10,
 OUT ULONG *lpCount
);
```

Arguments

lpUI10	A USERINFO_10 structure for the information on the user for whom folder information is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
lpCount	On success, a count of the user folders for this user is returned. Note that this count is only of the user-created folders. It does not include the default folders such as Trash and Main.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves the number of user-created folders that belong to the specified user. The count does not include default folders such as Trash and Main.

See Also

RFaxGetFolderCount2() below

RFaxGetFolderCount2()

Deprecated.

This function gets the number of user folders.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderCount2(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 lpUI10,
 OUT ULONG *lpCount,
 OUT OPTIONAL ULONG *lpSubCount
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure for the information on the user for whom folder information is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
lpCount	The resultant folder count.
lpSubCount	The resultant subfolder count. Optional.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to the RFaxGetFolderCount() function except you can specify a fax server and include the user's subfolders.

See Also

RFaxGetFolderCount() on the previous page

RFaxGetFolderList()

Deprecated.

This function gets a list of the specified user's folders.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderList(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 lpUI10,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure for the information on the user for whom folder information is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
lpLH	On success, returns a handle to a list of the folders for the specified user in FOLDERLISTELEMENT0 structures. When done with this list, close the handle with RFaxCloseListHandle().

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns a handle to a list of folders for the specified user in FOLDERLISTELEMENT0 structures.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFolderList2() below

RFaxCloseListHandle() on page 258

RFaxGetFolderList2()

This function gets a list of the specified user's folders and lets you specify subfolders.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderList2(
 IN RFAXSERVERHANDLE hServer,
 IN short sUserInfoLevel,
 IN RFAXPVOID lpUI,
 IN ushort usParentID,
 IN short sFolderElemLevel,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sUserInfoLevel	Specify the information level. This argument dictates which structure was passed in as lpUI. Must be 10 or 11.
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument) to receive the information on the specified user.

usParentID	Specify the parent ID of the folder whose children will be in the list. Use 0 (zero) to specify the root folder.
sFolderElemLevel	Currently must be set to 0 (zero).
lpLH	On success, returns a handle to a list of the folders for the specified user in FOLDERLISTELEMENT0 structures. When done with this list, close the handle with RFaxCloseListHandle().

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns a handle to a list of folders for the specified user in FOLDERLISTELEMENT0 structures.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFolderList() on the previous page

RFaxCloseListHandle() on page 258

RFaxGetFolderList3()

This function gets a list of the specified user's folders.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderList3(
 IN RFAXSERVERHANDLE hServer,
 IN ushort usGetOpt,
 IN FAXHANDLE hOwner,
 IN ushort usParentID,
 IN RFAXUSN usn,
 IN Ushort usInfoLevel,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usGetOpt	Combination of GETFOLDERS_???.
hOwner	User whose folders should be retrieved.
usParentID	Specify the parent ID of the folder whose children will be in the list. Use 0 (zero) to specify the root folder.
usn	Specify non-zero to limit the list to only folders modified after the corresponding USN.
usInfoLevel	Currently must be 0 or 1 for FOLDERLISTELEMENT0,1.
lpLH	If return == 0, returns a handle to a list of FOLDERLISTELEMENT# structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns a handle to a list of folders for the specified user in FOLDERLISTELEMENT0 structures.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetFolderList2() on page 238

RFaxCloseListHandle() on page 258

RFaxGetFolderName()

Deprecated.

This function gets the name of the specified folder for the given user.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderName(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 lpUI10,
 IN USHORT usFolderID,
 OUT RFAXSTR pszFoldername,
 IN ULONG ulMaxFoldername
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure with the information on the user whose folder info is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
usFolderID	Specify the ID of the folder to get. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
pszFoldername	Name of the folder is returned here.
ulMaxFoldername	Maximum size of string at pszFoldername.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns the name of the specified folder.

See Also

RFaxGetFolderList() on page 238

RFaxGetFolderName2()

Deprecated.

This function gets the name of the specified folder for the given user.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderName2(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_11 lpUI11,
 IN USHORT usFolderID,
 OUT RFAXSTR pszFoldername,
 IN ULONG ulMaxFoldername
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_11 structure with the information on the user whose folder info is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
usFolderID	Specify the ID of the folder to get. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
pszFoldername	Name of the folder is returned here.
ulMaxFoldername	Maximum size of string at pszFoldername.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns the name of the specified folder.

See Also

RFaxGetFolderName() on the previous page

RFaxGetFolderName3() below

RFaxGetFolderName3()

Deprecated.

This function gets the name of the specified folder for the given user.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderName3(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_11 lpUI11,
 IN USHORT usFolderID,
 IN BOOL fOutputUTF8,
 OUT RFAXSTR pszFoldername,
 IN ULONG ulMaxFoldername
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI11	A USERINFO_11 structure with the information on the user whose folder info is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
usFolderID	Specify the ID of the folder to get. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
fOutputUTF8	
pszFoldername	Name of the folder is returned here.
ulMaxFoldername	Maximum size of string at pszFoldername.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns the name of the specified folder.

See Also

RFaxGetFolderName() on the previous page

RFaxGetFolderName2() on the previous page

RFaxGetFolderSpace()

Deprecated.

This function gets the number of user folders.

C Prototype

```
DWORD RFAXAPI RFaxGetFolderSpace(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 lpUI10,
 IN OUT ULONG *pulAvailableSpace
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure with the information on the user whose folder info is requested. Such a structure can be obtained via a call to an RFaxLoadUser() function.
pulAvailableSpace	On success, returns the amount of space in bytes that is available for folder names and IDs.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves the amount of space in bytes that is available for creating folders.

RFaxIsBuiltInFolderID()

Indicates whether a folder is a native RightFax folder.

C Prototype

```
FAXBOOL RFAXAPI RFaxIsBuiltInFolderID(
 USHORT usFolderID
);
```

Arguments

usFolderID	If true, the folder is a folder that was built by the system.

RFaxRenameFolder()

Deprecated.

This function re-names the specified folder for the given user.

C Prototype

```
DWORD RFAXAPI RFaxRenameFolder(
 IN RFAXSERVERHANDLE hServer,
 IN PUSERINFO_10 pUI10,
 IN USHORT usFolderID,
 IN RFAXCSTR pszNewFolderName,
 IN BOOL bSaveUser
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpUI10	A USERINFO_10 structure with the information on the user whose folder is to be re-named. Such a structure can be obtained via a call to an RFaxLoadUser() function.
usFolderID	Specify the ID of the folder to re-name. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().

pszNewFolderName	New name for the folder.
bSaveUser	Set to True to save the user record after performing the create. Otherwise, the change will not be reflected until the user record is saved via a call to RFaxSaveUser().

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function re-names the specified folder for the given user. This function will fail if an attempt is made to re-name one of the static folders or there is not enough room for the new folder name.

See Also

RFaxSaveUser() on page 162

RFAXERR_NOFOLDERSPACE on page 468

RFaxRenameFolder2()

This function re-names the specified folder for the given user.

C Prototype

```
DWORD RFAXAPI RFaxRenameFolder(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwner,
 IN USHORT usFolderID,
 IN RFAXCSTR pszNewFolderName,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwner	AHandle of owning user or NULL for unowned.
usFolderID	Specify the ID of the folder to re-name. This is the ID stored in the FOLDERLISTELEMENT0 structure. Such a structure can be obtained by traversing the list returned by RFaxGetFolderList().
pszNewFolderName	New name for the folder.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function re-names the specified folder for the given user. This function will fail if an attempt is made to re-name one of the static folders or there is not enough room for the new folder name.

See Also

RFaxSaveUser() on page 162

RFAXERR_NOFOLDERSPACE on page 468

Chapter 27: Phonebook functions

The phonebook functions open a RightFax phonebook or individual phonebook entry for a specific user and add, edit, and delete phonebooks and phonebook entries.

RFaxCopyPhone()

This function copies a phonebook item (group or entry) to another user.

C Prototype

```
DWORD RFAXAPI RFaxCopyPhone(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN RFAXCSTR lpUser
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle2() or RFaxCreateServerHandle3().
handle	Handle a phonebook item to copy. The item can be an entry or a phonebook group.
lpUser	User to copy the item to.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

Use this function to copy phonebook items to another user.

RFaxDeletePhone()

This function deletes a phonebook entry or group.

C Prototype

```
DWORD RFAXAPI RFaxDeletePhone(
 IN FAXSERVERHANDLE hServer,
 IN FAXHANDLE handle
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of a phonebook item to delete. The item can be an entry or a phonebook group.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Deleting a phonebook group does not delete the individual entries named in or used by the group.

See Also

RFaxGetPhoneList() on page 247

RFaxDeletePhonebook()

This function deletes a phonebook.

C Prototype

```
DWORD RFAXAPI RFaxDeletePhonebook(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Specify the handle of the phonebook to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetPhonebookEntriesList()

This function obtains a list of phonebook entries.

C Prototype

```
DWORD RFAXAPI RFaxGetPhonebookEntriesList(
 IN RFAXSERVERHANDLE hServer,
 IN PPHONEBOOKINFO_10 lpPBI10,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpPBI10	A PHONEBOOKINFO_10 structure for the information on the phonebook.
lpLH	Returns a handle to a list of PHONEITEM structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves a list of phonebook entriesthat can then be displayed to the user or exported to a database file.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxGetPhonebooksList()

Gets a list of phonebooks.

C Prototype

```
DWORD RFAXAPI RFaxGetPhonebooksList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN FAXHANDLE hGroupHandle,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

usInfoLevel	Specify the level of information to load. This must be set to 10.
hGroupHandle	The handle of the group by which to filter the group's access control.
lpLH	Returns a handle to a list of PHONEBOOKINFO_10 structures

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_NOTFOUND	The fax list is empty.

RFaxGetPhoneGroupList()

This function loads members of the specified PHONEITEM group.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneGroupList(
 IN RFAXSERVERHANDLE hServer,
 IN PHONEITEM *pPhone,
 IN OUT RFAXLISTHANDLE *phMemberList,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
*pPhone	Phone (group) item from which to get the members
*phMemberList	On success, this argument returns a handle to list of phone IDs of type char[18]

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function loads members of the specified PHONEITEM group, including those stored in the PHONEITEM and those stored in the extension list associated with the phone item.

See Also

RFaxGetPhoneGroupList2() below

RFaxGetPhoneGroupList2()

This function loads members of the specified PHONEITEM group.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneGroupList2(
 IN RFAXSERVERHANDLE hServer,
 IN PHONEITEM *pPhone,
 IN short sUserInfoLevel,
 IN RFAXPVOID lpUI,
 IN RFAXCSTR lpPassword,
 OUT RFAXLISTHANDLE *phMemberList
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
*pPhone	Phone (group) item from which to get the members
sUserInfoLevel	Level of data structure pointed to by the lpUI argument (10 or 11)
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument).
lpPassword	Password for the user specified in lpUI argument

*phMemberList	On success, returns a handle to list of PHONEITEM.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function loads members of the specified PHONEITEM group, including those stored in the PHONEITEM and those stored in the extension list associated with the phone item.

See Also

RFaxGetPhoneGroupList() on the previous page

RFaxGetPhoneGroupList3()

This function loads members of the specified PHONEITEM group. This version takes a handle to the originating phone item group rather than the PHONEITEM structure.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneGroupList3(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hPhone,
 IN short sUserInfoLevel,
 IN RFAXPVOID lpUI,
 IN RFAXCSTR lpPassword,
 OUT RFAXLISTHANDLE *phMemberList
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
*pPhone	Phone (group) item from which to get the members
sUserInfoLevel	Level of data structure pointed to by the lpUI argument (10 or 11)
lpUI	A USERINFO_10 or USERINFO_11 structure (dependent upon the sUserInfoLevel argument).
lpPassword	Password for the user specified in lpUI argument
*phMemberList	On success, returns a handle to list of PHONEITEM.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function loads members of the specified PHONEITEM group, including those stored in the PHONEITEM and those stored in the extension list associated with the phone item.

See Also

RFaxGetPhoneGroupList() on the previous page

RFaxGetPhoneList()

This function loads a list of phonebook entries of a specified user.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneList(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN RFAXCSTR lpUserID,
 IN RFAXCSTR lpPassword,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, or 1 depending on whether you want the PHONELISTELEMENT0, or the PHONELISTELEMENT1 structure returned.
lpUserID	Specify the user ID of the phonebook to load.
lpPassword	Specify the password of the user specified in lpUserID. If this is Null or an empty string, then only the public entries of lpUserID will be loaded.
lpLH	Depending on the value of sInfoLevel, returns a handle to a list of the PHONELISTELEMENT0 or the PHONELISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Null cannot be passed for the lpLH argument, however, the handle that the lpLH argument points to must be initialized to Null. hList must be assigned zero before being passed to this function.

This function will load the phonebook for the user and all phonebooks linked by the user. If you don't want to load all of the linked phonebooks, use the RFaxGetPhoneList2() function.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetPhoneList2() below

RFaxGetPhoneList3() on page 250

RFaxGetPhoneList2()

Allows you to load a list of phonebook entries of a specified user with different information levels.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneList2(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN BOOL bLoadOthers,
 IN RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpPassword,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, or 1 depending on whether you want the PHONELISTELEMENT0, or the PHONELISTELEMENT1 structure returned.
bLoadOthers	Specify True to load the phonebook for the user and all phonebooks linked by the user. Specify False to ensure that only the user's directly owned phonebook items are loaded.
lpUserID	Specify the user ID of the phonebook to load.
lpPassword	Specify the password of the user specified in lpUserID. If this is Null or an empty string, then only the public entries of lpUserID will be loaded.
lpLH	Depending on the value of sInfoLevel, returns a handle to a list of the PHONELISTELEMENT0 or the PHONELISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The lpLH argument must be an initialized to Null.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetPhoneList() on page 247

RFaxGetPhoneList3() on the next page

RFaxGetPhoneList4()

Allows you to load a list of phonebook entries of a specified user with different information levels.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneList4(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN OPTIONAL RFAXCSTR lpSearchString,
 IN USHORT usGetOptions
 IN USHORT usSearchOptions
 IN RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpPassword,
 IN OPTIONAL LONG lMaxRead,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, or 1 depending on whether you want the PHONELISTELEMENT0, or the PHONELISTELEMENT1 structure returned.
lpSearchString	Search and return only entries whose name contains the specified search string.
usGetOptions	See *GETPHONES_???* on page 435.
usSearchOptions	See *SEARCHPHONEOPTS_???* on page 477.
lpUserID	Specify the user ID of the phonebook to load.
lpPassword	Specify the password of the user specified in lpUserID. If this is Null or an empty string, then only the public entries of lpUserID will be loaded.
lMaxRead	Maximum number of items to read (0 = unlimited).
lpLH	Depending on the value of sInfoLevel, returns a handle to a list of the PHONELISTELEMENT0 or the PHONELISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The lpLH argument must be an initialized to Null.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetPhoneList() on page 247

RFaxGetPhoneList2() on page 248

RFaxGetPhoneList3()

Allows you to load a list of phonebook entries of a specified user with different information levels.

C Prototype

```
DWORD RFAXAPI RFaxGetPhoneList3(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN BOOL bLoadOthers,
 IN SHORT sGetOptions,
 IN RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpPassword,
 IN OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Specify the information level. Must be 0, or 1 depending on whether you want the PHONELISTELEMENT0, or the PHONELISTELEMENT1 structure returned.
bLoadOthers	Specify True to load the phonebook for the user and all phonebooks linked by the user. Specify False to ensure that only the user's directly owned phonebook items are loaded.
sGetOptions	0=everything 1=PBEntryNotList
lpUserID	Specify the user ID of the phonebook to load.
lpPassword	Specify the password of the user specified in lpUserID. If this is Null or an empty string, then only the public entries of lpUserID will be loaded.
lpLH	Depending on the value of sInfoLevel, returns a handle to a list of the PHONELISTELEMENT0 or the PHONELISTELEMENT1 structure.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The lpLH argument must be an initialized to Null.

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

See Also

RFaxGetPhoneList() on page 247

RFaxGetPhoneList2() on page 248

RFaxLoadPhonebook()

This function loads a global phonebook by its handle or name.

C Prototype

```
DWORD RFAXAPI RFaxLoadPhonebook(
 IN RFAXSERVERHANDLE hServer,
 IN short sWhichKey,
 IN OPTIONAL FAXHANDLE hPhonebook,
 IN OPTIONAL RFAXCSTR lpPhonebookName
 OUT PPHONEBOOKINFO_10 lpPBI10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().

sWhichKey	0=handle 1=lpPhonebookName
hPhonebook	Handle of the phonebook.
lpPhonebookName	Name of the phonebook.
lpPBI10	Returns a PHONEBOOKINFO_10 structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

RFaxLoadPhoneByHandle()

This function loads a phonebook entry by its handle.

C Prototype

```
DWORD RFAXAPI RFaxLoadPhoneByHandle(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 OUT PPHONEITEM lpPI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Specify the handle of the phonebook item.
lpPI	Returns a PHONEITEM structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

To use this function, the you must know the handle of the phonebook entry to load. To load a phonebook entry with an unknown handle, use RFaxLoadPhoneByName().

See Also

RFaxLoadPhoneByName() below

RFaxLoadPhoneByName()

This function loads a phonebook entry by the ID of the entry.

C Prototype

```
DWORD RFAXAPI RFaxLoadPhoneByName(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpEntry,
 IN RFAXCSTR lpUser,
 IN RFAXCSTR lpPassword,
 OUT PPHONEITEM lpPI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpEntry	Specify the name of the phonebook entry.
lpUserID	Specify the user ID of the owner of the phonebook entry.
lpPassword	Specify the password of lpUser. If the password is empty, then only the public entries of the users specified in lpUser will be loaded.
lpPI	Returns a PHONEITEM structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

To load a phonebook entry without the user's password, use RFaxLoadPhoneByHandle().

See Also

RFaxLoadPhoneByHandle() on the previous page

RFaxLoadPhoneByName2() below

RFaxLoadPhoneByName2()

This function loads a phonebook entry by the ID of the entry.

C Prototype

```
DWORD RFAXAPI RFaxLoadPhoneByName2(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpEntry,
 IN RFAXCSTR lpUser,
 IN RFAXCSTR lpPassword,
 IN USHORT usOptions,
 OUT PPHONEITEM lpPI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpEntry	Specify the name of the phonebook entry.
lpUserID	Specify the user ID of the owner of the phonebook entry.
lpPassword	Specify the password of lpUser. If the password is empty, then only the public entries of the users specified in lpUser will be loaded.
usOptions	See LOADPHONEOPT_???.
lpPI	Returns a PHONEITEM structure.

Return Values

Return	Result
RFAX_SUCCESS	Success
RFAXERR_???	Failure

Remarks

To load a phonebook entry without the user's password, use RFaxLoadPhoneByHandle().

See Also

RFaxLoadPhoneByHandle() on the previous page

RFaxLoadPhoneByName() on the previous page

RFaxSavePhone()

This function saves a new phonebook item (group or entry).

C Prototype

```
DWORD RFAXAPI RFaxSavePhone(
 IN RFAXSERVERHANDLE hServer,
 IN PPHONEITEM lpPI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpPI	Specify the PHONEITEM structure that you want to save.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to add new entries only. To update an existing entry, use RFaxUpdatePhone().

See Also

RFaxUpdatePhone() below

RFaxSavePhonebook()

This function saves a new phonebook.

C Prototype

```
DWORD RFAXAPI RFaxSavePhonebook(
 IN RFAXSERVERHANDLE hServer,
 IN PPHONEBOOKINFO_10 lpPBI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpPBI	Pointer to the PHONEBOOKINFO_10 structure with the information to save.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSavePhoneGroupList()

C Prototype

```
DWORD RFAXAPI RFaxSavePhoneGroupList(
 IN RFAXSERVERHANDLE hServer,
 IN PHONEITEM *pPhone,
 IN RFAXLISTHANDLE hlGroupMemberList,
 IN RFAXCSTR lpPassword
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
*pPhone	Specify the Phone (group) item that you want to save.
hlGroupMemberList	Handle to a list of phone IDs (char[18]).
lpPassword	The owner's (pPhone -> user_id's) password.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxGetPhoneGroupList() on page 246

RFaxUpdatePhone() below

RFaxUpdatePhone()

This function updates an existing phonebook item (group or entry).

C Prototype

```
DWORD RFAXAPI RFaxUpdatePhone(
 IN RFAXSERVERHANDLE hServer,
 IN PPHONEITEM lpPI
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpPI	Specify the PHONEITEM structure to update.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to update exiting phonebook entries. It will fail is the entry doesn't already exist. Use RFaxSavePhoneGroupList() to add new items.

Chapter 28: Extension functions

The extension functions let you work with extension records. Extension records are used when you need more space for a record than is available. The current version of the RightFax API eliminates the need for these functions in most cases; however, they are documented here for purposes of backward compatibility.

RFaxDeleteExtension()

This function deletes an extension.

C Prototype

```
DWORD RFAXAPI RFaxDeleteExtension(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hExtension
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hExtension	Handle of the extension to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes the specified extension.

RFaxFindExtension()

This function finds an extension of the specified type for the owner object.

C Prototype

```
DWORD RFAXAPI RFaxFindExtension(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwnerObject,
 IN unsigned long ulExtensionType,
 OUT PEXTENSIONINFO_10 lpEXT10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwnerObject	Handle of the owning object.
ulExtensionType	One of the EXTENSIONTYPE_??? constants.
lpEXT10	On success, returns information on the extension found.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function searches for the specified extension. On success, information about the extension is returned. The specific union data that is valid is dependent upon the ulExtensionType. See the EXTENSIONTYPE_??? constants for details.

RFaxGetExtensionList()

This function gets a handle to a list of extensions for the specified object.

C Prototype

```
DWORD RFAXAPI RFaxGetExtensionList(
 IN RFAXSERVERHANDLE hServer,
 IN short sInfoLevel,
 IN FAXHANDLE hOwnerObject,
 OUT RFAXLISTHANDLE *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sInfoLevel	Information level. Must be 10, which specifies a list of EXTENSIONINFO_10 structures.
hOwnerObject	Handle of the owning object.
lpLH	On success, returns a handle to a list of EXTENSIONINFO_10 objects.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The list handle returned by this function should be closed using RFaxCloseListHandle() in order to free the resources used by the list.

RFaxLoadExtension()

This function loads an extension.

C Prototype

```
DWORD RFAXAPI RFaxLoadExtension(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hExtension,
 OUT PEXTENSIONINFO_10 lpEXT10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hExtension	Handle of the extension to load.
lpEXT10	On success, returns information about the extension loaded.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function loads the specified extension. On success, information about the extension is returned.

See Also

RFaxSaveExtension() below

RFaxSaveExtension()

This function saves an extension with specified information about the owner object.

This function saves the information for the specified extension. If modifying an existing extension, the “handle” field of the EXTENSIONINFO_10 structure must be set to a valid extension handle.

C Prototype

```
DWORD RFAXAPI RFaxSaveExtension(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hOwnerObject,
 IN SHORT fNew,
 IN PEXTENSIONINFO_10 lpEXT10
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hOwnerObject	Handle of the owning object.
fNew	Set to True to create a new extension. Set to False to modify the existing extension. If False, the “handle” field of the EXTENSIONINFO_10 structure must be set to a valid extension handle.
lpEXT10	An EXTENSIONINFO_10 object containing information on the extension to be saved. This information can be obtained via a call to RFaxLoadExtension(), RFaxFindExtension(), or from the list returned by RFaxGetExtensionList().

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Chapter 29: List functions

The list functions are generic functions that manipulate lists of RightFax elements created using functions such as RFaxGetBillCodeList() and RFaxGetUserList().

RFaxCloseListHandle()

This function closes a list handle created by RFaxCreateList() or returned from one of the other API calls.

C Prototype

```
void RFAXAPI RFaxCloseListHandle(
 IN RFAXLISTHANDLE hList
);
```

Arguments

hList	Handle of the list to close.

Return Values

None.

Remarks

It is important to close the list handle to free the resources used by the list.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() below

RFaxCreateList()

This function creates a list that can contain zero to 2 billion like-sized object/elements, each up to 64KB in size.

C Prototype

```
RFAXLISTHANDLE RFAXAPI RFaxCreateList(
 IN unsigned short usElementSize,
 IN unsigned long ulStartingElementCount
);
```

Arguments

usElementSize	Specify the size of an element in the list. All elements must be the same size, although they do not need to be homogeneous.
ulStartingElementCount	Specify the maximum number of elements to store in the list. The size of the list in elements can be adjusted using the RFaxListSetElementCount() function.

Return Values

Return	Result
The handle of the list	Success
Null	Failure

Remarks

List handles acquired with this function, or with any other API function that returns a list handle, must be closed by the caller. Failure to do so will cause memory and resource leaks.

All elements in a list are initialized to, or filled with, zero.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList2() below
RFaxCloseListHandle() on the previous page
RFaxListSetElementCount() on page 268

RFaxCreateList2()

This function creates a list and sets an initial capacity for it. You can create an easily expandable list by setting the capacity to a value much greater than what is needed for the current amount of data.

C Prototype

```
RFAXLISTHANDLE RFAXAPI RFaxCreateList2(
 IN unsigned short usElementSize,
 IN unsigned long ulStartingElementCount
 IN unsigned long ulStartingElementCapacity
);
```

Arguments

usElementSize	Specify the size of an element in the list. All elements must be the same size, although they do not need to be homogeneous.
ulStartingElementCount	Specify the maximum number of elements to store in the list. The number of elements in lists can be adjusted using the RFaxListSetElementCount() function.
ulStartingElementCapacity	Specify the maximum number of elements that could be stored in this list. This is the maximum capacity of the list.

Return Values

Return	Result
The handle of the list	Success
Null	Failure

Remarks

List handles acquired with this function, or with any other API function that returns a list handle, must be closed by the caller. Failure to do so will cause memory and resource leaks.

All elements in a list are initialized to, or filled with, zero.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on the previous page

RFaxCloseListHandle() on the previous page

RFaxListSetElementCount() on page 268

RFaxListAppendElement()

This function adds an element to a list.

C Prototype

```
DWORD RFAXAPI RFaxListAppendElement(
 IN RFAXLISTHANDLE hList,
 IN RFAXPVOID lpBuffer
);
```

Arguments

hList	Handle of the list to add to.

lpBuffer	Pointer to a buffer containing the element data to add to the list.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElementCount() on page 262

RFaxListSetElementCount() on page 268

RFaxListSetElement() on page 268

RFaxListConcatLists()

This function adds the contents of the second list to the end of the first list.

C Prototype

```
DWORD RFAXAPI RFaxListConcatLists(
 IN RFAXLISTHANDLE hList1,
 IN RFAXLISTHANDLE hList2
);
```

Arguments

hList1	Handle of the list to receive the data in hList2.
hList2	Handle of the list containing the data to add to hList1.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function adds the data in the list identified by hList2 to the end of the data in the list identified by hList1. If the element sizes in the two lists do not match, the function will fail with RFAXERR_UNEQUAL_LISTS.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxListConcatLists2() below

RFaxCreateList() on page 258

RFaxListConcatLists2()

This function adds the contents of the second list to the end of the first list.

C Prototype

```
DWORD RFAXAPI RFaxListConcatLists2(
 IN RFAXLISTHANDLE hList1,
 IN RFAXLISTHANDLE hList2
 IN BOOL bAppend
);
```

Arguments

hList1	Handle of the list to receive the data in hList2.
hList2	Handle of the list containing the data to add to hList1.
bAppend	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function adds the data in the list identified by hList2 to the end of the data in the list identified by hList1. If the element sizes in the two lists do not match, the function will fail with RFAXERR_UNEQUAL_LISTS.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxListConcatLists() on the previous page

RFaxCreateList() on page 258

RFaxListCopyList()

This function creates a list by copying data elements from a data source.

C Prototype

```
DWORD RFAXAPI RFaxListCopyList(
 IN RFAXLISTHANDLE hListSource,
 OUT RFAXLISTHANDLE phListCopy
);
```

Arguments

hListSource	Handle of the list to copy from.
phListCopy	Handle of the newly created list.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListDeleteElement()

This function deletes an element from a list.

C Prototype

```
DWORD RFAXAPI RFaxListDeleteElement(
 IN RFAXLISTHANDLE hList,
 IN unsigned long ulIndex
);
```

Arguments

hList	Handle of the list to add to.
ulIndex	Specify the index of the list item to retrieve. The index is zero-based.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElementCount() on the next page

RFaxListSetElementCount() on page 268

RFaxListSetElement() on page 268

RFaxListDeleteElementbyPtr()

This function deletes an element from a list by its pointer.

C Prototype

```
DWORD RFAXAPI RFaxListDeleteElementByPtr(
 IN RFAXLISTHANDLE hList,
 IN RFAXPVOID pvElement
);
```

Arguments

hList	Handle of the list to add to.
pvElement	Pointer of the element to delete.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with all RightFax versions.

See Also

RFaxListGetElementPtr() on the next page

RFaxListGetElement()

This function retrieves an element from a list.

C Prototype

```
DWORD RFAXAPI RFaxListGetElement Lib
 IN RFAXLISTHANDLE hList,
 IN unsigned long ulIndex,
 IN RFAXPVOID lpBuffer
);
```

Arguments

hList	Handle of the list.
ulIndex	Index of the item to get from the list.
lpBuffer	Pointer to buffer large enough to hold a single element of the list. The caller is responsible for allocation and deallocation of the buffer space.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Attempting to retrieve an element from an empty list will return an error of RFAXERR_LIST_EMPTY. Attempting to retrieve an element beyond the end of a list will return an error of RFAXERR_LIST_END.

Remarks

This function returns a copy of the element stored in the list. If the element size is significant, loops examining all elements of a list may incur a performance penalty due to the buffer copies. The function RFaxListGetElementPtr() can be used to retrieve a pointer to the element.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListSetElement() on page 268

RFaxListGetElementCount()

This function retrieves the maximum number of elements/items that can be stored in a list.

C Prototype

```
DWORD RFAXAPI RFaxListGetElementCount(
 IN RFAXLISTHANDLE hList,
 OUT unsigned long *lpulElementCount
);
```

Arguments

hList	Handle of the list.
lpulElementCount	Pointer to unsigned long into which the function will return the maximum element count (the maximum size of the list).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns the maximum number of elements that can be stored in a list. The element count is not the number of valid elements. For example, a list can be limited to a maximum of 100 elements, and the first 25 are valid. The list does not track the number of valid elements – that is left to the caller.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElementSize() below

RFaxListSetElementCount() on page 268

RFaxListGetElementPtr()

This function gets the pointer of an element in a list.

C Prototype

```
DWORD RFAXAPI RFaxListGetElementPtr(
 IN RFAXLISTHANDLE hList,
 IN unsigned long ulIndex,
 OUT RFAXPVOID *lpPtr
);
```

Arguments

hList	Handle of the list.
ulIndex	Specify the index of the list to set (i.e., 0–*n*).
lpPtr	Return a pointer to the element.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is similar to RFaxListGetElement() except that a pointer to the list element is returned. This avoids a buffer copy, which can be significant for larger element sizes. Because a pointer to an element is returned, care must be taken in the use of the pointer, i.e., it can become stale if the list is destroyed or the list size is adjusted.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElementSize()

This function retrieves the size, in bytes, of an individual element.

C Prototype

```
DWORD RFAXAPI RFaxListGetElementSize(
 IN RFAXLISTHANDLE hList,
```

```
 OUT unsigned short *lpusElementSize
);
```

Arguments

hList	Handle of the list.
lpusElementSize	Pointer to unsigned short into which the function will return the element size.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElementCount() on page 262

RFaxListGetFirstElement()

This function retrieves the first element in the list.

C Prototype

```
DWORD RFAXAPI RFaxListGetFirstElement(
 IN RFAXLISTHANDLE hList,
 IN RFAXPVOID lpBuffer
);
```

Arguments

hList	Handle of the list.
lpBuffer	Pointer to a buffer large enough to hold a single element of the list. The caller is responsible for allocation and deallocation of the buffer space.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves the first element of a list and re-sets the current element index to zero.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetLastElement() on the next page

RFaxListGetNextElement() on page 266

RFaxListGetPreviousElement() on page 267

RFaxListGetFirstElementPtr()

This function retrieves the pointer to the first element in the list.

C Prototype

```
DWORD RFAXAPI RFaxListGetFirstElementPtr(
 IN RFAXLISTHANDLE hList,
 OUT RFAXPVOID *lpPtr
);
```

Arguments

hList	Handle of the list.
lpPtr	Returns a pointer to the element.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success

Return	Result
RFAXERR_???	Failure

Remarks

This function retrieves the pointer to the first element in the list and sets the current element index to the first element.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetFirstElement() on the previous page

RFaxListGetNextElementPtr() on the next page

RFaxListGetLastElementPtr() below

RFaxListGetLastElement()

This function retrieves the last element in the list.

C Prototype

```
DWORD RFAXAPI RFaxListGetLastElement(
 IN RFAXLISTHANDLE hList,
 IN RFAXPVOID lpBuffer
);
```

Arguments

hList	Handle of the list.
lpBuffer	Pointer to a buffer large enough to hold a single element of the list. The caller is responsible for allocation and deallocation of the buffer space.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Retrieves the last element of a list and re-sets the current element index to the last element in the list.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetFirstElement() on the previous page

RFaxListGetNextElement() on the next page

RFaxListGetPreviousElement() on page 267

RFaxListGetLastElementPtr()

This function retrieves the pointer to the last element in the list.

C Prototype

```
DWORD RFAXAPI RFaxListGetLastElementPtr(
 IN RFAXLISTHANDLE hList,
 OUT RFAXPVOID *lpPtr
);
```

Arguments

hList	Handle of the list.
lpPtr	Returns a pointer to the element.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves the pointer to the last element in the list and sets the current element index to the last element.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetLastElement() on the previous page

RFaxListGetFirstElementPtr() on page 264

RFaxListGetNextElementPtr() below

RFaxListGetNextElement()

This function retrieves the next element in the list.

C Prototype

```
DWORD RFAXAPI RFaxListGetNextElement(
 IN RFAXLISTHANDLE hList,
 IN RFAXPVOID lpBuffer
);
```

Arguments

hList	Handle of the list.
lpBuffer	Pointer to a buffer large enough to hold a single element of the list. The caller is responsible for allocation and deallocation of the buffer space.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Failure.
RFAXERR_LIST_END	Attempted to retrieve an element past the end of a list.

Remarks

This function retrieves the next element, with respect to the current element index, and increments the current element index.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetFirstElement() on page 264

RFaxListGetLastElement() on the previous page

RFaxListGetPreviousElement() on the next page

RFaxListGetNextElementPtr()

This function retrieves the pointer to the next element in the list.

C Prototype

```
DWORD RFAXAPI RFaxListGetNextElementPtr(
IN RFAXLISTHANDLE hList,
OUT RFAXPVOID *lpPtr
);
```

Arguments

hList	Handle of the list.
lpPtr	Returns a pointer to the element.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves the pointer to the next element, with respect to the current element index, and increments the current element index.

Compatibility
Compatible with all RightFax versions.

See Also
RFaxCreateList() on page 258

RFaxListGetNextElement() on the previous page

RFaxListGetFirstElementPtr() on page 264

RFaxListGetLastElementPtr() on page 265

RFaxListGetPreviousElement()

This function retrieves the previous element in the list.

C Prototype
```
DWORD RFAXAPI RFaxListGetPreviousElement(
 IN RFAXLISTHANDLE hList,
 IN RFAXPVOID lpBuffer
);
```

Arguments
hList	Handle of the list.
lpBuffer	Pointer to a buffer large enough to hold a single element of the list. The caller is responsible for allocation and deallocation of the buffer space.

Return Values
Return	Result
RFAX_SUCCESS (0)	Success.
RFAXERR_???	Attempted to retrieve an element before the first element.

Remarks
This function retrieves the previous element, with respect to the current element index, and decrements the current element index.

Compatibility
Compatible with all RightFax versions.

See Also
RFaxCreateList() on page 258

RFaxListGetFirstElement() on page 264

RFaxListGetNextElement() on the previous page

RFaxListGetLastElement() on page 265

RFaxListGetPreviousElementPtr()

This function retrieves the pointer to the previous element in the list.

C Prototype
```
DWORD RFAXAPI RFaxListGetPreviousElementPtr(
 IN RFAXLISTHANDLE hList,
 OUT RFAXPVOID *lpPtr
);
```

Arguments
hList	Handle of the list.
lpPtr	Returns a pointer to the element.

Return Values
Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks
Retrieves the pointer to the previous element, with respect to the current element index, and decrements the current element index.

Compatibility
Compatible with all RightFax versions.

See Also
RFaxCreateList() on page 258

RFaxListGetPreviousElement() on the previous page

RFaxListGetFirstElementPtr() on page 264

RFaxListGetNextElementPtr() on page 266

RFaxListGetLastElementPtr() on page 265

RFaxListSetElement()

This function updates or initializes an element in a list.

C Prototype

```
DWORD RFAXAPI RFaxListSetElement(
 IN RFAXLISTHANDLE hList,
 IN unsigned long ulIndex,
 IN RFAXPVOID lpBuffer
);
```

Arguments

hList	Handle of the list.
ulIndex	Specify the zero-based index of the list item to set, or update. The list cannot be extended by setting an element beyond the end of the list.
lpBuffer	Pointer to buffer large enough to hold a single object of the type stored in the list. The caller is responsible for allocation and deallocation of the buffer space.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

If the list is empty, i.e., it has zero elements, then the function will return RFAXERR_LIST_EMPTY. If the caller attempts to set an element beyond the end of the list, then the function returns RFAXERR_LIST_END.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElement() on page 262

RFaxListSetElementCount() below

RFaxListSetElementCount()

This function changes the maximum number of elements/items that can be stored in a list.

C Prototype

```
DWORD RFAXAPI RFaxListSetElementCount(
 IN RFAXLISTHANDLE hList,
 IN unsigned long ulElementCount
);
```

Arguments

hList	Handle of the list.
ulElementCount	The new size of the list, in elements.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

The size of the list can be changed at any time. Shrinking a list will cause the loss of the tail, or higher indexed, elements. Expanding a list will add additional element storage space to the end of the list. Any added element space will be initialized to zero.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

RFaxListGetElementSize() on page 263

RFaxListGetElementCount() on page 262

RFaxListSort()

This function sorts the list.

C Prototype

```
DWORD RFAXAPI RFaxListSort(
 IN RFAXLISTHANDLE hList,
 IN int (*CompareFunc)( const void
 *Element1,const void *Element2 )
);
```

Arguments

hList	Handle of the record to sort
CompareFunc	A user-supplied comparison function. This function is called to compare individual elements in the list. The function is passed pointers to two elements and should return an integer value based on the comparison: **Return valueComparison result** <0 p1 is less than p2 0 p1 equals p2 >0 p1 is greater than p2

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function sorts the list. The current element is re-set to 0.

Compatibility

Compatible with all RightFax versions.

See Also

RFaxCreateList() on page 258

Chapter 30: Workflow functions

The worklow functions create, edit, load, and delete workflows.

RFaxCompleteWorkflow()

This function attempts to complete a workflow.

C Prototype

```
DWORD RFAXAPI RFaxCompleteWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IOPTIONAL IN FAXHANDLE hCompletionWorkflow
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of fax to complete its current workflow.
hCompletionWorkflow	Optional workflow to which to submit the fax upon completion.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function attempts to complete a workflow. All required values must have been already submitted.

See Also

RFaxStartWorkflow() on page 285

RFaxCompleteWorkflowFax()

This function attempts to complete a workflow.

C Prototype

```
DWORD RFAXAPI RFaxCompleteWorkflowFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 OPTIONAL IN FAXHANDLE hCompletionWorkflow
 OPTIONAL IN RFAXCSTR pszComment
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of fax to complete its current workflow.
hCompletionWorkflow	Optional workflow to which to submit the fax upon completion.

pszComment	Optional comment for completion history record, up to 126 characters.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function attempts to complete a workflow. All required values must have been already submitted.

See Also

RFaxExceptWorkflowFax() on page 273

RFaxRejectWorkflowFax() on page 281

RFaxDeleteWorkflow()

This function deletes an existing workflow.

C Prototype

```
DWORD RFAXAPI RFaxDeleteWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function deletes a workflow but does not delete any of the corresponding values saved to faxes while they were in the workflow.

See Also

RFaxSaveWorkflow() on page 282

RFaxDeleteWorkflowDelegation()

This function unlinks a user or group from a workflow.

C Prototype

```
DWORD RFAXAPI RFaxDeleteWorkflowDelegation(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel 10,
 IN PWORKFLOWDELEGATIONINFO_10 pdelegation,
 IN ULONG ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level. Specify 10.
pdelegation	Delegation to delete.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success

Return	Result
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowDelegation() on page 282

RFaxDeleteWorkflowField()

This function deletes a workflow field.

C Prototype

```
DWORD RFAXAPI RFaxDeleteWorkflowField(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE handle,
 IN OPTIONAL FAXHANDLE hWorkflow
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of workflow field that should be deleted or NULL.
hWorkflow	Handle of workflow whose fields should be deleted or NULL.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowField() on page 283

RFaxDeleteWorkflowFieldChoices()

This function deletes the metadata choices selected by a user in a workflow.

C Prototype

```
DWORD RFAXAPI RFaxDeleteWorkflowFieldChoices(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN RFAXLISTHANDLE hList,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Level of information must be 10.
hList	List of *WORKFLOWFIELDCHOICEINFO_10* on page 389 to delete.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowFieldChoices() on page 283

RFaxGetWorkflowFieldChoiceList() on page 276

RFaxDeleteWorkflowIntelligence()

This function deletes the portion of a workflow that is associated with Intelligent Workflows.

C Prototype

```
DWORD RFAXAPI RFaxDeleteWorkflowIntelligence(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN UINT ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the Intelligent Workflow.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowIntelligence() on page 284

RFaxDeleteWorkflowRelations()

This function removes links from a workflow to other workflows.

C Prototype

```
DWORD RFAXAPI RFaxDeleteWorkflowRelations(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN RFAXLISTHANDLE hList,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 0 or 10.
hList	List of *WORKFLOWRELATIONLISTELEMENT0* on page 392or *WORKFLOWRELATIONINFO_10* on page 493 to delete.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowRelations() on page 284

RFaxGetWorkflowRelationList() on page 278

RFaxExceptWorkflowFax()

This function attempts to except a fax from a workflow.

C Prototype

```
DWORD RFAXAPI RFaxExceptWorkflowFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 OPTIONAL IN FAXHANDLE hExceptionWorkflow
 OPTIONAL IN RFAXCSTR pszComment
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of fax to except from its current workflow.
hCompletionWorkflow	Optional workflow to which to submit the fax upon exception.
pszComment	Optional comment for completion history record, up to 126 characters.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function attempts to except a fax from a workflow. All required values must have been already submitted.

See Also

RFaxCompleteWorkflowFax() on page 270

RFaxRejectWorkflowFax() on page 281

RFaxExtractWorkflowIntelligence()

This function instigates a capture using the specified intelligence on the given fax and downloads results to the specified file. The workflow values of the fax will be updated with the captured fields.

C Prototype

```
DWORD RFAXAPI RFaxExtractWorkflowIntelligence(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN FAXHANDLE hWorkflowIntelligence,
 IN RFAXCSTR pszOutFilename,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Fax for which the path should be set.
hWorkflowIntelligence	Handle of the Intelligent Workflow.
pszOutFilename	Path to the file where results will be written.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxExtractWorkflowIntelligenceForImage() below

RFaxExtractWorkflowIntelligenceForImage()

This function uploads the specified image file to the server which performs an extraction based on the intelligence provided in szRecoSequentialInput. In contrast to *RFaxExtractWorkflowIntelligenceForImage()* above, it doesn't use a fax or the intelligence in the database but rather the provided information.

C Prototype

```
DWORD RFAXAPI
RFaxExtractWorkflowIntelligenceForImage(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR szlocalImgfilepath,
 IN RFAXCSTR szRecoSequentialInput,
 IN RFAXCSTR pszOutFilename,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
szlocalImgfilepath	Local path to the TIFF image.
szRecoSequentialInput	OCR sequential input as JSON string.
pszOutFilename	Path to the file where results will be written.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxExtractWorkflowIntelligence() on the previous page

RFaxGetWorkflowCounts()

This function gets a count of faxes for each workflow up to the maximum that will fit in the given array. You have the option to get a count for a specific user.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowCounts(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
 OUT WORKFLOW_FAX_COUNT paCounts,
 IN WORKFLOW_FAX_COUNT paCounts,
 IN USHORT usMaxCounts,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usOptions	Reserved.
hUser	Handle of user whose faxes are to be considered or NULL for all users
paCounts	In: Array of *WORKFLOW_FAX_COUNT* on page 388 Out: On success, up to usMaxCounts *WORKFLOW_FAX_COUNT* on page 388 elements.
usMaxCounts	Maximum number of counts to return.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxGetWorkflowList() on page 277

RFaxGetWorkflowDelegationList()

This function gets the users and groups that are linked to a workflow.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowDelegationList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hWorkflow,
 IN USHORT usInfoLevel 0,
 IN ULONG ulReserved,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hWorkflow	Handle of workflow whose delegations should be listed.
usInfoLevel	Information level. Specify 0.
ulReserved	Reserved for future use. Specify 0.
*lpLH	If return = 0, returns a handle to list of *WORKFLOWDELEGATIONLISTELEMENT0* on page 389structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowDelegation() on page 282

RFaxDeleteWorkflowDelegation() on page 271

RFaxGetWorkflowFieldChoiceList()

This function gets a list of metadata choices for fields that can be selected by a user in a workflow.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowFieldChoiceList(
 IN RFAXSERVERHANDLE hServer,
 IN SHORT sWhichHandle,
 IN FAXHANDLE handle,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
sWhichHandle	0 if handle refers to a single field for which choices should be listed. 1 if handle refers to a workflow for which choices should be listed.
handle	Handle of field or workflow for which choices should be listed. See sWhichHandle.
usInfoLevel	Level of information must be 10.
ulReserved	Reserved for future use. Specify 0.
lpLH	If return value is 0, returns a handle to list of *WORKFLOWFIELDCHOICEINFO_10* on page 389 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowFieldChoices() on page 272

RFaxGetWorkflowFieldChoiceList() on the previous page

RFaxGetWorkflowFieldList()

This function gets a list of fields associated with a workflow.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowFieldList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hWorkflow,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hWorkflow	Workflow for which to list the fields, or Null for all.
usInfoLevel	Information level 10.
ulReserved	Reserved for future use. Specify 0.
*lpLH	If return = 0, returns a handle to list of *WORKFLOWFIELDINFO_10* on page 390 structures

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadWorkflowField() on page 280

RFaxSaveWorkflowField() on page 283

RFaxDeleteWorkflowField() on page 272

RFaxGetWorkflowIntelligenceList()

This function gets a list of Intelligent Workflows.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowIntelligenceList(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN UINT ulReserved,
 IN ULONG ulReserved2,
 OUT RFAXLISTHANDLE FAR* lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level. Must be 10.
ulReserved	Reserved for future use. Specify 0.
ulReserved2	Reserved for future use. Specify 0.
lpLH	If return == 0, returns a handle to list of WORKFLOWINTELLIGENCEINFO_10.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxGetWorkflowList()

This function gets a list of workflows.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hUser,
```

```
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hUser	User whose workflows should be listed or NULL for all.
usInfoLevel	Information level 0 or 10.
ulReserved	Reserved for future use. Specify 0.
*lpLH	If return = 0, returns a handle to a list of *WORKFLOWLISTELEMENT0* on page 391 or *WORKFLOWINFO_10* on page 390 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxGetWorkflowCounts() on page 275

RFaxGetWorkflowRelationList()

This function gets the workflows that are linked to the specified workflow.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowRelationList(
 IN RFAXSERVERHANDLE hServer,
 IN OPTIONAL FAXHANDLE hWorkflow,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved
 OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hWorkflow	Handle of workflow for which relations should be listed.
usInfoLevel	Information level 0 or 10.
hList	List of *WORKFLOWRELATIONLISTELEMENT0* on page 392or *WORKFLOWRELATIONINFO_10* on page 493 to delete.
ulReserved	Reserved for future use. Specify 0.
*lpLH	If return value is 0, returns a handle to list of WORKFLOWRELATIONLISTELEMENT0 or WORKFLOWRELATIONINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowRelations() on page 273

RFaxSaveWorkflowRelations() on page 284

RFaxGetWorkflowValueList()

This function gets a list of workflow values for the specified fax.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkflowValueList(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN FAXHANDLE hWorkflow,
```

```
    IN RFAXCSTR pszWorkflowName,
    IN USHORT usInfoLevel,
    IN ULONG ulReserved,
    OUT RFAXLISTHANDLE FAR *lpLH
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Fax for which the workflow values should be listed.
hWorkflow	Handle of the workflow for which values should be listed, or Null.
pszWorkflowName	Name of the workflow for which values should be listed, or Null.
usInfoLevel	Information level 10.
ulReserved	Reserved for future use. Specify 0.
*lpLH	If return = 0, returns a handle to list of WORKFLOWVALUEINFO_10 structures.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxSaveWorkflowValue() on page 284

RFaxSaveWorkflowValues() on page 285

RFaxLoadWorkflow()

This function loads a workflow.

C Prototype

```
DWORD RFAXAPI RFaxLoadWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE handle,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT RFAXPVOID pworkflow
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	
usInfoLevel	Information level, 10 or 0.
ulReserved	Reserved for future use. Specify 0.
pworkflow	*WORKFLOWINFO_10* on page 390 or *WORKFLOWLISTELEMENT0* on page 391.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxLoadWorkflowByName()

This function loads a workflow.

C Prototype

```
DWORD RFAXAPI RFaxLoadWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR pszName,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT RFAXPVOID pworkflow
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszName	Name of workflow to load.
usInfoLevel	Information level, 10 or 0.
ulReserved	Reserved for future use. Specify 0.
pworkflow	*WORKFLOWINFO_10* on page 390 or *WORKFLOWLISTELEMENT0* on page 391.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxLoadWorkflowField()

This function loads a workflow field.

C Prototype

```
DWORD RFAXAPI RFaxLoadWorkflowField(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hField,
 IN USHORT usInfoLevel,
 IN ULONG ulReserved,
 OUT PWORKFLOWFIELDINFO_10 pworkflowfieldinfo
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hField	Handle of workflow field to load.
usInfoLevel	Information level must be 10.
ulReserved	Reserved for future use. Specify 0.
pworkflowfieldinfo	

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxGetWorkflowFieldList() on page 277

RFaxSaveWorkflowField() on page 283

RFaxDeleteWorkflowField() on page 272

RFaxLoadWorkflowIntelligence()

This function loads an Intelligent Workflow.

C Prototype

```
DWORD RFAXAPI RFaxLoadWorkflowIntelligence(
 IN RFAXSERVERHANDLE hServer,
```

```
 IN FAXHANDLE handle,
 IN USHORT usInfoLevel,
 IN UINT ulReserved,
 OUT RFAXPVOID pWorkflowIntelligence,
 IN RFAXCSTR pszZonalInfoFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
handle	Handle of the Intelligent Workflow.
usInfoLevel	Information level. Must be 10.
ulReserved	Reserved for future use. Specify 0.
pWorkflowIntelligence	WORKFLOWINTELLIGENCEINFO_10 *.
pszZonalInfoFile	Optional path to file to receive zonal information.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadWorkflowIntelligenceByWorkflow() below

RFaxLoadWorkflowIntelligenceByWorkflow()

This function loads an Intelligent Workflow.

C Prototype

```
DWORD RFAXAPI
RFaxLoadWorkflowIntelligenceByWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hWorkflow,
 IN USHORT usInfoLevel,
 IN UINT ulReserved,
 OUT RFAXPVOID pWorkflowIntelligence,
 IN RFAXCSTR pszZonalInfoFile
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hWorkflow	Handle of the Intelligent Workflow.
usInfoLevel	Information level. Must be 10.
ulReserved	Reserved for future use. Specify 0.
pWorkflowIntelligence	WORKFLOWINTELLIGENCEINFO_10 *.
pszZonalInfoFile	Optional path to file to receive zonal information.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxLoadWorkflowIntelligence() on the previous page

RFaxRejectWorkflowFax()

This function attempts to reject a fax in a workflow.

C Prototype

```
DWORD RFAXAPI RFaxRejectWorkflowFax(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 OPTIONAL IN FAXHANDLE hRejectionWorkflow
 OPTIONAL IN RFAXCSTR pszComment
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle of fax to reject from its current workflow.
hCompletionWorkflow	Optional workflow to which to submit the fax upon rejection.
pszComment	Optional comment for completion history record, up to 126 characters.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function attempts to reject a fax from a workflow. All required values must have been already submitted.

See Also

RFaxCompleteWorkflowFax() on page 270

RFaxExceptWorkflowFax() on page 273

RFaxSaveWorkflow()

This function saves a workflow.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel 10,
 IN PWORKFLOWINFO_10 pworkflowinfo,
 IN ULONG ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level. Must be 10.
pworkflowinfo	
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflow() on page 271

RFaxSaveWorkflowDelegation()

This function links a user or group to a workflow.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowDelegation(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel 10,
 IN PWORKFLOWDELEGATIONINFO_10 pdelegation,
 IN ULONG ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level. Specify 10.
pdelegation	Delegation to save.

ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowDelegation() on page 271

RFaxSaveWorkflowField()

This function saves a workflow field.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowField(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 OUT PWORKFLOWFIELDINFO_10 pworkflowfieldinfo
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level must be 10.
pworkflowfieldinfo	Workflow field information to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowField() on page 272

RFaxSaveWorkflowFieldChoices()

This function saves the metadata choices selected by a user in a workflow.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowFieldChoices(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN RFAXLISTHANDLE hList,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Level of information must be 10.
hList	List of *WORKFLOWFIELDCHOICEINFO_10* on page 389 to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowFieldChoices() on page 272

RFaxGetWorkflowFieldChoiceList() on page 276

RFaxSaveWorkflowIntelligence()

This function creates or updates an Intelligent Workflow. While creating, the WORKFLOWINTELLIGENCEINFO_10.handle should be left untouched. If updating, specify the WORKFLOWINTELLIGENCEINFO_10.handle.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowIntelligence(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel 10,
 IN PWORKFLOWINTELLIGENCEINFO_10
pWorkflowIntelligence,
 IN RFAXCSTR pszZonalInfo,
 IN ULONG ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level. Must be 10.
pWorkflowIntelligence	
pszZonalInfo	Optional. Zonal information to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowIntelligence() on page 272

RFaxSaveWorkflowRelations()

This function links a workflow to other workflows.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowRelations(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN RFAXLISTHANDLE hList,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 0 or 10.
hList	List of *WORKFLOWRELATIONLISTELEMENT0* on page 392or *WORKFLOWRELATIONINFO_10* on page 493 to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxDeleteWorkflowRelations() on page 273

RFaxGetWorkflowRelationList() on page 278

RFaxSaveWorkflowValue()

This function saves a workflow value for the workflow, field, and fax specified in the WORKFLOWVALUEINFO.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowValue(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN PWORKFLOWVALUEINFO_10 pworkflowvalueinfo,
 IN ULONG ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 10.
pworkflowvalueinfo	Workflow value information to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxGetWorkflowValueList() on page 278

RFaxSaveWorkflowValues() below

RFaxSaveWorkflowValues()

This function saves multiple workflow values at the same time.

C Prototype

```
DWORD RFAXAPI RFaxSaveWorkflowValues(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usInfoLevel,
 IN RFAXLISTHANDLE hValuesList,
 IN ULONG ulReserved,
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usInfoLevel	Information level 10.
hValuesList	List of *WORKFLOWVALUEINFO_10* on page 392 values to save.
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Using this function to save multiple values simultaneously can reduce the amount of data produced by saving multiple values separately, because each save creates a history record..

See Also

RFaxGetWorkflowValueList() on page 278

RFaxSaveWorkflowValue() on the previous page

RFaxStartWorkflow()

This function starts the specified workflow on the given fax.

C Prototype

```
DWORD RFAXAPI RFaxStartWorkflow(
 IN RFAXSERVERHANDLE hServer,
 IN FAXHANDLE hFax,
 IN FAXHANDLE hWorkflow,
 IN ULONG ulReserved
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
hFax	Handle to the fax that will enter the specified workflow.
hWorkflow	Workflow to start for this fax
ulReserved	Reserved for future use. Specify 0.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxCompleteWorkflow() on page 270

Chapter 31: Miscellaneous functions

Miscellaneous functions are available to perform a variety of tasks.

RFaxCheckDatabaseLock()

This function changes the password of a user ID.

C Prototype

```
DWORD RFaxCheckDatabaseLock(
 IN RFAXSERVERHANDLE hServer,
 OUT LOCKDBDATA pdblock
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle2().
pdblock	Lock information returned on success.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

See Also

RFaxCreateServerHandle2() on page 27

RFaxCTimeToValues()

This function converts a number from ctime to Gregorian time.

C Prototype

```
VOID RFAXAPI RFaxCTimeToValues(
 ULONG ulCTime,
 short *lpsYear,
 short *lpsMonth,
 short *lpsDay,
 short *lpsHour,
 short *lpsMin,
 short *lpsSec
);
```

Arguments

ulCTime	Specify the ctime to convert.
lpsYear	Returns the year in the format YYYY.
lpsMonth	Returns the month in the format MM.
lpsDay	Returns the day in the format DD.
lpsHour	Returns the hour in the format HH.
lpsMin	Returns the minute in the format MM.
lpsSec	Returns the second in the format SS.

Return Values

None.

Remarks

In RightFax, time is stored in ctime, or the number of seconds since 12:00 A.M. January 1, 1970. This function converts it from ctime to Gregorian time.

See Also

RFaxValuesToCTime() on page 305

RFaxG3ToMFTIFF()

This function converts the image from RightFax G3 format (.301, .302, etc.) into a set of single-page TIFF Class F files, e.g., one TIFF file per page of fax.

C Prototype

```
DWORD RFAXAPI RFaxG3ToMFTIFF(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXSTR lpDestFile,
 IN OPTIONAL RFAXCSTR lpImageDesc,
 IN OPTIONAL RFAXCSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.
sNumPages1	Number of pages that you want to export to the TIFF file. Must be at least 1.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages in the image specified in lpSourceFile2.
lpDestFile	Specify the path and file name of the first TIFF file. The file name should be in the form BASE###.TIF, where ### is a three-digit page number. The page number will be incremented for each destination page, e.g., BASE001.TIF, BASE002.TIF.
lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different RightFax G3 files that will be combined in the sequence you specify them. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information about the fax in the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToTIFF() on page 291

RFaxFileRead() on page 35

RFaxLoadFax() on page 96

RFaxG3ToTIFF()

This function converts the image from RightFax G3 format (.301, .302, etc.) into a single-file, multi-page TIFF file using G3 compression, also known as TIFF Class F.

C Prototype

```
DWORD RFAXAPI RFaxG3ToTIFF(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXCSTR lpDestFile,
 IN OPTIONAL RFAXSTR lpImageDesc,
 IN OPTIONAL RFAXSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.
sNumPages1	Number of pages to export to the TIFF file. Must be at least 1. The number of pages can be determined from the numpages member of the FAXINFO_10 or FAXINFO_11 structure.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages to export from the image specified in lpSourceFile2. The number of pages can be determined from the numpages member of the FAXINFO_10 or FAXINFO_11 structure.
lpDestFile	Specify the path and file name of the TIFF image including the extension (i.e., .tif). If the file already exists, it will be overwritten.
lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different RightFax G3 files that will be combined in the sequence you specify them. This function should not be used to combine two faxes. Instead, use RFaxCombineFaxes() to combine two or more faxes. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information inside the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToMFTIFF()

RFaxG3ToPDF()

RFaxG3ToPDF2()

RFaxFileRead()

RFaxLoadFax()

RFaxG3ToPDF()

This function converts the image from RightFax G3 format (.301, .302, etc.) into PDF.

C Prototype

```
DWORD RFAXAPI RFaxG3ToPDF(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXCSTR lpDestFile,
 IN OPTIONAL RFAXCSTR lpImageDesc,
 IN OPTIONAL RFAXCSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.
sNumPages1	Number of pages that you want to export to the PDF file. Must be at least 1.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages in the image specified in lpSourceFile2.
lpDestFile	Specify the path and file name of the first PDF file. The file name should be in the form BASE###.PDF, where ### is a three-digit page number. The page number will be incremented for each destination page, e.g., BASE001.PDF, BASE002.PDF.
lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different RightFax G3 files that will be combined in the sequence you specify them. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information about the fax in the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToTIFF()

RFaxG3ToPDF()

RFaxFileRead()

RFaxLoadFax()

RFaxG3ToPDF2()

This function converts the image from RightFax G3 format (.301, .302, etc.) into PDF.

C Prototype

```
DWORD RFAXAPI RFaxG3ToPDF2(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXCSTR lpDestFile,
 IN OPTIONAL RFAXCSTR lpImageDesc,
 IN OPTIONAL RFAXCSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.

sNumPages1	Number of pages that you want to export to the PDF file. Must be at least 1.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages in the image specified in lpSourceFile2.
lpDestFile	Specify the path and file name of the first PDF file. The file name should be in the form BASE###.PDF, where ### is a three-digit page number. The page number will be incremented for each destination page, e.g., BASE001.PDF, BASE002.PDF.
lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).
thumbnailsInPdfs	Include thumbnail images.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different RightFax G3 files that will be combined in the sequence you specify them. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information about the fax in the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToTIFF()

RFaxG3ToPDF()

RFaxFileRead()

RFaxLoadFax()

RFaxG3ToTIFF()

This function converts the image from RightFax G3 format (.301, .302, etc.) into a single-file, multi-page TIFF file using G3 compression, also known as TIFF Class F.

C Prototype

```
DWORD RFAXAPI RFaxG3ToTIFF(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXCSTR lpDestFile,
 IN OPTIONAL RFAXSTR lpImageDesc,
 IN OPTIONAL RFAXSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.
sNumPages1	Number of pages to export to the TIFF file. Must be at least 1. The number of pages can be determined from the numpages member of the FAXINFO_10 or FAXINFO_11 structure.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages to export from the image specified in lpSourceFile2. The number of pages can be determined from the numpages member of the FAXINFO_10 or FAXINFO_11 structure.
lpDestFile	Specify the path and file name of the TIFF image including the extension (i.e., .tif). If the file already exists, it will be overwritten.

lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different RightFax G3 files that will be combined in the sequence you specify them. This function should not be used to combine two faxes. Instead, use RFaxCombineFaxes() to combine two or more faxes. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information inside the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToMFTIFF()

RFaxG3ToPDF()

RFaxG3ToPDF2()

RFaxFileRead()

RFaxLoadFax()

RFaxGetAPIVersion()

This function gets the compilation version (time) of the RightFax API.

C Prototype

```
DWORD RFAXAPI RFaxGetAPIVersion(
void );
```

Arguments

None.

Return Values

Returns the compilation version as a ctime (time_t) value. This value is the number of seconds elapsed since 12:00 midnight, January 1, 1970.

Remarks

This function identifies the RightFax API by compilation time.

See Also

RFaxCTimeToValues() on page 287

RFaxValuesToCTime() on page 305

RFaxVersionCheck() on page 305

RFaxGetLocalSupportInfo()

This function sets the support information from the server.

C Prototype

```
DWORD RFAXAPI RFaxGetLocalSupportInfo(
 IN RFAXSERVERHANDLE hServer,
 OUT LOCALSUPPORTINFO *pInfo
);
```

Arguments

hServer	Specify a handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pInfo	Pointer to LOCALSUPPORTINFO structure to receive local support information.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Gets the local support information from the server bin folder in the file Contact.txt. Gets the number of lines that will fit in the LOCALSUPPORTINFO structure (24 lines).

RFaxGetObjectCount()

This function gets a count of objects in RightFax.

C Prototype

```
DWORD RFAXAPI RFaxGetObjectCount(
 IN RFAXSERVERHANDLE hServer,
 IN unsigned short usObjectType,
 IN RFAXCSTR lpUserID,
 IN FAXHANDLE hHandle,
 OUT long *lpCount
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usObjectType	Specify one of the COUNT_??? constants to specify the information to count.
lpUserID	Specify the user ID if usObjectType is set to COUNT_FAXES, COUNT_PHONES, COUNT_DEVICETYPES: type; integration t ype
hHandle	Specify the handle of the fax if usObjectType is set to COUNT_HISTORY.
lpCount	Returns the total number of the objects specified.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function is useful for getting a count of specific elements.

RFaxGetServerTimeBias()

This function gets the server UTC bias (offset to GMT) in minutes.

C Prototype

```
ULONG RFAXAPI RFaxGetServerTimeBias(
 IN RFAXSERVERHANDLE hServer,
 IN PSERVERINFO2 pServerInfo,
 OUT PLONG plServerBias
);
```

Arguments

hServer	Handle of the RightFax server if pServerInfo argument is Null.
pServerInfo	Previously loaded server information or use Null to query server information.
plServerBias	Server bias in minutes.

Return Values

Return	Result
RFAX_SUCCESS	Success.
RFAXERR_INVALID_PARAMETER	Both hServer and pServerInfo arguments are Null. One must be non-Null.
RFAXERR_INVALID_PARAMETER	The plServerBias argument is Null.
RFAXERR_INVALID_HANDLE	The hServer argument (via RFaxGetServerInfo2()) is invalid.
RFAXERR_BADPARM	The time zone information in pServerInfo is invalid.

See Also

RFaxLocalToServerTime()

RFaxServerToLocalTime()

RFaxGetTAPIDeviceList()

This function returns a list of TAPI device strings.

C Prototype

```
DWORD RFAXAPI RFaxGetTAPIDeviceList(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXSTR lpBuffer,
 IN int iSizeBuffer
);
```

Arguments

hServer	Specify a handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpBuffer	Returns a list of strings identifying the TAPI devices on the server. This list consists of strings delimited by Nulls (zeros). The final string is terminated with two Nulls in a row.
iSizeBuffer	Specify the length of the buffer at lpBuffer.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function returns a list of strings representing TAPI devices.

RFaxGetUniqueFileName()

This function obtains a unique file name from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetUniqueFileName(
 IN RFAXSERVERHANDLE hServer,
 OUT RFAXSTR pszFileName
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
pszFileName	Returns the base name for a unique file name.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to get a unique file name from the RightFax server before copying files to be converted. This will ensure that the file will not be overwritten by another application.

See Also

RFaxGetUniqueFileNumber() below

RFaxGetUniqueFileNumber()

This function obtains a unique file number from the fax server.

C Prototype

```
DWORD RFAXAPI RFaxGetUniqueFileNumber(
 IN RFAXSERVERHANDLE hServer,
 IN int iNumFiles,
 OUT unsigned long *pulUniqueNum
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
iNumFiles	The number of file numbers that are needed. iNumFiles unique numbers are reserved starting with the value returned by this call. This value should be 1 or more.
pulUniqueNum	The first in the sequence of unique file numbers.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

Use this function to get a unique file number from the RightFax server before copying files to be converted. This will ensure that the file will not be overwritten by another application. Multiple file numbers can be requested by specifying more than 1 for iNumFiles. The value returned in pulUniqueNum is the first in the sequence of unique file numbers.

See Also

RFaxGetUniqueFileName() on the previous page

RFaxGetWorkRequestCounts()

This function gets the number of work requests currently queued on the server.

C Prototype

```
DWORD RFAXAPI RFaxGetWorkRequestCounts(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulReqFilter,
 IN RFAXPVOID reserved2,
 IN RFAXPVOID reserved3,
 IN RFAXPVOID reserved4,
 IN SHORT sInfoLevel,
 OUT RFAXPVOID lpReturnInfo
);
```

Arguments

hServer	Specify the handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2()
ulReqFilter	Specify a combination of REQUEST_??? bit flags.
reserved2	Reserved.(0 or null).
reserved3	Reserved.(0 or null).
reserved4	Reserved.(0 or null).
sInfoLevel	Specify the level of information. Must be 0.
lpReturnInfo	Address of WORKREQCOUNTS_0 buffer.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function retrieves the number of work requests currently queued on the server.

See Also

RFaxCreateServerHandle() on page 26*RFaxCreateServerHandle2()* on page 27

RFaxIntToImageExt()

This function converts a page value (1–*n*) to "3##", the RightFax format for counting pages.

C Prototype

```
RFAXSTR RFAXAPI RFaxIntToImageExt(
 IN int iPage,
 OUT RFAXSTR lpExt
);
```

Arguments

iPage	Integer page value (1–*n*) to convert to "3##" format.
lpExt	Returns the long pointer converted to "3##".

Return Values

Returns the long pointer converted to "3##".

Remarks

This function converts the RightFax format of counting pages to a base 10 version of the number.

See Also

RFaxPageExtToInt() on page 299

RFaxLocalToServerTime()

This function converts a local ctime value to a server ctime value.

C Prototype

```
ULONG RFAXAPI RFaxLocalToServerTime(
 IN RFAXSERVERHANDLE hServer,
 IN PSERVERINFO2 pServerInfo,
 IN LONG timeLocal,
 OUT LONG* ptimeServer
);
```

Arguments

hServer	Handle of the RightFax server if pServerInfo argument is Null.
pServerInfo	Previously loaded server information or use Null to query server information.
timeLocal	Local time_t value to convert or zero for offset.
ptimeServer	Corresponding server time_t value.

Return Values

Return	Result
RFAX_SUCCESS	Success.
RFAXERR_INVALID_PARAMETER	Both hServer and pServerInfo arguments are Null. One must be non-Null.
RFAXERR_INVALID_PARAMETER	The ptimeServer argument is Null.
RFAXERR_INVALID_HANDLE	The hServer argument (via RFaxGetServerInfo2()) is invalid.
RFAXERR_BADPARM	The time zone information in pServerInfo is invalid.

Remarks

To obtain the offset between the server bias and the local bias (local bias relative to the server), pass in zero for timeLocal. The result is the relative bias in seconds.

See Also

RFaxGetServerTimeBias() on page 293

RFaxMPTIFFToG3()

This function converts a multi-page TIFF file to RightFax G3 format (.301, .302, etc.)

C Prototype

```
DWORD RFAXAPI RFaxMPTIFFToG3(
 IN RFAXCSTR lpSourceFile,
 IN RFAXCSTR lpDestFile,
 IN OPTIONAL short sNumPages,
 IN OUT long *plDestPageNum
);
```

Arguments

lpSourceFile	Complete file name of the TIFF source (including the file name extension).
lpDestFile	Destination file name (without the file name extension).
sNumPages	Number of pages to pull from the source TIFF file. Specify 0 for all pages.
plDestPageNum	On inbound, specify the starting page and on outbound, the destination page number (1-based). This number is modified during the call to reflect the next page to be converted in destination file.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts the files directly. It does not need the fax server to be running.

The destination file is relative to the computer executing the API call. To be specific, you can use UNC format to specify the path.

See Also

RFaxG3ToTIFF() on page 291

RFaxG3ToMFTIFF() on page 288

RFaxFileRead() on page 35

RFaxLoadFax() on page 96

RFaxMPTIFFToPDF()

This function converts a multi-page TIFF file to PDF.

C Prototype

```
DWORD RFAXAPI RFaxMPTIFFToPDF(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXCSTR lpDestFile,
 IN OPTIONAL RFAXCSTR lpImageDesc,
 IN OPTIONAL RFAXCSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.
sNumPages1	Number of pages that you want to export to the PDF file. Must be at least 1.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages in the image specified in lpSourceFile2.
lpDestFile	Specify the path and file name of the first PDF file. The file name should be in the form BASE###.PDF, where ### is a three-digit page number. The page number will be incremented for each destination page, e.g., BASE001.PDF, BASE002.PDF.
lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different TIFF files that will be combined in the sequence you specify them. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information about the fax in the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToTIFF()

RFaxFileRead()

RFaxLoadFax()

RFaxMPTIFFToPDF2()

This function converts the image from TIFF file format to PDF.

C Prototype

```
DWORD RFAXAPI RFaxMPTIFFToPDF2(
 IN RFAXCSTR lpSourceFile1,
 IN short sNumPages1,
 IN OPTIONAL RFAXCSTR lpSourceFile2,
 IN OPTIONAL short sNumPages2,
 IN RFAXSTR lpDestFile,
 IN OPTIONAL RFAXCSTR lpImageDesc,
 IN OPTIONAL RFAXCSTR lpDateTime
);
```

Arguments

lpSourceFile1	Name of first file to convert. The .3?? extension is optional.
sNumPages1	Number of pages to convert. Must be at least 1.
lpSourceFile2	Second file name to convert.
sNumPages2	Number of pages in the image specified in lpSourceFile2.
lpDestFile	Specify the path and file name of the first PDF file. The file name should be in the form BASE###.PDF, where ### is a three-digit page number. The page number will be incremented for each destination page, e.g., BASE001.PDF, BASE002.PDF.
lpImageDesc	Optional string for TIFF tag #270 (IMAGE DESCRIPTION).
lpDateTime	Optional string for TIFF tag #306 (IMAGE DATE/TIME).
thumbnailsInPdfs	Include thumbnail images.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function converts files directly and does not require the fax server to be running. You can specify two different TIFF files that will be combined in the sequence you specify. The destination file is relative to the computer executing the API call. If you are not sure, you can use UNC format to specify the path. The optional TIFF tags are useful for storing information about the fax in the image itself, but you must use a third-party application to extract that information.

See Also

RFaxG3ToTIFF() on page 291

RFaxFileRead() on page 35

RFaxLoadFax() on page 96

RFaxPageExtToInt()

This function converts a page value from the RightFax format of counting pages (".3##" or "3##") to integer page value (1–*n*).

C Prototype

```
int RFAXAPI RFaxPageExtToInt(
 IN RFAXCSTR lpExt
);
```

Arguments

lpExt	Long pointer to the extension to convert to integer page values.

Return Values

The integer value of the page numbers.

Remarks

This function is useful for determining the number of pages for a fax.

See Also

RFaxIntToImageExt() on page 296

RFaxSelectDefaultLanguage()

This function specifies the default language that is used in localizing globalized strings such as history details, fax status, term stats, error codes, and error results.

C Prototype

```
DWORD RFAXAPI RFaxSelectDefaultLanguage(
 IN RFAXCSTR lpLangSelection Spanish,
 IN USOPTIONS 0
);
```

Arguments

lpLangSelection	Language choice (Spanish, German, English, French, French-Canadian, Italian, Japanese, Korean, Portuguese, Chinese, Arabic, Russian).
usOptions	Reserved.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSelectLanguage()

This function specifies the language that is used in localizing globalized strings such as history details, fax status, term stats, error codes, and error results.

C Prototype

```
DWORD RFAXAPI RFaxSelectLanguage(
 IN RFAXSERVERHANDLE hServer,
 IN RFAXCSTR lpLangSelection Spanish,
 IN USOPTIONS 0
);
```

Arguments

hServer	Handle of the RightFax server created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
lpLangSelection	Language choice (Spanish, German, English, French, French-Canadian, Italian, Japanese, Korean, Portuguese, Chinese, Arabic, Russian).
usOptions	Reserved.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

RFaxSendFSMsg()

This function sends a message to the fax server.

C Prototype

```
DWORD RFAXAPI RFaxSendFSMsg(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT usMsgType,
 IN OPTIONAL FAXHANDLE handle,
 IN OPTIONAL RFAXCSTR lpUserID,
 IN OPTIONAL RFAXCSTR lpInfo,
 IN OPTIONAL ULONG ulParam1
);
```

Arguments

hServer	Specify a handle that was created using RFaxCreateServerHandle() or RFaxCreateServerHandle2().
usMsgType	The type of message to send. Specify one of the FSMSG_??? constants.
handle	Handle, dependent upon the message type.
lpUserID	ID of the user motivating action, dependent upon the message type.
lpInfo	Information string, dependent upon the message type.
ulParam1	Argument, dependent upon the message type.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function sends a message to the fax server. The argument definitions are dependent upon the message being sent. See FSMSG_??? for details.

See Also

RFaxCreateServerHandle() on page 26

RFaxCreateServerHandle2() on page 27

FSMSG_??? on page 427

RFaxServerToLocalTime()

This function converts a server ctime value to a local ctime value.

C Prototype

```
ULONG RFAXAPI RFaxServerToLocalTime(
 IN RFAXSERVERHANDLE hServer,
 IN PSERVERINFO2 pServerInfo,
 IN LONG timeServer,
 OUT LONG* ptimeLocal
);
```

Arguments

hServer	Handle of the RightFax server if pServerInfo argument is Null.
pServerInfo	Previously loaded server information or use Null to query server information.
timeServer	Server time_t value to convert or zero for offset.
ptimeLocal	Corresponding local time_t value.

Return Values

Return	Result
RFAX_SUCCESS	Success.
RFAXERR_INVALID_PARAMETER	Both hServer and pServerInfo arguments are Null. One must be non-Null.
RFAXERR_INVALID_PARAMETER	The ptimeLocal argument is Null.
RFAXERR_INVALID_HANDLE	The hServer argument (via RFaxGetServerInfo2()) is invalid.
RFAXERR_BADPARM	The time zone information in pServerInfo is invalid.

Remarks

To obtain the offset between the local bias and the server bias (server bias relative to the local), pass in zero for timeServer. The result is the relative bias in seconds.

See Also

RFaxLocalToServerTime()

RFaxGetServerTimeBias()

RFaxSetPercentPtr()

This function sets a pointer to a USHORT value that the client API will update with a completion percentage as it performs lengthy or multiple tasks.

C Prototype

```
void RFAXAPI RFaxSetPercentPtr(
 IN RFAXSERVERHANDLE hServer,
 IN USHORT *lpPercent
);
```

Arguments

hServer	Handle of the RightFax server.
lpPercent	Pointer to the percent that RightFax will fill with the percent complete when applicable.

Return Values

None.

Remarks

This function provides feedback to user interfaces. This function points the API to an application variable that the API will modify when executing an API call that takes a long time to complete. This mechanism is useful when the calling application is multi-threaded and is executing the RightFax API on a background worker thread. The function RFaxSetPercentCallback() may be more useful.

The following RightFax API calls provide completion feedback through the value to which the lpPercent argument points:

- RFaxLibDocFromFax()
- RFaxLoadFax2() (when loading fax pages)
- RFaxCombineFaxes()
- RFaxCombineLibDocs()
- RFaxGetGroupList()
- RFaxGetSigList()
- RFaxGetFaxList()
- RFaxGetFaxList2()
- RFaxGetLibDocList()
- RFaxGetLibDocList2()
- RFaxGetLibDocList3()
- RFaxGetLibDocList4()
- RFaxGetFormList()
- RFaxGetPrinterList()

- RFaxGetUserList()
- RFaxGetPhoneList()
- RFaxGetPhoneList2()

See Also

RFaxSetPercentCallback() below

RFaxSetPercentCallback()

This function sets a pointer to a callback function that the RightFax API will periodically call while executing a function that may take a long time to complete.

C Prototype

```
void RFAXAPI RFaxSetPercentCallback(
 IN RFAXSERVERHANDLE hServer,
 IN void (__cdecl *Callback)()
);
```

Arguments

hServer	Handle of the RightFax server.
Callback	Pointer to a function that the API will call back to when executing an API call that can take a long period of time. The function is called with a single argument, the percent of the operation that is complete.

Return Values

None.

Remarks

This function provides feedback to user interfaces. The function sets up a callback function that the API will periodically call when executing an API function that can take a long period of time. The callback function typically updates a status bar or status dialog box.

The following RightFax API calls provide completion feedback through this callback function:

- RFaxGetSigList()
- RFaxGetFaxList()
- RFaxGetFaxList2()
- RFaxGetLibDocList()
- RFaxGetLibDocList2()
- RFaxGetLibDocList3()
- RFaxGetLibDocList4()
- RFaxGetFormList()
- RFaxGetPrinterList()
- RFaxGetUserList()
- RFaxGetPhoneList()
- RFaxGetPhoneList2()

See Also

RFaxSetPercentPtr() on the previous page

RFaxSetPercentCallback2() below)

RFaxSetPercentCallback2()

This function sets a pointer to a callback function that the RightFax API will periodically call while executing a function that may take a long time to complete. The callback returns a boolean value indicating whether to stop the operation.

C Prototype

```
void RFAXAPI RFaxSetPercentCallback2(
 IN RFAXSERVERHANDLE hServer,
 IN bool(__cdecl *Callback)()
);
```

Arguments

hServer	Handle of the RightFax server.

Callback	If Callback return value is TRUE, the operation will be stopped; if the return value is FALSE, the operation will continue.

Return Values

None.

Remarks

This function provides feedback to user interfaces. The function sets up a callback function that the API will periodically call when executing an API function that can take a long period of time. The callback function returns TRUE or FALSE.

The following RightFax API calls provide completion feedback through this callback function:

- RFaxGetSigList()
- RFaxGetFaxList()
- RFaxGetFaxList2()
- RFaxGetLibDocList()
- RFaxGetLibDocList2()
- RFaxGetLibDocList3()
- RFaxGetLibDocList4()
- RFaxGetFormList()
- RFaxGetPrinterList()
- RFaxGetUserList()
- RFaxGetPhoneList()
- RFaxGetPhoneList2()

See Also

RFaxSetPercentCallback() on the previous page

RFaxSetPercentPtr() on page 301

RFaxSetThreadResolution()

The API can now detect concurrent usage of a single connection (RFAXSERVERHANDLE) by multiple threads. This function can be used to specify the method to handle these situations.

C Prototype

```
DWORD RFAXAPI RFaxSetThreadResolution(
 IN RFAXSERVERHANDLE hServer,
 IN ApiThreadResolution threadResolution
);
```

Arguments

hServer	Handle of the RightFax server.
threadResolution	A value of RFAXTHREADING_??? (ApiThreadResolution).

RFaxStringProtocol()

This function returns a string version of the specified RFax communication protocol value.

C Prototype

```
DWORD RFAXAPI RFaxStringProtocol(
 IN DWORD dwProtocol,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

dwProtocol	Specify the COMMPROTOCOL_??? value.
pszBuff	Returns a string version of the error value specified above.
usBuffLen	Specify the length of the buffer at pszBuff.

Return Values

Return	Result
RFAX_SUCCESS (0)	Success
RFAXERR_???	Failure

Remarks

This function gets a string version of an COMMPROTOCOL_??? value.

See Also

COMMPROTOCOL_??? on page 402

RFaxStringRFaxErr()

This function returns a string version of the specified RFax error value.

C Prototype

```
DWORD RFAXAPI RFaxStringRFaxErr(
 IN DWORD dwRFaxErr,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

dwRFaxErr	Specify the RFAXERR_??? error value.
pszBuff	Returns a string version of the error value specified above.
usBuffLen	Specify the length of the buffer at pszBuff.

Return Values

Returns the number of bytes copied into the destination buffer, pszBuff.

Remarks

This function gets a string version of an RFAXERR_??? value.

See Also

RFAXERR_??? on page 466

RFaxStringRFaxErr2()

This function returns a string version of the specified RFax error value from the specified server and converts errors on the client side if possible.

C Prototype

```
DWORD RFAXAPI RFaxStringRFaxErr2(
 IN RFAXSERVERHANDLE hServer
 IN DWORD dwRFaxErr,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

hServer	Handle of the server from which to get the error.
dwRFaxErr	Specify the RFAXERR_??? error value.
pszBuff	Returns a string version of the error value specified above.
usBuffLen	Specify the length of the buffer at pszBuff.

Return Values

Returns the number of bytes copied into the destination buffer, pszBuff.

Remarks

This function gets a string version of an RFAXERR_??? value.

See Also

RFAXERR_??? on page 466

RFaxStringRFaxErr3()

This function returns a string version of the specified RFax error value from the specified server and converts errors on the client side if possible.

C Prototype

```
DWORD RFAXAPI RFaxStringRFaxErr3(
 IN RFAXSERVERHANDLE hServer
 IN DWORD dwRFaxErr,
 IN BOOL fOutputUTF8,
 OUT RFAXSTR pszBuff,
 IN USHORT usBuffLen
);
```

Arguments

hServer	Handle of the server from which to get the error.
dwRFaxErr	Specify the RFAXERR_??? error value.
fOutputUTF8	
pszBuff	Returns a string version of the error value specified above.
usBuffLen	Specify the length of the buffer at pszBuff.

Return Values

Returns the number of bytes copied into the destination buffer, pszBuff.

Remarks

This function gets a string version of an RFAXERR_??? value.

See Also

RFAXERR_??? on page 466

RFaxValuesToCTime()

Converts a number from Gregorian time to ctime.

C Prototype

```
ULONG RFAXAPI RFaxValuesToCTime(
 IN short sYear,
 IN short sMonth,
 IN short sDay,
 IN short sHour,
 IN short sMin,
 IN short sSec
);
```

Arguments

sYear	Specify the year. Use the format YYYY.
sMonth	Specify the month. Use the format MM.
sDay	Specify the day. Use the format DD.
sHour	Specify the hour. Use the format HH.
sMin	Specify the minute. Use the format MM.
sSec	Specify the second. Use the format SS.

Return Values

Returns the ctime value corresponding to the specified Gregorian date.

Remarks

In RightFax, time is stored in ctime, or the number of seconds since 12:00 A.M. January 1, 1970. This function converts from Gregorian time to ctime.

See Also

RFaxCTimeToValues() on page 287

RFaxVersionCheck()

Checks whether the specified server matches the specified version criteria.

C Prototype

```
DWORD RFAXAPI RFaxVersionCheck(
 IN RFAXSERVERHANDLE hServer,
 IN ULONG ulRequiredVersion,
 IN OPTIONAL ULONG ulRequiredBuildDate
);
```

Arguments

hServer	Handle of the server to check.
ulRequiredVersion	Minimum version to check for, e.g., 0x0700 or 0x0600,
ulRequiredBuildDate	Minimum build date needed (0xYYYYMMDD, such as 0x19991022). Enter zero if the build date is not needed.

Return Values

Return	Result
"RFAX_SUCCESS	Success
RFAXERR_NOTSUPPORTED	Server does not meet specified criteria
RFAXERR_INVALID_HANDLE	Server not found

See Also

RFaxGetAPIVersion() on page 292

Appendix A: Structures and objects

Refer to the RFAPI.H file for exact C definition of these structures.

ARCHIVEREQUEST

The ARCHIVEREQUEST structure is used with calls to the RFaxGetArchiveEvent() function.

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.
ULONG	ulRequestType	See REQUEST_???.
FAXHANDLE	handle	Handle of the fax to archive.
SHORT	fDeleteFax	True = Delete fax after completed
CHAR	szUserID[22]	User ID.

AUDITLISTELEMENT0

The AUDITLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of fax.
ULONG	ulTimeConnected	Connection time.
CHAR	userid[22]	Owner's user ID.
CHAR	module[40]	The module being used.
CHAR	szClientName[16]	Name of the client machine.
ULONG	ulProcessId	Process ID.

AUTOPRINTREQ

The AUTOPRINTREQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
CHAR	printerid[10]	ID of the printer.
SHORT	startpage	Starting page.
SHORT	endpage	Ending page.
UCHAR	printcover	Flag indicating whether or not to print the cover page.
UCHAR	flags	See PRINTFLAG_???.
UCHAR	res	See PRINTRES_???.
UCHAR	size	See PRINTSIZE_???.
UCHAR	source	See PRINTSOURCE_???.
UCHAR	ucRetryCount	Number of attempts that have been made to process this print request.
FAXBOOL	fDeleteFax	
CHAR	d.szUserID[22]	Unused. Use hPrintingUser instead.
UCHAR	d.extra1.ucZeroByte	Must be set to zero.
UCHAR	d.extra1.ucOutputBin	See PRINTOUTPUT_???..
UCHAR	d.extra1.ucJobType	0 = Fax 1 = Confirmation sheet (auto-print sent)
UCHAR	d.extra1.ucPriority	0 = Low 1 = Medium 2 = High
UCHAR	d.extra1.aucAcctCode[8]	16 numeric digits max (stored as BCD, one nibble per digit).
USHORT	d.extra1.usSecurityCode	4 numeric digits max (stored as binary).
UCHAR	d.extra1.ucDuplex	See PRINTDUPLEX_???.
UCHAR	d.extra1.ucCopies	0 = Default 1 = Single copy 2 = 2 copies *n* = *n* copies, etc.
UCHAR	d.extra1.ucReserved[6]	Reserved. Initialize to zero.

BILLINFO_10

The BILLINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	Reserved for future use (set to 0).
CHAR	bill_info1[16]	Billing information field 1 name.
CHAR	bill_info2[16]	Billing information field 2 name.
CHAR	description[120]	Description for use of the billing information fields.
CHAR	reserved1[32]	Reserved for future use (set to 0).
CHAR	reserved2[48]	Reserved for future use (set to 0)

BILLLISTELEMENT0

The BILLLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
CHAR	bill_info1[16]	Billing information field 1.
CHAR	bill_info2[16]	Billing information field 2.
CHAR	description[62]	Shortened description for quick-loading purposes.

CCMAIL1REQ

The CCMAIL1REQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
FAXBOOL	fDeleteFax	Delete fax after routing.
SHORT	sMsgType	One of the MSGTYPE_??? constants.
CHAR	szMsgText[200]	Message text.

C Data Type	Member	Description
CHAR	szUserID[22]	Owner's user ID.
CHAR	szMailAddr[60]	Routing information.
AUTOPRINTREQ	print	AUTOPRINTREQ structure.
USHORT	usFileFormat	One of the IMAGEFMT_??? constants.
SHORT	sOCRError	If the fax was OCRed, this will indicated whether it was successful or not.

COMBINEINFO

The COMBINEINFO structure.

C Data Type	Member	Description
FAXHANDLE	hFax	Handle of fax.
USHORT	usFirstPage	First page to combine.
USHORT	usLastPage	Last page to combine. Do not set to zero (0), or an error will be returned.

COMPLETEREQUEST

The COMPLETEREQUEST structure is used with calls to the RFaxGetCompletionEvent() function.

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.
ULONG	ulRequestType	See REQUEST_???.
FAXHANDLE	handle	Handle of fax to for complete request event.
SHORT	fDeleteFax	True = Delete fax after completed.

COMPLETEREQUEST2

The COMPLETEREQUEST2 structure is used with calls to the RFaxGetCompletionEvent2() function.

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.

C Data Type	Member	Description
ULONG	ulRequestType	See REQUEST_???.
FAXHANDLE	handle	Handle of fax to for complete request event.
SHORT	fDeleteFax	True = Delete fax after completed.
CHAR	szUserID[22]	Owner's user ID.
USHORT	usNotifyChannel	Reserved. Initialize to zero.
FAXBOOL	fProdFax	Indicates this was a production (Integration Module) job.
short	sAttempt	Number of attempts that have been made to process this completion request.
CHAR	szReserved[460]	Reserved. Initialize to zero.

CONNECTIONATTRIBUTES

The CONNECTIONATTRIBUTES structure is used in RFaxSetConnectionAttributes.

C Data Type	Member	Description
BOOL	observeSharedServicesClientFailover	When set to true, the client will not fail over to another collective node if the Enable Shared Services client fail over is not selected on the General tab of the server control panel.
BOOL	neverUseSharedServices ClientFailOver	When set to true, the client never fails over to another collective node when using the API.
BOOL	detectIssuesOnPing	When set to true, issues that may affect the operation of the API are detected and appropriate errors returned. When set to false, clients wait for the issues to be resolved and no error is returned.
BOOL	failOnDatabaseDisconnect	When set to true, fails when the database server becomes disconnected from the data source.When set to false, continues trying the operation until the data source connection is re-established.
SHORT	simulatesSlowConnection	Reserved.
BOOL	showCommands	Shows API commands and result code on stdout.

CONNECTIONINFO_10

The CONNECTIONINFO_10 structure.

C Data Type	Member	Description
ULONG	ulFlags	See *CONNECTIONFLAGS_??? on page 403*.

C Data Type	Member	Description
TCHAR	szName[32]	Name of the connection.
ULONG	ulType	Connection type. See *CONNECTIONTYPE_??? on page 403*.
ULONG	ulTypeFlags	Flags for a connection type. Reserved for future use.
TCHAR	szTypeParam[64]	Parameter for a connection type.
TCHAR	szHost[256]	Host name or IP address.
USHORT	usPort	Network port number.
TCHAR	szUserName[64]	User identity on remote system.
TCHAR	szUserPass[64]	Password for szUserName.
CHAR	reserved[14]	Reserved for future use. Initialize to zero.

COVERSHEETINFO_10

The COVERSHEETINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle and primary key.
CHAR	szFilename[260]	File name.
CHAR	szShortFilename[14]	Short file name.
CHAR	szCoverSheetId[22]	User assigned cover sheet ID.
CHAR	szDescription[36]	User description of cover sheet.
BOOL	bIsSystemDefault	TRUE=This is the system default cover sheet. To set this value, call RFaxSetDefaultCoversheet().
TCHAR	szFormat[130]	FCS format (MIME type).
ULONG	ulFlags	See COVERSHEETFLAGS_???.
CHAR	szReserved[38]	

CVL_ATTACHITEM

The CVL_ATTACHITEM structure.

C Data Type	Member	Description
CVL_ATTACHTYPE_???	Type	See CVL_ATTACHTYPE_???.
ULONG	ulFlags	See CVL_ATTACHFLAG_???,
ULONG	ulLibDocUsage	One of the LIBDOC_USAGE_??? constants.
ULONG	ulSize	Attachment file size.
TCHAR	szFilename[MAX_PATH]	Attachment file name (on disk).
TCHAR	szDescription[MAX_PATH]	Attachment file descrilption.
TCHAR	szIdentifier[MAX_PATH]	LIBDOC identifier (if applicable)
FAXHANDLE	handle	Handle to a LIBDOC>

DELEGATEINFO_10

The DELEGATEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of delegate.
ULONG	ulFlags	Combination of DLGFLAGS_???.
CHAR	delegator_userid[22]	User ID of delegator.
CHAR	delegate_id[22]	User ID or group ID of delegate. If this is a group ID, DLGFLAGS_GROUP must be specified in ulFlags.
ULONG	ulReadPermissions	Read permissions to assign to delegate as defined in the DLGPERM_R_??? constant.
ULONG	ulWritePermissions	Write permissions to assign to delegate as defined in the DLGPERM_W_??? constant.
ULONG	ulDeletePermissions	Delete permissions to assign to delegate as defined in the DLGPERM_D_??? constant.
ULONG	ulMiscPermissions	Miscellaneous permissions to assign to delegate as defined in the DLGPERM_M_??? constant.
CHAR	reserved[950]	Reserved. Initialize to zero.

DELEGATELISTELEMENT0

The DELEGATELISTELEMENT0 structure is used with the RFaxGetDelegateList() function.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of delegate.
ULONG	ulFlags	Combination of DLGFLAGS_???.
CHAR	delegator_userid[22]	User ID of delegator.
CHAR	delegate_id[22]	User ID of delegate.
ULONG	ulReadPermissions	Read permissions to assign to delegate as defined in the DLGPERM_R_??? constant.
ULONG	ulWritePermissions	Write permissions to assign to delegate as defined in the DLGPERM_W_??? constant.
ULONG	ulDeletePermissions	Delete permissions to assign to delegate as defined in the DLGPERM_D_??? constant.
ULONG	ulMiscPermissions	Miscellaneous permissions to assign to delegate as defined in the DLGPERM_M_??? constant.
FAXHANDLE	hDelegator	The handle of the user that is delegating.
FAXHANDLE	hDelegatee	The handle of the user or group that is the delegate.
CHAR	reserved[22]	Reserved. Initialize to zero.

DELEGATETEMPLATELISTELEMENT0

The DELEGATETEMPLATELISTELEMENT0 structure is used with the RFaxGetDelegateTemplateList() function.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of delegate.
CHAR	userid[22]	User ID of the user who created the template.
CHAR	role_id[64]	Name of the template as provided by the user.
CHAR	role_reserved[192]	
ULONG	ulReadPermissions	Read specific permissions this template provides.
ULONG	ulWritePermissions	Write specific permissions this template provides.
ULONG	ulDeletePermissions	Delete specific permissions this template provides.
ULONG	ulMiscPermissions	Miscellaneous specific permissions this template provides.

DEVICEINFO_10

The DEVICEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of the device.
SHORT	deviceType [32]	Unique identifier of the device type. Read-only for an existing device.
CHAR	identifieraddressingMode [48]	Method the device uses to enforce a valid address to a destination.
CHAR	description[128]	Device description entered by and for display to the user.
ULONG	creationTime	When created.
ULONG	timeLastUsed	When last used.
SHORT	timesUsed	Number of times the device has been used.
CHAR	toName[60]	Default recipient name for addressing the fax.
CHAR	fromName[101]	Default sender's name for addressing the fax.
CHAR	userid[60]	Associated user ID. Some devices do not have an associated user ID, such as simple printers or MFPs that only send fax via email.
CHAR	networkAddress[16]	Address of the device.
SHORT	port	Printer port.
CHAR	printDriver[128]	Printer driver.
CHAR	parameters[128]	
SHORT	useSecureConnection	Whether to use a secure connection to the device.
SHORT	integrationStatus	Status of the integration.
CHAR	integrationType[24]	Type of integration.
TCHAR	paperType[24]	Type of printer paper.
TCHAR	defaultScanSize[24]	Paper size. If provided, the device will presume this size unless overridden. Primarily provided for devices that cannot detect paper size by default.
SHORT	userAuthMode	Method the device uses to authenticate users.
SHORT	isRegistered	If true, the device was registered.
CHAR	guestUserId[22]	The login name of the account that should be used to send faxes for this device if no other user is specified by either signing in, or some other method. Must map to a RightFax user ID.
CHAR	dataFlows[128]	

C Data Type	Member	Description
CHAR	uniqueID[38]	Unique ID that can hold a 36-character GUID or IPV6 address.
ULON	ulFlags	
UCHAR	reserved[32]	

DEVICEINFO_11

The DEVICEINFO_11 structure.

C Data Type	Member	Description
FAXHANDLE	handle	
CHAR	deviceType [32]	Uniquely identifies device type. Read-only for existing device.
CHAR	addressingMode [48]	How the device enforces valid addressing (destination)
CHAR	description[128]	Description.
ULONG	creationTime	When created.
ULONG	timeLastUsed	When last used.
SHORT	timesUsed	
CHAR	toName[60]	Default to name.
CHAR	fromName[101]	Default from name.
CHAR	userid[60]	Associated user ID. Some devices do not have an associated user ID, such as simple printers or MFPs that only send fax via email.
CHAR	networkAddress[16]	Address of device.
SHORT	port	Printer port.
CHAR	printDriver[128]	Printer driver.
CHAR	parameters[128]	
SHORT	useSecureConnection	Whether to use a secure connection to the device.
SHORT	integrationStatus	Status of the integration.
CHAR	integrationType[24]	Type of integration.

C Data Type	Member	Description
TCHAR	paperType[24]	Printer paper type.
TCHAR	defaultScanSize[24]	If provided, the device will presume this size unless over-ridden. Primarily provided for devices that can not detect paper-size by default.
SHORT	userAuthMode	How the device authenticates users.
SHORT	isRegistered	If true, the device was registered.
CHAR	guestUserId[22]	The login name of the account that should be used to send faxes for this device if no other user is specified by signing in or some other method. Must map to a RightFax user ID.
CHAR	dataFlows[128]	
CHAR	uniqueID[38]	Unique ID that can hold a 36 character GUID or an IPV6 address.
ULON	ulFlags	
CHAR	emailaddr[256]	
UCHAR	reserved[656]	

EXTENSIONINFO_10

The EXTENSIONINFO_10 structure.

C Data Type	Union Member Name	Member	Description
FAXHANDLE	N/A	handle	Handle of extension.
ULONG	N/A	ulFlags	Reserved. Initialize to zero.
ULONG	N/A	ulType	Dictates which member of the union is valid and the type of extension represented. See EXTENSIONTYPE_???.
CHAR	phonegrp	keys[PHNGRPEXT_KEYCOUNT][18]	Phone group keys. See PHNGRPEXT_???.
CHAR	phonegrp	reserved[14]	Reserved. Initialize to zero.
CHAR	userfolder	szFolderNames[1400]	Folder names separated by Nulls.
CHAR	SMSInfo	szSMSServiceID[22]	Service ID for general user notifications.
CHAR	SMSInfo	szSMSDestNumber[64]	SMS destination number.
CHAR	UserPagerNotify	szPagerID[64]	Deprecated. Pager ID.

C Data Type	Union Member Name	Member	Description
CHAR	UserPagerNotify	szServiceID[22]	Service ID for general user notifications.
ULONG	UserPagerNotify	ulTypes1	See USER_PAGERNOTIFY_???.
ULONG	UserPagerNotify	ulTypes2	See USER_PAGERNOTIFY_???.
CHAR	UserPagerNotify	szCSIDList[256]	Call subscriber ID.
CHAR	UserPagerNotify	szWorkflowNotificationServiceID1[22]	Service ID for workflow notifications.
CHAR	UserPagerNotify	szWorkflowNotificationServiceID2[22]	Service ID for workflow notifications.
CHAR	AdminPagerNotify	szPagerID[64]	Deprecated. Pager ID.
CHAR	AdminPagerNotify	szServiceID[22]	Service ID.
ULONG	AdminPagerNotify	ulTypes1	See ADMIN_PAGERNOTIFY_???.
ULONG	AdminPagerNotify	ulTypes2	See ADMIN_PAGERNOTIFY_???.
USHORT	AdminPagerNotify	usHeartBeatInterval	Interval for heartbeat notification in 1/60ths of a second.
ULONG	AdminPagerNotify	ulQueueFaxDepth	The number of faxes in the queue.
ULONG	AdminPagerNotify	ulQueuePageDepth	The number of pages in the queue.
USHORT	AdminPagerNotify	usDigitsExpected	Expected number of DID digits.

FAXFILTERINFO_10

The FAXFILTERINFO_10 structure for saving search criteria.

C Data Type	Member	Description
FAXHANDLE	handle	
FAXHANDLE	owner	
FAXFILTERFLAGS	ulFlags	Reserved for future use.
CHAR	filterName[40]	
BYTE	ucReserved[586]	

FAXINFO_10

The FAXINFO_10 structure contains a union. The valid member of the union is indicated by the ulType member. See EXTENSIONINFO_10.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for this fax record.
CHAR	owner_id[22]	User ID of fax owner.
ULONG	ffr_flags	Internal use. See FFRFLAGS_???.
USHORT	links	Number of links to this record (when 0, the image is deleted).
USHORT	oldjobid	Network print queue job ID.
USHORT	faxtype	See FAXBODYTYPE_???.
CHAR	faxfilename[14]	Body image name in the RightFax\Image folder.
CHAR	inputfilename[14]	Source file name in the RightFax\Outgoing folder.
USHORT	numpages	Number of images associated with record (not including cover).
SHORT	papernum	Index of paper type (if used).
SHORT	printjobtype	Type of data in print job. See PRINTJOBTYPE_???.
SHORT	finemode	Fine mode fax, True or False.
CHAR	szBFTFile[14]	File name of a binary file attachment that resides in the RightFax\BFT folder.
ULONG	ulBFTBytes	Length of binary file.
CHAR	szOrigBFTName [14]	File name of original BFT file.
ULONG	ulImageBytes	Number of bytes consumed by body image(s).
LONG	jobid	
BYTE	body_ocr_status	Storage for status while fax body is being OCRed. See *OCRSTAT_???* on page 457.
BYTE	body_ocr_error_ code	Storage for error code from fax body being OCRed.
CHAR	ffr_reserved[6]	Reserved.
ULONG	fr_flags	Fax record flags. See FAXFLAG_???.
CHAR	to_faxnum[32]	Destination fax number.
CHAR	to_contactnum[32]	Destination voice number.
CHAR	to_name[60]	Destination name.

C Data Type	Member	Description
CHAR	to_company[60]	Destination company name.
CHAR	to_citystate[60]	Destination location name.
CHAR	from_name[60]	Source name.
CHAR	from_phonenum [32]	Source voice number.
CHAR	billinfo1[16]	Billing code 1.
CHAR	billinfo2[16]	Billing code 2.
CHAR	faxdidnum[32]	Source fax number.
CHAR	operatornum[32]	Source generic voice number.
CHAR	generalfaxnum[32]	Source generic fax number.
SHORT	call_back	Request call back, True or False.
LONG	fax_datetime	Time fax created, in ctime type (GMT0).
ULONG	send_time	Time elapsed in sending.
USHORT	fax_status	Fax status flags. See FAXSTAT_???.
USHORT	fax_error_code	Fax error code flags. See FAXERROR_???.
USHORT	fax_disposition	Fax disposition.
USHORT	fax_termstat	See TERMSTAT_???.
CHAR	remoteid[22]	ID of sending fax machine.
FAXHANDLE	faxfile_dba	Database address of fax file record(s) belonging to this fax.
FAXHANDLE	faxnote_dba	Database address of fax note record belonging to this fax.
CHAR	fcsfile[14]	File name of generated cover sheet.
CHAR	fcstext[14]	File name of source for cover sheet. See fcstype.
USHORT	fax_id	ID provided by hardware when fax job is scheduled.
USHORT	usNotifyChannel;	Reserved. Do not touch. Initialize to zero.
CHAR	phone_user_id[22]	Reserved. Do not touch. Initialize to zero.

C Data Type	Member	Description
CHAR	phone_entry_id[18]	Reserved. Do not touch. Initialize to zero.
LONG	faxsend_datetime	Time the fax should be sent (delay send), in ctime type (GMT0). This can be set to zero (0) when there is no delay set for the fax.
CHAR	unique_id[16]	Unique user or system provided ID for fax job.
USHORT	fcstype	Set bit 5 (10 hex, 16 decimal) when changing cover sheet via fcstext field.
UCHAR	emailtype	One of the FAXEMAILTYPE_??? constants (used only if FAXFLAG_GATEWAYFAX is set in fr_flags).
UCHAR	ucPriority	Fax job priority level. See PRIORITY_???.
USHORT	usSendChannel	Hardware fax channel to use for send.
USHORT	usCustom1	User-defined field.
USHORT	usBoardSrv	Fax board server to use for send. Reserved for future use.
USHORT	oldfax_status	Storage for status while fax is being printed.
USHORT	oldfax_error_code	Storage for error code while fax is being printed.
USHORT	usPagesInFront	Pages ahead of this fax to be sent.
UCHAR	ucMaxTries	Number of times this fax should be re-tried if sending fails for any reason.
UCHAR	ucTryInterval	Time interval (in minutes) between fax re-tries.
USHORT	usFolder	Folder number assigned to the fax. Default is 0, the Main folder.
USHORT	usSendNotify	
CHAR	szSecureCSID[22]	If specified then the remote machine CSID must match this string.
CHAR	szFromNameCont [41]	Extended portion of return address.
CHAR	ucAutoFwdCount	Count of auto forward hops this fax has made.
LONG	histchg_datetime	Touched every time a history record is added to fax.
ULONG	tuifax_dba	Reserved. Do not touch. Initialize to zeroes.
ULONG	fr_flags2	Fax record flags. See FAXFLAG2_???.
FAXHANDLE	faxcclist_dba	Reserved. Do not touch. Initialize to zeroes.
CHAR	cmddata[14]	Reserved. Do not touch. Initialize to zeroes.

FAXINFO_11

The FAXINFO_11 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for this fax record (these are re-used).
CHAR	owner_id[22]	User ID of fax owner.
ULONG	ffr_flags	Internal use. See FFRFLAGS_???.
USHORT	links	Number of links to this record (when 0, image is deleted).
USHORT	oldjobid	Network print queue job ID number (useful?)
USHORT	faxtype	See FAXBODYTYPE_???.
CHAR	faxfilename[14]	Body image name in the RightFax\Image folder.
CHAR	reserved1[14]	Source file name in OUTGOING dir
USHORT	numpages	Number of images associated with record (not including cover).
SHORT	papernum	Index of paper type (if used).
SHORT	printjobtype	Type of data in print job. See PRINTJOBTYPE_???.
SHORT	finemode	Fine mode, True or False.
CHAR	szBFTFile[14]	Binary file name in the RightFax\BFT folder.
ULONG	ulBFTBytes	Length of binary file.
CHAR	szOrigBFTName[14]	File name of original BFT file.
ULONG	ulImageBytes	Number of bytes consumed by body image(s).
LONG	jobid	
BYTE	body_ocr_status	Storage for status while fax body is being OCRed. See OCRSTAT_???.
BYTE	body_ocr_error_code	Storage for error code from fax body being OCRed.
CHAR	ffr_reserved[6]	Reserved.
ULONG	fr_flags	Fax record flags. See FAXFLAG_???.
CHAR	to_faxnum[32]	Destination fax number.
CHAR	to_contactnum[32]	Destination voice number.

C Data Type	Member	Description
CHAR	to_name[60]	Destination name.
CHAR	to_company[60]	Destination company name.
CHAR	to_citystate[60]	Destination location name.
CHAR	from_name[60]	Source name.
CHAR	from_phonenum[32]	Source voice number.
CHAR	billinfo1[16]	Billing code 1.
CHAR	billinfo2[16]	Billing code 2.
CHAR	faxdidnum[32]	Source fax number.
CHAR	operatornum[32]	Source generic voice number.
CHAR	generalfaxnum[32]	Source generic fax number.
SHORT	call_back	Request callback, True or False.
LONG	fax_datetime	Time fax created, in ctime type (GMT0).
ULONG	send_time	Time elapsed in sending.
USHORT	fax_status	Fax status flags. See FAXSTAT_???.
USHORT	fax_error_code	Fax error code flags. See FAXERROR_???.
USHORT	fax_disposition	Fax disposition.
USHORT	fax_termstat	Fax terminating status. See TERMSTAT_???.
CHAR	remoteid[22]	ID of sending fax machine.
FAXHANDLE	faxfile_dba	Database address of fax file record(s) belonging to this fax.
FAXHANDLE	faxnote_dba	Database address of fax note record belonging to this fax.
CHAR	fcsfile[14]	File name of generated cover sheet.
CHAR	fcstext[14]	File name of source for cover sheet. See fcstype.
USHORT	usNotifyChannel;	Reserved. Do not touch. Initialize to zero.
CHAR	phone_user_id[22]	Reserved. Do not touch. Initialize to zero.
CHAR	phone_entry_id[18]	Reserved. Do not touch. Initialize to zero.

C Data Type	Member	Description
LONG	faxsend_datetime	Time the fax should be sent (delay send), in ctime type (GMT0). This can be set to zero (0) when there is no delay set for the fax.
CHAR	unique_id[16]	Unique user or system provided ID for fax job.
USHORT	fcstype	Set bit 5 (10 hex, 16 decimal) when changing cover sheet via fcstext field.
UCHAR	emailtype	One of the FAXEMAILTYPE_??? constants (used only if FAXFLAG_GATEWAYFAX is set in fr_ flags).
UCHAR	ucPriority	Fax job priority level. See PRIORITY_???.
USHORT	usSendChannel	Hardware fax channel to use for send.
USHORT	usCustom1	User-defined field.
USHORT	usBoardSrv	Fax board server to use for send. Reserved for future use.
USHORT	oldfax_status	Storage for status while fax is being printed.
USHORT	oldfax_error_code	Storage for error code while fax is being printed.
USHORT	usPagesInFront	Pages ahead of this fax to be sent.
UCHAR	ucMaxTries	Number of times this fax should be re-tried if sending fails for any reason.
UCHAR	ucTryInterval	Time interval (in minutes) between fax re-tries.
USHORT	usFolder	Folder number assigned to the fax. Default is 0, the Main folder.
USHORT	usSendNotify	
CHAR	szSecureCSID[22]	If specified then the remote machine CSID must match this string.
CHAR	szFromNameCont[41]	Extended portion of return address.
CHAR	ucAutoFwdCount	Count of auto forward hops this fax has made.
LONG	histchg_datetime	Touched every time a history record is added to fax.
ULONG	tuifax_dba	Reserved. Do not touch. Initialize to zeroes.
ULONG	fr_flags2	Fax record flags. See FAXFLAG2_???.
FAXHANDLE	faxcclist_dba	Reserved. Do not touch. Initialize to zeroes.
CHAR	cmddata[14]	Reserved. Do not touch. Initialize to zeroes.
CHAR	szAltFax[32]	Alternate fax number to send to on even attempts, e.g., 2nd, 4th, etc.

C Data Type	Member	Description
ULONG	ulFCSBytes	Number of bytes consumed by cover sheet image.
CHAR	szComment[70]	Comment field for user comments.
RFAXTIME	lExpiresIn	Length of time in ctime type (GMT0) since the moment the fax was created that in which the fax will expire (can no longer be sent).
UCHAR	szMsgFromTransport[32]	Message from the communication transport layer.
UCHAR	ucRecipNotifyType	
UCHAR	ucPreferredContentType	Preferred content type. See CONTENTTYPE_???
CHAR	szRecipNotifyAddress	
CHAR	szDelegate_id[22]	ID of authorized delegate sending or forwarding the fax.
CHAR	szFaxGUID[64]	Unique Identifier assigned by the DocTransport.
LONG	fax_completion_datetime	Show Completion Time.
CHAR	szJobId[32]	RF Connect job id.
CHAR	szANI[32]	ANI
CHAR	szNotifyQueue[64]	Fully qualified notification queue name (e.g., computer/$private/myqueue). Use %QueueServer% for substitution with the name of the queue server.
ULONG	ulNotifyQueueFlags	See NQFLAGS_???.
CHAR	inputfilename[14]	Source file name in the RightFax\Outgoing folder.
FAXHANDLE	hWorkflow	The workflow, if any, this fax is currently a member of.
FAXHANDLE	hParentFax	When created from another fax, as in the case with a forward or broadcast, the handle of the originating fax. NULL otherwise.
BYTE	print_status	Storage for status while fax is being printed. See *PRINTSTAT_???* on page 463.
BYTE	fcs_ocr_status	Storage for status while FCS is being OCRed. See *OCRSTAT_???* on page 457.
BYTE	print_error_code	Storage for error code while fax is being printed.
BYTE	fcs_ocr_error_code	Storage for error code when FCS is OCRed.
CHAR	szCommentCont[82]	150 total usable comment characters 70-null+82-null=150.
CHAR	szJobIdCont[32]	

C Data Type	Member	Description
FAXHANDLE	hLastOwner	Last user to own the fax. Also set in the case of a forwarded fax.
CHAR	fr_reserved2[482]	Reserved for future use. Initialize to 0 when creating a new fax.

FAXLISTELEMENT0

The FAXLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See FAXFLAG_???.
RFAXTIME	fax_datetime	'C' time type (GMT0).
CHAR	to_name[18]	To name for fax list element.
CHAR	to_number[16]	To number.
SHORT	numpages	Total number of pages.
USHORT	fax_status	See FAXSTAT_???.
USHORT	fax_error_code	See FAXERROR_???.
time_t	faxsend_datetime	Time fax should be sent (delay send), in ctime type (GMT0). This can be set to zero (0) when there is no delay set for the fax.
USHORT	otherstuff	Bit 0 = Fine mode? Bit 1 = Created from PCL file?
USHORT	displayflags	Bit 0 = Has cover sheet? Bit 1 = Has body? Bit 2 = Has BFT attachment? Bit 4 = BFT only, no images?
USHORT	numtrxrecs	Total number of transaction records.
ULONG	ulBFTBytes	Size of BFT file in bytes.
CHAR	szOrigBFTName[14]	Name of BFT file.
USHORT	usPagesInFront	Total number of pages in front of job.

C Data Type	Member	Description
CHAR	billinfo1[16]	Billing information field 1.
CHAR	billinfo2[16]	Billing information field 2.
CHAR	unique_id[16]	Unique ID of the fax.
USHORT	usFolderIndex	Index of the folder where the fax belongs.
LONG	histchg_datetime	ctime indication when last history element was added.
CHAR	userid[22]	User ID of the owner of the fax.
USHORT	usStatusType	One of the STATUS_TYPE_??? constants; set by FAXSTRINGSTAT_SETSTATUSTYPE
DWORD	dwStatusID	One of the IDS_RFAPI_FAXSTAT_??? constants; set by FAXSTRINGSTAT_SETSTATUSID.
CHAR	reserved[92]	Reserved. Initialize to zero.

FAXLISTELEMENT1

The FAXLISTELEMENT1 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See FAXFLAG_???.
LONG	fax_datetime	ctime type (GMT0).
CHAR	to_name[60]	To name for fax list element.
CHAR	to_number[32]	To number.
SHORT	numpages	Total number of pages.
USHORT	fax_status	See FAXSTAT_???.
USHORT	fax_error_code	See FAXERROR_???.
LONG	faxsend_datetime	Time fax should be sent (delay send), in ctime type (GMT0). This can be set to zero (0) when there is no delay set for the fax.
USHORT	otherstuff	Bit 0 = Fine mode? Bit 1 = Created from PCL file?

C Data Type	Member	Description
USHORT	displayflags	Bit 0 = Has cover sheet? Bit 1 = Has body? Bit 2 = Has BFT attachment? Bit 4 = BFT only, no images?
USHORT	numtrxrecs	Total number of transaction records.
ULONG	ulBFTBytes	Size of BFT file in bytes.
CHAR	szOrigBFTName[14]	Name of BFT file.
USHORT	usPagesInFront	Total number of pages in front of job.
CHAR	billinfo1[16]	Billing information field 1.
CHAR	billinfo2[16]	Billing information field 2.
CHAR	unique_id[16]	Unique ID of the fax.
USHORT	usFolderIndex	Index of the folder where the fax belongs.
LONG	histchg_datetime	ctime indication when the last history element was added.
CHAR	userid[22]	User ID of the owner of the fax.
USHORT	usStatusType	One of the STATUS_TYPE_??? constants; set by FAXSTRINGSTAT_SETSTATUSTYPE
DWORD	dwStatusID	One of the IDS_RFAPI_FAXSTAT_??? constants; set by FAXSTRINGSTAT_SETSTATUSID.
CHAR	reserved[34]	Reserved for future use.

FAXLISTELEMENT2

The FAXLISTELEMENT2 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See FAXFLAG_???.
LONG	fax_datetime	ctime type (GMT0).
CHAR	to_name[60]	To name for fax list element.
CHAR	to_number[32]	To number.

C Data Type	Member	Description
SHORT	numpages	Total number of pages.
USHORT	fax_status	See FAXSTAT_???.
USHORT	fax_error_code	See FAXERROR_???.
LONG	faxsend_datetime	Time fax should be sent (delay send), in ctime type (GMT0). This can be set to zero (0) when there is no delay set for the fax.
USHORT	otherstuff	Bit 0 = Fine mode? Bit 1 = Created from PCL file?
USHORT	displayflags	Bit 0 = Has cover sheet? Bit 1 = Has body? Bit 2 = Has BFT attachment? Bit 4 = BFT only, no images?
USHORT	numtrxrecs	Total number of transaction records.
ULONG	ulBFTBytes	Size of BFT file in bytes.
CHAR	szOrigBFTName[14]	Name of BFT file.
USHORT	usPagesInFront	Total number of pages in front of job.
CHAR	billinfo1[16]	Billing information field 1.
CHAR	billinfo2[16]	Billing information field 2.
CHAR	unique_id[16]	Unique ID of the fax.
USHORT	usFolderIndex	Index of the folder where the fax belongs.
LONG	histchg_datetime	ctime indication when the last history element was added.
CHAR	userid[22]	User ID of the owner of the fax.
USHORT	usStatusType	One of the STATUS_TYPE_??? constants.
DWORD	dwStatusID	Reserved. Initialize to zero.
CHAR	szComment[70]	User comment for the fax.
ULONG	flags2	Fax record flags. See FAXFLAG2_???. Equivalent to FAXINFO_##.fr_flags2.
ULONG	ffr_flags	Fax record flags. See FFRFLAGS_???. Equivalent to FAXINFO_##.ffr_flags.

C Data Type	Member	Description
CHAR	to_company[60]	To company.
CHAR	szMsgFromTransport[32]	Message from transport (for DocPlus but usable by any transport).
CHAR	reserved[4]	Reserved. Initialize to zero.

FAXLISTELEMENT3

The FAXLISTELEMENT3 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See FAXFLAG_???.
LONG	fax_datetime	ctime type (GMT0).
CHAR	to_name[60]	To name for fax list element.
CHAR	to_number[32]	To number.
SHORT	numpages	Total number of pages.
USHORT	fax_status	See FAXSTAT_???.
USHORT	fax_error_code	See FAXERROR_???.
LONG	faxsend_datetime	Time fax should be sent (delay send), in ctime type (GMT0). This can be set to zero (0) when there is no delay set for the fax.
USHORT	otherstuff	Bit 0 = Fine mode? Bit 1 = Created from PCL file?
USHORT	displayflags	Bit 0 = Has cover sheet? Bit 1 = Has body? Bit 2 = Has BFT attachment? Bit 4 = BFT only, no images?
USHORT	numtrxrecs	Total number of transaction records.
ULONG	ulBFTBytes	Size of BFT file in bytes.
CHAR	szOrigBFTName[14]	Name of BFT file.

C Data Type	Member	Description
USHORT	usPagesInFront	Total number of pages in front of job.
CHAR	billinfo1[16]	Billing information field 1.
CHAR	billinfo2[16]	Billing information field 2.
CHAR	unique_id[16]	Unique ID of the fax.
USHORT	usFolderIndex	Index of the folder where the fax belongs.
LONG	histchg_datetime	ctime indication when the last history element was added.
CHAR	userid[22]	User ID of the owner of the fax.
USHORT	usStatusType	One of the STATUS_TYPE_??? constants.
DWORD	dwStatusID	Reserved. Initialize to zero.
CHAR	szComment[70]	User comment for the fax.
ULONG	flags2	Fax record flags. See FAXFLAG2_???. Equivalent to FAXINFO_##.fr_flags2.
ULONG	ffr_flags	Fax record flags. See FFRFLAGS_???. Equivalent to FAXINFO_##.ffr_flags.
CHAR	to_company[60]	To company.
CHAR	szMsgFromTransport[32]	Message from transport (for DocPlus but usable by any transport).
CHAR	szFaxDIDNum[32]	Recipient's DID number.
LONG	fax_completion_datetime	The completion time expressed as time_t. 0 if fax is not yet complete.
CHAR	szAltFax[32]	For cases where the destination address is longer than 32 (ie:email address)
CHAR	szDelegateId[22]	For cases where the fax was originally sent by a delegate.
CHAR	szJobId[32]	RF Connect Job ID.
CHAR	szANI[32]	ANI.
CHAR	reserved[10]	Reserved. Initialize to zero.

FAXLISTELEMENT4

The FAXLISTELEMENT4 structure. FAXLISTELEMENT4 is derived from *FAXLISTELEMENT3* on the previous page and, as such, includes everything in FAXLISTELEMENT3 plus the following additional fields.

C Data Type	Member	Description
CHAR	szLockingUserId[22]	Locking user, if locked.
FAXHANDLE	hWorkflow	Worklow that the fax is in, if any.
BYTE	print_status	Stores the status while the fax is printed. See *PRINTSTAT_???* on page 463.
BYTE	print_error_code	Stores the error code while the fax is printed.
BYTE	body_ocr_status	Stores the status while the fax body is OCRed. See *OCRSTAT_???* on page 457.
BYTE	body_ocr_error_code	Stores the error code while the fax body is OCRed.
BYTE	fcs_ocr_status	Stores the status while the fax FCS is OCRed. See *OCRSTAT_???* on page 457.
BYTE	fcs_ocr_error_code	Stores the error code while the fax FCS is OCRed.
CHAR	lastOwnerId[22]	Last owner, if there was one.
CHAR	reserved2[30]	Reserved.

FAXLISTELEMENT5

The FAXLISTELEMENT5 structure. FAXLISTELEMENT5 is derived from *FAXLISTELEMENT3* on page 330 and, as such, includes everything in FAXLISTELEMENT4 plus the following additional fields.

C Data Type	Member	Description
CHAR	szCommentCont[82]	150 total usable comment characters 70-null+82-null=150.
CHAR	szJobIdCont[32]	
CHAR	reserved3[242]	

FILELISTELEMENT0

The FILELISTELEMENT0 structure is used in the *lpLH argument of the RFaxGetFileList2() function.

C Data Type	Member	Description
ULONG	ulFileSize	Size of the file.
ULONG	ulFileAttr	Correlates to the Windows WIN32_FIND_DATA.dwFileAttributes field.
UCHAR	szFilename[256]	Name of the file.

C Data Type	Member	Description
UCHAR	szShortFilename[14]	Returns the short file name in calls to RFaxGetFileList2().
FAXHANDLE	Handle	For future use when FCS files have handles.
LONG	tLastModified	
CHAR	szReserved[226]	Reserved. Initialize to zero.

FILERTE1REQ

The FILERTE1REQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
FAXBOOL	fDeleteFax	Delete fax after routing.
USHORT	usFileFormat	
CHAR	szFilePath[72]	
AUTOPRINTREQ	print	
CHAR	szUserID[22]	
SHORT	sOCRError	

FAXNUMBERLISTELEMENT

The FAXNUMBERLISTELEMENT structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
TCHAR	faxNumber[80]	Fax number.
TCHAR	ownerId[22]	The ID of the user or group with the fax number as routing code.
BYTE	reserved[132]	

FOLDERLISTELEMENT0

The FOLDERLISTELEMENT0 structure is used with the RFaxGetFolderList2() function.

C Data Type	Member	Description
FAXHANDLE	hOwner	Database handle for particular list element.
USHORT	usID	ID of this folder.
USHORT	usParentID	ID of parent folder.
CHAR	szName[256]	Name of the folder.

FOLDERLISTELEMENT1

The FOLDERLISTELEMENT1 structure is used with the RFaxGetFolderList2() and *RFaxGetFolderList3()* on page 239 functions.

C Data Type	Member	Description
FAXHANDLE	hOwner	Database handle for particular list element.
USHORT	usID	ID of this folder.
USHORT	usParentID	ID of parent folder.
CHAR	szName[240]	Name of the folder.
ULONG	ulType	
ULONG	ulFlags	See *FOLDERFLAGS_???* on page 427.
RFAXUSN	usn	

FORMINFO_10

The FORMINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle of form.
ULONG	ft_flags	See FORMFLAG_???.
SHORT	ft_num	Form index number.
CHAR	ft_code[16]	ID of the form.
CHAR	ft_name[40]	Descriptive name of the form.
CHAR	ft_filename[20]	File name associated to the form.

C Data Type	Member	Description
SHORT	ft_startpage	Starting page.
SHORT	ft_numpages	Total pages.
SHORT	ft_nexttype	Index number of the next form.
CHAR	ft_securityid[20]	This field can contain a user ID or group ID.

FORMLISTELEMENT0

The FORMLISTELEMENT0 structure is used with the RFaxGetFormList() function.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See FORMFLAG_???.
SHORT	num	Form index number.
CHAR	code[16]	ID of the form.
CHAR	name[40]	Descriptive name of the form.
CHAR	filename[20]	File name associated to the form.
SHORT	startpage	Starting page.
SHORT	numpages	Total pages.
SHORT	nexttype	Index number of the next form.
CHAR	securityid[20]	This field can contain a user ID or group ID.

GENMAIL1REQ

The GENMAIL1REQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
FAXBOOL	fDeleteFax	Delete fax after routing.
SHORT	sMsgType	One of the MSGTYPE_??? constants.
CHAR	szMsgText[200]	Message text.

C Data Type	Member	Description
CHAR	szUserID[22]	Owner's user ID.
CHAR	szMailAddr[60]	Routing information.
UCHAR	ucGenMailType	One of the GENMAILTYPE_??? constants.
UCHAR	ucEmailRouteForm	The email routing form to use.
UCHAR	ucArchiveFax	TRUE=Queue fax for archiving.
UCHAR	ucIncludeThumbnail	TRUE=Include thumbnail.
UCHAR	ucThumbnailFileType	Thumbnail file type: 0=TIF 1=PDF
UCHAR	ucThumbnailPages	Thumbnail pages to include: 0=First only 1=All
UCHAR	ucIncludeThumbnailRecv	Deprecated.
UCHAR	ucThumbnailFileTypeRecv	Deprecated.
UCHAR	ucThumbnailPagesRecv	Deprecated.
UCHAR	ucIncludeWebLink	Include the web link formerly specified globally in the CPL that is now specified at the group or user levels.
CHAR	reserved2[3]	Reserved. Initialize to zero.
AUTOPRINTREQ	print	AUTOPRINTREQ structure.
USHORT	usFileFormat	One of the IMAGEFMT_??? constants.
SHORT	sOCRError	If the fax was OCRed, this will indicated whether it was successful or not.
CHAR	szMailAddrCont[52]	If non-blank, then append to szMailAddr. When GENMAILTYPE_SMS, this is the pager device ID.

GENMAIL2REQ

The GENMAIL2REQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
FAXBOOL	fDeleteFax	Delete fax after routing.
SHORT	sMsgType	One of the MSGTYPE_??? constants.
CHAR	reserved1[272]	Reserved.
UCHAR	ucGenMailType	One of the GENMAILTYPE_??? constants.
UCHAR	ucReserved3	Reserved.
CHAR	reserved2[108]	Reserved.

GENMAILPREVIEWREQ

The GENMAILPREVIEWREQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
FAXBOOL	fDeleteFax	Delete fax after routing.
CHAR	szUserID[22]	User ID
CHAR	szMailAddr[100]	Mail Address.
CHAR	szToName[60]	To Name.
CHAR	szToFaxNum[32]	Destination.
UCHAR	ucEmailRouteForm	Reserved. For internal use only.
USHORT	usFileFormat	One of the USER_ROUTEFMT_??? constants.
UCHAR	ucGenMailType	One of the GENMAILTYPE_??? constants.

GETNEWFAXCOUNT

The GETNEWFAXCOUNT structure.

C Data Type	Member	Description
ULONG	ulTotalFaxCount	Total number of faxes.
ULONG	ulTotalPageCount	Total number of pages.
ULONG	ulRecvFaxCount	Received faxes.

C Data Type	Member	Description
ULONG	ulRecvPageCount	Received pages.
ULONG	ulNewFaxCount	New faxes.
ULONG	ulNewPageCount	New pages.
ULONG	ulSentFaxCount	Sent faxes.
ULONG	ulSentPageCount	Sent pages.
ULONG	ulInProcessFaxCount	In-process faxes.
ULONG	ulInProcessPageCount	In-process pages.
ULONG	ulFailedFaxCount	Failed faxes.
ULONG	ulFailedPageCount	Failed pages.
ULONG	ulSuccessFaxCount	Successful faxes.
ULONG	ulSuccessPageCount	Successful pages.

GETPAGEDFAXLISTPARAMETERS

The GETPAGEDFAXLISTPARAMETERS structure provides fax retrieval parameters for *RFaxGetPagedFaxList()* on page 87 and *RFaxGetPagedFaxList2()* on page 88.

C Data Type	Member	Description
USHORT	usInfoLevel	Must be 0, 1, 2, or 3.
RFAXCSTR	pszUserID	User ID of owner of the fax. Superseded by RFAXFAXFILTER.userId when pfilter is specified.
DWORD	dwOptions	See *FAXGETLIST_???* on page 421 flags or 0 for none.
USHORT	usFolder	Folder to load. 0 = all. See *FAXFOLDER_???* on page 421. Superseded by RFAXFAXFILTER.iFolderIndex when pfilter is specified.
USHORT	usFilter	Filter to use in loading. 0 = none. See *FAXFILTER_???* on page 417. Superseded by RFAXFAXFILTER.usFilter when pfilter is specified.
RFAXFAXFILTER	pfilter	Optional fax filtration options or NULL. See *RFAXFAXFILTER* on page 360.
UCHAR	ucReserved[128]	Reserved for future use.

GETPAGEDLISTPARAMETERS

The GETPAGEDLISTPARAMETERS structure.

C Data Type	Member	Description
USHORT	usOrder	Order in which to load the paged records. See ???ORDER_??? functions.
BOOL	fAscending	TRUE for ascending; FALSE for descending.
LONG	lPageIndex	0-based index of page to return.
USHORT	usPageSize	Number of faxes per page

GROUPINFO_10

The GROUPINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular element.
USHORT	usMembers	Total number of members in the group.
ULONG	flags	See GRPFLAG_???.
CHAR	groupid[22]	ID of the group.
CHAR	admin[22]	Deprecated. Use ahAdmins instead.
CHAR	alt_admin	
CHAR	group_fcs[14]	Fax cover sheet file for the group.
USHORT	notify_type	See NOTIFY_TYPE_???.
USHORT	usRecvFaxAge	Viewed or printed received faxes.
USHORT	usSentFaxAge	Sent successfully.
USHORT	usDelFaxAge	Days to keep deleted fax records.
ULONG	ulGroupRoutingCode	Should be ULONG_MAX if SFD is disabled.
USHORT	usSFDType	See GRPSFDFLAG_???.
USHORT	usSFDFlags	Bit 0 = SFD enabled? Bit 1 = Requires login?

C Data Type	Member	Description
ULONG	ulChangeTag	Last time the group membership or group data changed.
USHORT	usMaxConcurrentPages	Total number of pages group can send.
USHORT	usMaxConcurrentFaxes	Total number of faxes group can send.
USHORT	usConcurrentDelayTime	Formatted as HHMM.
USHORT	usConcurrentStartTime	Formatted as HHMM.
USHORT	usConcurrentEndTime	Formatted as HHMM.
USHORT	usNewFaxAge	Unviewed and unprinted received faxes.
USHORT	usFailedFaxAge	Abandoned outbound faxes.
USHORT	usINCSentFaxAge	Info-not-complete outbound faxes.
ULONG	ulFCSDlgCtrl1	Turns off fields in client interface. See GRP_FCSDLGCTRL1_???.
ULONG	ulFCSDlgCtrl2	Turns off fields in client interface. See GRP_FCSDLGCTRL2_???.
ULONG	ulMembers	The number of members in the group.
USHORT	usHotLinkFaxAge	Hot link faxes/documents.
CHAR	szGroupRoutingCode[22]	
ULONG	ulReserved2	
CHAR	reserved2[244]	
USHORT	usEndErrorFaxAge	Specifies fax aging for completed error (red status).
USHORT	usEndProbFaxAge	Specifies fax aging for permanent problem (permanent yellow status).
FAXHANDLE	ahAdmins[10]	Array of administrator handles replacing admin and alt_admin.
UCHAR	ucNotifyAttachmentType	Format of attachment included with notifications. Use GRP_ATTACHTYPE_???. GRPFLAG_NOTIFYINCLUDEFAX bit flag must be specified in GROUPINFO_10.flags.
UCHAR	ucNotifyAttachmentPages	Pages included with notifications. Use GRP_ATTACHPAGES_???. Requires GROUPINFO_10.flags and GRPFLAG_NOTIFYINCLUDEFAX.
FAXHANDLE	hFaxNumber	Optional reference to a fax number (via FAXNUMBERLISTELEMENT.handle) corresponding to this group's szGroupRoutingCode.
USHORT	usEDCArchiveFlags	Specify the EDC module to use when archiving by group.

C Data Type	Member	Description
USHORT	usMaxSendPages	Specify the maximum number of pages that members are allowed to send in one fax.
ULONG	ulDefaultOptions	Holds bitwise options that need to be available across clients. See DEFAULT_OPTS_???.
FAXHANDLE	hLibDoc	Specify the library document that must be at the beginning of all fax bodies.
INT	iRetentionDays	Group retention days for purging archived faxes.
FAXHANDLE	hGatewayService	Preferred e-mail gateway.
CHAR	reserved[6]	Reserved. Initialize to zero.

HISTLISTELEMENT0

The HISTLISTELEMENT0 structure contains a union. The datatype held in the union is specified by the d.physical.tr_rectype. This means that none of the various union types can use more then 480 bytes of data, else they may overwrite the tr_rectype field. The constants HISTTYPE_???_* can be used with the tr_rectype field. The structure also includes an unnamed union.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
LONG	tr_datetime	ctime when transmission event happened.
USHORT	d.trx.elapsed_time	Elapsed time in two-second intervals.
BYTE	d.trx.hangup_status	Reserved. Initialize to zero.
USHORT	d.trx.bad_page_count	Total number of bad or unsent pages.
USHORT	d.trx.channel_used	Channel used to send fax.
USHORT	d.trx.disposition	Contains the final termination status of the fax as reported by the fax board.
USHORT	d.trx.termstat	See TERMSTAT_???.
CHAR	d.trx.remoteid[22]	CSID of remote fax machine.
USHORT	d.trx.usBoardType	Type of board used. See TRXREC_BOARDTYPE_???.
USHORT	d.trx.good_page_count	Number of sent pages.
CHAR	d.trx..szCXMessageID[32]	CallXpress Message ID.
CHAR	d.trx..szFaxNumber[32]	Fax number that the fax was sent to.

C Data Type	Member	Description
CHAR	d.trx..szMsgFromTransport[96]	Message from transport.
CHAR	d.trx..szDelegate_id[22]	Delegate ID.
UCHAR	d.trx.reserved2[252]	This is not a zero terminated string.
CHAR	d.trx.szRemoteServer[48]	Computer name of remote server if sent using LCR.
CHAR	d.trx.szANI[32]	ANI from inbound call, if supplied. usFlags can be examined to determine if ANI is valid (bit 2).
ULONG	d.trx.aulAOCData[3]	Advice of charges (AOC) data from PSTN if supplied. usFlags can be examined to determine if AOC is valid (bit 3).
USHORT	d.trx.usISDNCauseVal	ISDN cause value, if returned from ISDN network interface board.
DWORD	d.trx.dwTransportError	
SHORT	d.trx.BTCallResult.sStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
SHORT	d.trx.BTCallResult.sLineStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
SHORT	d.trx.BTFaxResult.sStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
SHORT	d.trx.BTFaxResult.sLineStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
USHORT	d.trx.usFlags	Bit 0 = Partial re-tries? Bit 1 = Sent remotely? Bit 2 = ANI is valid? Bit 3 = AOC is valid?
USHORT	d.trx.gammaerr	Only valid if usBoardType = TRXREC_BOARDTYPE_GAMMA
ULONG	d.netforward.ulFlags	Reserved. Initialize to zero.
CHAR	d.netforward.szPrevOwnerID[22]	Previous owner's ID.
CHAR	d.netforward.szUsersForwardedTo[128]	List of users fax was forwarded to, separated by commas.
CHAR	d.netforward.szNotes[128]	Notes.
ULONG	d.route.ulFlags	See HISTROUTEFLAG_???
CHAR	d.route.szPrevOwnerID[22]	Previous owner's ID.
CHAR	d.route.szNotes[454]	Notes.
ULONG	d.print.ulFlags	Reserved. Initialize to zero.

C Data Type	Member	Description
ULONG	d.print.ulTimeToPrint	Time, in seconds, needed to print job.
ULONG	d.print.ulError1	Error from the print attempt.
ULONG	d.print.ulError2	Error from the print attempt.
USHORT	d.print.usPagesPrinted	Number of pages printed.
USHORT	d.print.usCopiesPrinted	Number of copies printed.
CHAR	d.print.szNetPrintID[10]	ID of printer where fax was printed.
CHAR	d.print.szMsg[128]	Message about printing.
ULONG	d.ocr.ulTimeToOCR	Total time, in seconds, to OCR.
ULONG	d.ocr.ulError1	Error from the OCR attempt.
CHAR	d.ocr.szMsg[408]	Message about OCR job.
ULONG	d.approved.ulFlags	Reserved. Initialize to zero.
CHAR	d.approved.szApproverID[22]	User ID of person who approved the fax.
CHAR	d.approved.szNotes[454]	Notes.
ULONG	d.disapproved.ulFlags	Reserved. Initialize to zero.
CHAR	d.disapproved.szApproverID[22]	User ID of person who disapproved the fax.
CHAR	d.disapproved.szNotes[454]	Notes.
CHAR	d.converr.szConvType[64]	Indicates the type of conversion that was attempted.
CHAR	d.converr.szErrorMsg[128]	Message about the conversion error.
ULONG	d.converr.ulError1	Error from the conversion attempt.
ULONG	d.converr.ulError2	Error from the conversion attempt.
CHAR	szErrorMsgCont[280]	Space for conversion results
ULONG	d.filerouted.ulFlags	
ULONG	d.filerouted.ulError	
CHAR	d.filerouted.szRoutePath[256]	
ULONG	d.securedocs.ulFlags	Reserved. Initialize to zero.

C Data Type	Member	Description
CHAR	d.securedocs.szSMTPSender[64]	Email from field.
CHAR	d.securedocs.szSMTPHost[64]	SMTP mail host.
CHAR	d.securedocs.szRemoteIP[32]	REMOTEADDR of HotLink recipient
CHAR	d.securedocs.szUserAgent[64]	USERAGENT used for HotLink recipient
CHAR	d.securedocs.szCurrentStatus[32]	Text status (“Emailed,” “Viewed,” etc.).
USHORT	d.securedocs.termstat	One of the TERMSTAT_??? constants.
CHAR	d.securedocs.szExtendStatus[202]	
CHAR	d.securedocs.reserved[16]	
ULONG	d.interconnect.ulFlags	
CHAR	d.interconnect.szFromServer[256]	
CHAR	d.interconnect.szMsg[128]	
ULONG	d.interconnect.ulError	
DWORD	d.---dwGenHistType	Item type. See GENHISTTYPE_??? macros.
DWORD	d.---.dwErrCode	RFAPI error code.
CHAR	d.---.szUserID[22]	User who generated this item.
DWORD	d.---.adwParms[8]	Optional details for this item.
CHAR	d.---.aszParms[6][32]	Optional details for this item.
CHAR	d.---.szShortMsg[32]	Short description, ex: "Combined" or "Forwarded."
CHAR	d.---.szDetailMsg[192]	Detailed description. See RFaxGetHistoryItemDetails().
UCHAR	d.physical.tr_data[480][1]	Reserved. Initialize to zero.
CHAR	d.physical.reserved[1]	Reserved. Initialize to zero.
UCHAR	d.physical.tr_rectype	See HISTTRXFLAG_???.

HISTLISTELEMENT1

The HISTLISTELEMENT1 structure contains the general history record for transport via API.

C Data Type	Member	Description
FAXHANDLE	handle	History record handle
FAXHANDLE	hOwner	Owning fax handle.
LONG	tr_datetime	Transaction time.
LONG	tCreationTime	History record creation time.
unsigned char	tr_rectype	Type of transaction.
unsigned char	reserved	
ULONG	ulFlags	
TCHAR	szUserID[22]	
TCHAR	szMsg[454]	
ULONG	ulError1	
ULONG	ulError2	
TCHAR	szPath[256]	
ULONG	ulType	
ULONG	aulNumbers[12]	
TCHAR	aszStrings[6][64]	
ULONG	aulData1[3]	

HISTLISTELEMENT2

The HISTLISTELEMENT2 structure is a larger typed history list element that takes advantage of the newer history record size. It contains a union. The datatype held in the union is specified by the tr_rectype. The maximum size of each structure within the union is 800 bytes.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
LONG	tr_datetime	ctime when transmission event happened.
UCHAR	tr_rectype	Can use HISTTYPE_???.
UCHAR	ucReserved1	Reserved for future use to maintain proper alignment.

C Data Type	Member	Description
UCHAR	aucReserved[40]	Reserved space for things that exist for every history record.
USHORT	d.trx.elapsed_time	Elapsed time in two-second intervals.
BYTE	d.trx.hangup_status	Reserved. Initialize to zero.
USHORT	d.trx.bad_page_count	Total number of bad or unsent pages.
USHORT	d.trx.channel_used	Channel used to send fax.
USHORT	d.trx.disposition	Contains the final termination status of the fax as reported by the fax board.
USHORT	d.trx.termstat	See TERMSTAT_???.
CHAR	d.trx.remoteid[22]	CSID of remote fax machine.
USHORT	d.trx.usBoardType	Type of board used. See TRXREC_BOARDTYPE_???.
USHORT	d.trx.good_page_count	Number of sent pages.
CHAR	d.trx.szCXMessageID[32]	CallXpress Message ID.
CHAR	d.trx.szFaxNumber[32]	Fax number that the fax was sent to.
CHAR	d.trx.szMsgFromTransport[96]	Message from transport.
CHAR	d.trx.szDelegate_id[22]	Delegate ID.
CHAR	d.trx.szRemoteServer[48]	Computer name of remote server if sent using LCR.
CHAR	d.trx.szANI	ANI from inbound call, if supplied. usFlags can be examined to determine if ANI is valid (bit 2).
ULONG	d.trx.aulAOCData[3]	Advice of charges (AOC) data from PSTN if supplied. usFlags can be examined to determine if AOC is valid (bit 3).
USHORT	d.trx.usISDNCauseVal	ISDN cause value, if returned from ISDN network interface board.
DWORD	d.trx.dwTransportError	
SHORT	d.trx.BTCallResult.sStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
SHORT	d.trx.BTCallResult.sLineStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
SHORT	d.trx.BTFaxResult.sStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW
SHORT	d.trx.BTFaxResult.sLineStatus	Only valid if usBoardType = TRXREC_BOARDTYPE_BROOKNEW

C Data Type	Member	Description
USHORT	d.trx.usFlags	Bit 0 = Partial re-tries? Bit 1 = Sent remotely. Bit 2 = ANI is valid. Bit 3 = AOC is valid. Bit 4 = ISDN Cause is valid. Bit 5 = EFSP sent fax, do not retry.
USHORT	d.trx.gammaerr	Only valid if usBoardType = TRXREC_BOARDTYPE_GAMMA
ULONG	d.netforward.ulFlags	Reserved. Initialize to zero.
CHAR	d.netforward.szPrevOwnerID[22]	Previous owner's ID.
CHAR	d.netforward.szUsersForwardedTo[128]	List of users fax was forwarded to, separated by commas.
CHAR	d.netforward.szNotes[128]	Notes.
ULONG	d.route.ulFlags	One of the ROUTEFLAGS_??? constants
CHAR	d.route.szPrevOwnerID[22]	Previous owner's ID.
CHAR	d.route.szNotes[454]	Notes.
ULONG	d.print.ulFlags	Reserved. Initialize to zero.
ULONG	d.print.ulTimeToPrint	Time, in seconds, needed to print job.
ULONG	d.print.ulError1	Error from the print attempt.
ULONG	d.print.ulError2	Error from the print attempt.
USHORT	d.print.usPagesPrinted	Number of pages printed.
USHORT	d.print.usCopiesPrinted	Number of copies printed.
CHAR	d.print.szNetPrintID[10]	ID of printer where fax was printed.
CHAR	d.print.szMsg[128]	Message about printing.
ULONG	d.ocr.ulTimeToOCR	Total time, in seconds, to OCR.
ULONG	d.ocr.ulError1	Error from the OCR attempt.
CHAR	d.ocr.szMsg[408]	Message about OCR job.
CHAR	d.converr.szConvType[64]	Conversion type.

C Data Type	Member	Description
ULONG	d.converr.ulError1	Error from the conversion attempt.
ULONG	d.converr.ulError2	Error from the conversion attempt.
CHAR	d.converr.szMsg[700]	Message about conversion.
ULONG	d.approved.ulFlags	Reserved. Initialize to zero.
CHAR	d.approved.szApproverID[22]	User ID of person who approved the fax.
CHAR	d.approved.szNotes[454]	Notes.
ULONG	d.disapproved.ulFlags	Reserved. Initialize to zero.
CHAR	d.disapproved.szApproverID[22]	User ID of person who disapproved the fax.
CHAR	d.disapproved.szNotes[454]	Notes.
ULONG	d.filerouted.ulFlags	
ULONG	d.filerouted.ulError	
CHAR	d.filerouted.szRoutePath[256]	
ULONG	d.securedocs.ulFlags	Reserved. Initialize to zero.
CHAR	d.securedocs.szSMTPSender[64]	Email from field.
CHAR	d.securedocs.szSMTPHost[64]	SMTP mail host.
CHAR	d.securedocs.szRemoteIP[32]	REMOTEADDR of HotLink recipient
CHAR	d.securedocs.szUserAgent[64]	USERAGENT used for HotLink recipient
CHAR	d.securedocs.szCurrentStatus[32]	Text status (“Emailed,” “Viewed,” etc.).
USHORT	d.securedocs.termstat	One of the TERMSTAT_??? constants.
CHAR	d.securedocs.szExtendStatus[202]	
CHAR	d.securedocs.reserved[16]	
ULONG	d.interconnect.ulFlags	
CHAR	d.interconnect.szFromServer[256]	
CHAR	d.interconnect.szMsg[128]	
ULONG	d.interconnect.ulError	

C Data Type	Member	Description
DWORD	d.general.dwGenHistType	Item type. See GENHISTTYPE_??? macros.
DWORD	d.general.dwErrCode	RFAPI error code.
CHAR	d.general.szUserID[22]	User who generated this item.
DWORD	d.general.adwParms[8]	Optional details for this item.
CHAR	d.general.aszParms[6][64]	Optional details for this item.
CHAR	d.general.szShortMsg[32]	Short description, ex: "Combined" or "Forwarded." This is only filled in by the server and is already localized for the client's langid.
CHAR	d.general.szDetailMsg[192]	Detailed description. See RFaxGetHistoryItemDetails(). This is only filled in by the server and is already localized for the client's langid.
UNSIGNED UCHAR	d.physical.tr_data[800][1]	This establishes the maximum size of each structure within the union to be 800 bytes.

ICONINFO_10

The ICONINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of icon.
FAXHANDLE	hOwner	Handle of owning user or NULL if no owner.
LONG	tLastUpdated	Last time the icon was updated.
TCHAR	szDescription[64]	Description of icon.
USHORT	usFlags	Flags for this icon. See *ICONFLAG_??? on page 444*.
CHAR	reserved[178]	

LIBDOCLISTELEMENT0

The LIBDOCLISTELEMENT0 structure is used with the RFaxGetLibDocList() function.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.

C Data Type	Member	Description
ULONG	flags	See GFLAG_???.
CHAR	objectid[22]	ID of the library document element.
CHAR	filename[14]	Name of the file associated to the library document.
CHAR	description[36]	Description of the library document.
USHORT	numpages	Total number of pages in the library document.

LIBDOCLISTELEMENT1

The LIBDOCLISTELEMENT1 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See GFLAG_???.
CHAR	objectid[22]	ID of the library document element.
CHAR	filename[14]	Name of the file associated to the library document.
CHAR	description[36]	Description of the library document.
USHORT	numpages	Total number of pages in the library document.
ULONG	ulTimesRequestedFOD	Number of times document has been requested using Docs-on-Demand.
ULONG	ulTimesUsedLAN	Number of times the document has been requested on the LAN.
ULONG	ulEmbargoDate	Time formatted in ctime indicating the date the library document is embargoed until.
ULONG	ulExpireDate	Time formatted in ctime when document is expires.
ULONG	ulParentFolder	ID of folder in which this document exists (0 if root).
ULONG	ulFolderID	Only valid if GFLAG_FOLDER is set.
ULONG	ulTimesRequestedWEB	Number of times the document has been requested using the Web Fax Tools.
ULONG	ulLastUsed	Date and time document was last used on LAN, Web, or FOD.
CHAR	szFodPassword[22]	Password for Docs-On-Demand.
ULONG	ulExcludedGroupCount	The number of groups this document is excluded from.

C Data Type	Member	Description
CHAR	reserved[996]	Reserved. Initialize to zero.

LIBDOCLISTELEMENT2

The LIBDOCLISTELEMENT2 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	See GFLAG_???.
CHAR	objectid[22]	ID of the library document element.
CHAR	filename[14]	Name of the file associated to the library document.
CHAR	description[36]	Description of the library document.
USHORT	numpages	Total number of pages in the library document.
ULONG	ulTimesRequestedFOD	Total number of times the library document has been requested via Docs-on-Demand.
ULONG	ulTimesUsedLAN	Total number of times the library document has been requested on the LAN.
ULONG	ulEmbargoDate	Time formatted in ctime indicating the date the library document is embargoed until.
ULONG	ulExpireDate	Time formatted in ctime when document is expires.
ULONG	ulParentFolder	ID of folder in which the document exists (0 if root).
ULONG	ulFolderID	Only valid if GFLAG_FOLDER is set.
ULONG	ulTimesRequestedWEB	Total number of times the library document has been requested on the Web.
ULONG	ulLastUsed	Date and time the document was last used on LAN, Web, or Fax-on-Demand.
CHAR	szFodPassword[22]	Password for Docs-On-Demand.
CHAR	reserved[62]	Reserved. Initialize to zero.

LOCALSUPPORTINFO

The LOCALSUPPORTINFO structure is used with the RFaxGetLocalSupportInfo() function.

C Data Type	Member	Description
CHAR	szText[24][60]	Up to 24 lines of support information from the Contact.txt file in the RightFax\Bin folder.

LOCKDBDATA

The LOCKDBDATA structure is used with the RFaxCheckDatabaseLock() function.

C Data Type	Member	Description
ULONG	ulflags	Reserved for future use; set to 0.
CHAR	szUserID[22]	The user motivating this lock. This must be an administrator with NT auth or valid password.
CHAR	szModule[128]	The module being used.
FAXBOOL	bBlockReads	Indicates the faxdb should return errors for all requests that use read-only access to the database. The faxserv should suspend all workflow ASAP.
FAXBOOL	bBlockWrites	Indicates the faxdb should return errors for all requests that use write access to the database. The faxserv should suspend all workflow ASAP.
FAXBOOL	bFSRecognized	Set by the faxserv when it has noticed the lock and stopped workflow.
FAXBOOL	bFDRecognized	Set by the faxdb when all threads have noticed the lock and started refusing access.

LOGININFO

The LOGININFO structure.

C Data Type	Member	Description
LOGINFLAGS	ulFlags	See LOGINFLAGS_???.
BYTE	SID[88]	Current user's NT SID.
CHAR	szNTAccount[1024]	Current user's NT account.
CHAR	szUserID[22]	Current user's matching RightFax user ID, if found.
FAXHANDLE	hUser	Current user's matchingRightFax user handle, if found.

MESSAGEREQUEST

The MESSAGEREQUEST structure is used with calls to the RFaxGetMessageEvent() function.

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.
ULONG	ulRequestType	See REQUEST_???.
FAXHANDLE	handle	Handle of the fax.
SHORT	fDeleteFax	Always 0.
SHORT	sMsgType	One of the MSGTYPE_??? constants.
CHAR	szMsgText[200]	OEM character set string.
CHAR	szUserID[22]	RightFax user ID to which to send message.

MSMAIL1REQ

The MSMAIL1REQ structure. See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
FAXBOOL	fDeleteFax	True = Delete fax after successful route.
SHORT	sMsgType	One of the MSGTYPE_??? constants.
CHAR	szMsgText[200]	Text of message.
CHAR	szUserID[22]	User ID.
CHAR	szNetwork[20]	MS Mail network.
CHAR	szPostOffice[20]	MS Mail PO.
AUTOPRINTREQ	print	AUTOPRINTREQ structure.
USHORT	usFileFormat	One of the IMAGEFMT_??? constants.
SHORT	sOCRError	If the fax was OCRed, this will indicated whether it was successful or not.

NOTEINFO_10

The NOTEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	Handle	Database handle for this fax record (these get re-used).

C Data Type	Member	Description
CHAR	text[21 * 70]	Up to 21 lines of up to 69 characters each. Each line containing text should be terminated with \n. Entire buffer terminated with \0.
CHAR	reserved[56]	Reserved. Initialize to zero.

OCRREQ

See WORKREQUEST for more information on this structure.

C Data Type	Member	Description
SHORT	usOCRFlags	One of the OCRFLAGS_??? constants.
SHORT	sStartpage	Starting page to OCR.
SHORT	sEndpage	Ending page to OCR.
SHORT	sLayout	One of the OCRLAYOUT_??? constants.
SHORT	sFormat	One of the OCRFORMAT_??? constants.
CHAR	szUserID[22]	Owner's user ID.
CHAR	szOutputName[14]	Output file name.
MSMAIL1REQ	routedata.msmail1	MSMAIL1REQ structure.
CCMAIL1REQ	routedata.ccmail1	CCMAIL1REQ structure.
WPOMAIL1REQ	routedata.wpomail1	WPOMAIL1REQ structure.
GENMAIL1REQ	routedata.genmail1	GENMAIL1REQ structure.
FILERTE1REQ	routedata.filerte1	FILERTE1REQ structure.
AUTOPRINTREQ	routedata.print	AUTOPRINTREQ structure.

PHONEBOOKINFO_10

The PHONEBOOKINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle of phonebook.

C Data Type	Member	Description
CHAR	phonebookName[32]	Phonebook name
CHAR	SQLFromClause[128]	
CHAR	SQLWhereClause[128]	
CHAR	SQLOrderBy[128]	
CHAR	SQLConnectionString[128]	
CHAR	SQLFields[NUM_FIELDS][64]	
CHAR	reserved[776]	

PHONEITEM

The PHONEITEM structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of phonebook item.
ULONG	flags	See PHONEFLAG_???.
CHAR	user_id[22]	ID of user.
CHAR	entry_id[18]	ID of entry.
CHAR	d.item.name[60]	Name.
CHAR	d.item.company[60]	Company.
CHAR	d.item.address[60]	Address.
CHAR	d.item.citystate[60]	City/State.
CHAR	d.item.fax1[32]	Fax #1.
CHAR	d.item.fax2[32]	Fax #2.
CHAR	d.item.billinfo1[18]	Billing information field 1.
CHAR	d.item.billinfo2[18]	Billing information field 2.
CHAR	d.item.voicenum1[32]	Voice telephone number 1.
CHAR	d.item.voicenum2[32]	Voice telephone number 2.

C Data Type	Member	Description
CHAR	d.item.reserved1[16]	Reserved. Initialize to zero.
CHAR	d.item.notes1[128]	Notes.
CHAR	d.item.szSecureCSID[22]	CSID of remote fax machine.
CHAR	d.item.reserved[42]	Reserved. Initialize to zero.
CHAR	d.keys[24][18]	Phonebook IDs of entries contained in group, if this is a group entry. Flags must be examined to determine whether or not this is a group or individual entry. Note that with RightFax 6.0, it is possible to have more than 24 entries in a group, but that functionality is not easily exposed in the API at this time.
CHAR	reserved[92]	Reserved. Initialize to zero.

PHONELISTELEMENT0

The PHONELISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of phonebook element.
ULONG	flags	See PHONEFLAG_???.
CHAR	id[18]	ID of phonebook entry.
CHAR	description[64]	This member lets you enter a recipient name, company name, and fax number into a single field. See notes below.
CHAR	name[26]	Recipient name (do not use if you are using the description member).
CHAR	company[20]	Recipient company (do not use if you are using the description member).
CHAR	fax[17]	Recipient fax number (do not use if you are using the description member).
CHAR	userid[22]	User ID of the owner of the phone list element.
CHAR	reserved[16]	Reserved. Initialize to zero.

PHONELISTELEMENT1

The PHONELISTELEMENT1 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of the phone element.
ULONG	flags	See PHONEFLAG_???.
CHAR	id[18]	ID of phonebook entry.
CHAR	name[64]	Name of phonebook entry.
CHAR	company[17]	Company name of phonebook entry.
CHAR	fax1[21]	Fax number 1.
CHAR	userid[22]	User ID.

PRINTERINFO_???

Constants for the flags field of PRINTERINFO_10

Constant	Value	Description
PRINTERFLAGS_DIRECTIP	0x00000002	Direct IP connection to the printer.

PRINTERINFO_10

The PRINTERINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular element.
ULONG	flags	One of the PRINTERFLAGS_??? constants.
CHAR	netprint_id[10]	ID of printer.
CHAR	description[48]	Description of printer.
CHAR	servername[48]	Name of the server and the queue is attached to.
CHAR	queuename[48]	Name of the print queue.
SHORT	printertype	See PRINTERTYPE_???.

C Data Type	Member	Description
SHORT	spooled	Indicates that the print job is spooled on the server.
SHORT	sNovellformtype	Reserved. Initialize to zero.
SHORT	sDefSize	See PAPERSIZE_???.
SHORT	sDefSource	See PAPERSOURCE_???.
CHAR	reserved[68]	Reserved. Initialize to zero.

PRINTERLISTELEMENT0

The PRINTERLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	flags	One of the PRINTERFLAGS_??? constants.
CHAR	netprint_id[10]	ID of the printer.
CHAR	description[48]	Description of the printer.
CHAR	servername[48]	Name of the server that the printer queue is attached to.
CHAR	queuename[48]	Name of the print queue.
SHORT	printertype	See PRINTERTYPE_???.
SHORT	spooled	Indicates that the print job is spooled on the server.
SHORT	sNovellformtype	Reserved. Initialize to zero.
SHORT	sDefSize	See PAPERSIZE_???.
SHORT	sDefSource	See PAPERSOURCE_???.

RECIPIENTLISTELEMENT0

The RECIPIENTLISTELEMENT0 structure. For each element that is not set, the server will use the value in the fax record. Therefore it is recommended that you initialize all elements to zero before use.

C Data Type	Member	Description
CHAR	szBillInfo1[16]	Accounting code #1.
CHAR	szBillInfo2[16]	Accounting code #2.
LONG	lFaxSendDateTime	0 if ASAP, else time fax should be sent (delay send) - 'C' time type (GMT0).
CHAR	szFromName[60]	Source name.
CHAR	szFromPhoneNum[32]	Source voice number.
CHAR	szToAltFax[32]	Alternate destination fax number.
CHAR	szToCityState[60]	Destination location name.
CHAR	szToContactNum[32]	Destination voice number.
CHAR	szToCompany[60]	Destination company name.
CHAR	szToFaxNum[32]	Destination fax number.
CHAR	szToName[60]	Destination name.
CHAR	szCmdDataFileName[14]	Cmd file name. Must be in in the RIGHTFAX\CMDDATA directory.
CHAR	szSecureCSID[22]	Blank if secure sending is not enabled.
UCHAR	ucPriority	See PRIORITY_???
CHAR	szUniqueID[16]	Unique user or system provided id for fax job.
SHORT	sCompleteEvent	0 = use default 1 = No complete event 2 = fire complete event
SHORT	sCover	0 = use default 1 = No cover sheet 2 = cover sheet
SHORT	sAutoDelete	0 = use default 1 = no delete 2 = delete on failure 3 = delete on success & failure
ULONG	ulLstFlags	See LSTFLAG__.
UCHAR	ucRecipNotifyType	Type of recipient notification to use. See NOTIFY_TYPE_???. Not all methods are currently supported.

C Data Type	Member	Description
UCHAR	ucPreferredContentType	Preferred type of file after conversion.
CHAR	szRecipNotifyAddress[32]	Recipient notification information.
UCHAR	aucReserved[64]	Reserved--always initialize to 0.(BEWARE of pack(2) and maintain proper size).

REQUESTSTATUS

The REQUESTSTATUS structure is used with calls to the RFaxSaveStatus2() function

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.
ULONG	handle	Object to which this status refers (usually a handle to a fax).
SHORT	status	Status result (0 = success, non-zero = failure).
CHAR	userid[22]	
USHORT	numpages	Number of pages produced by operation.
ULONG	numbytes	Number of bytes produced by operation.
CHAR	reserved[84]	

RFAXFAXFILTER

The RFAXFAXFILTER structure includes an unnamed union.

Initialize any empty structure fields to zero for maximum data transfer rate to the server.

C Data Type	Member	Description
CHAR	userId[22]	User performing the search. Should always be specified. Required when using SEARCH_INCLUDE_DELEGATORS or SEARCH_INCLUDE_SELF.
ULONG	ulAndFlags	Bit field using FAXFLAG_???. Only documents that have ALL of these flags will be returned.
ULONG	ulAndNotFlags	Bit filed using FAXFLAG_???. Only documents that do NOT have ALL of these flags will be returned.
ULONG	ulOrFlags	Bit filed usingFAXFLAG_???. Only documents that have ANY one of these flags will be returned.

C Data Type	Member	Description
ULONG	ulOrNotFlags	Bit filed usingFAXFLAG_???. Only documents that do NOT have ANY one of these flags will be returned.
ULONG	ulAndFlags2	Bit filed using FAXFLAG2_???. Only documents that have ALL of these flags will be returned.
ULONG	ulAndNotFlags2	Bit filed using FAXFLAG2_???. Only documents that do NOT have ALL of these flags will be returned.
ULONG	ulOrFlags2	Bit filed using FAXFLAG2_???. Only documents that have ANY one of these flags will be returned.
ULONG	ulOrNotFlags2	Bit filed usingFAXFLAG2_???. Only documents that do NOT have ANY one of these flags will be returned.
TIME_T	startTime	Only documents created at or after this time will be returned. Set to zero to disable.
TIME_T	endTime	Only documents created at or before this time will be returned. Set to zero to disable.
ULONG	searchFlags	Bit field using members of RFAXFAXFILTERFLAGS.
FAXHANDLE	userHandleLIst [256/sizeof (FAXHANDLE)]	Contiguous array of user handles whose documents will be included.
FAXHANDLE	groupHandleLIst [256/sizeof (FAXHANDLE)]	Contiguous array of group handles whose documents will be included. To use, specify FAXFILTER_ USERGROUP_HANDLES in searchFlags.
CHAR	csid[22]	Searches for matching values in FAXINFO_11.szSecureCSID. Use % as wildcard; %% for percent symbol.
CHAR	comment[70]	Searches for matching values in FAXINFO_11.szComment. Use % as wildcard; %% for percent symbol.
CHAR	destination[32]	Searches for matching values in FAXINFO_11.to_faxnum. Use % as wildcard; %% for percent symbol.
CHAR	toName[60]	Searches for matching values in FAXINFO_11.to_name. Use % as wildcard; %% for percent symbol.
CHAR	generalVoice[32]	Searches for matching values in FAXINFO_11.operatornum. Use % as wildcard; %% for percent symbol.
CHAR	company[60]	Searches for matching values in FAXINFO_11.to_company. Use % as wildcard; %% for percent symbol.
CHAR	cityState[60]	Searches for matching values in FAXINFO_11.to_citystate. Use % as wildcard; %% for percent symbol.

C Data Type	Member	Description
CHAR	billInfo1[16]	Searches for matching values in FAXINFO_11.billinfo1. Use % as wildcard; %% for percent symbol.
CHAR	billInfo2[16]	Searches for matching values in FAXINFO_11.billinfo2. Use % as wildcard; %% for percent symbol.
CHAR	aniMatch[32]	Searches for matching values in HISTLISTELEMENT0.d.trx.szANI. Use % as wildcard; %% for percent symbol.
INT	iFolderIndex	Folder index to use.
CHAR	wordSearch[100]	Requires RightFax Server 16.2 or later. Searches, by word, for matching values inFAXINFO_11.szSecureCSID, szComment, to_faxnum, to_name, operatornum, to_company, to_citystate, billinfo1, billinfo2. For example, an entry of John Doe would find documents with John OR Doe in any of these fields.
USHORT	usFaxFilter	Requires RightFax Server 16.2 or later. Specify one of FAXFILTER_???. This is equivalent to the usFilter parameters to RFaxGetFaxList#. Ignored when FAXFILTER_NONE.
SHORT	timeBias	Defaults to UTC. Set as if using Windows Time Zone bias information, such as Tucson (-7).
CHAR	jobId[32]	Searches for matching values in FAXINFO_11.szJobId
CHAR	uniqueId[16]	Searches for matching values in FAXINFO_11.unique_id.
FAXHANDLE	hFax	Searches for matching values inFAXINFO_11.handle.
CHAR	toContactNumber[32]	Searches for matching values in FAXINFO_11.to_contactnum.
CHAR	ocrText[64]	When specified, indicates OCR text that a fax must have.
FAXHANDLE	hWorkflow	Handle of workflow or NULL if none.
CHAR	workflowValue[32]	Optional workflow value.
ULONG	errorCodeFlags	Bit field using members of *RFAXFAXFILTERERRORCODEFLAGS_??* on page 471.
BYTE	ucReserved[638]	

RFAXQUERYINFO

The RFAXQUERYINFO structure is used with calls to the RFaxQueryFaxesToCSV(), RFaxQueryFaxesToCSV2() and RFaxEnumAllFaxes() functions.

C Data Type	Member	Description
SHORT	sInfoLevel	Must be 0 or 1.

C Data Type	Member	Description
SHORT	sStartDateMonth	01–12.
SHORT	sStartDateDay	01–31.
SHORT	sStartDateYear	The desired year minus 1900. For example, the year 2002 is represented as 102. This value must be >=70.
SHORT	sEndDateMonth	01–12.
SHORT	sEndDateDay	01–31.
SHORT	sEndDateYear	The desired year minus 1900. For example, the year 2002 is represented as 102. This value must be >=70.
SHORT	fIncludeSent	0 = False, 1 = True.
SHORT	fIncludeRecv	0 = False, 1 = True.

RFAXQUERYINFO2

The RFAXQUERYINFO2 structure is used with the call to RFaxEnumAllFaxes2().

C Data Type	Member	Description
SHORT	sInfoLevel	Must be 0 or 1.
SHORT	sDateCheckType	0 = check date/time against FAXINFO_10.fax_datetime 1 = check against FAXINFO_10.histchg_datetime
SHORT	sStartDateMonth	01–12.
SHORT	sStartDateDay	01–31.
SHORT	sStartDateYear	The desired year minus 1900. For example, the year 2002 is represented as 102. This value must be >=70.
SHORT	sStartHour	00–23.
SHORT	sStartMinute	00–59.
SHORT	sEndDateMonth	01–12.
SHORT	sEndDateDay	01–31.

C Data Type	Member	Description
SHORT	sEndDateYear	The desired year minus 1900. For example, the year 2002 is represented as 102. This value must be >=70.
SHORT	sEndHour	00–23.
SHORT	sEndMinute	00–59.
SHORT	fIncludeSent	0 = False, 1 = True.
SHORT	fIncludeRecv	0 = False, 1 = True.

RFAXQUERYINFO3

The RFAXQUERYINFO3 structure is used with the call to RFaxEnumAllFaxes3().

C Data Type	Member	Description
SHORT	sInfoLevel	Must be 0 or 1.
SHORT	sStartDateMonth	01–12.
SHORT	sStartDateDay	01–31.
SHORT	sStartDateYear	The desired year minus 1900. For example, the year 2002 is represented as 102. This value must be >=70.
SHORT	sStartHour	00–23.
SHORT	sStartMinute	00–59.
SHORT	sEndDateMonth	01–12.
SHORT	sEndDateDay	01–31.
SHORT	sEndDateYear	The desired year minus 1900. i.e., the year 2002 is represented as 102. This value must be >=70.
SHORT	sEndHour	00–23.
SHORT	sEndMinute	00–59.
SHORT	fIncludeSent	0 = False, 1 = True.
SHORT	fIncludeRecv	0 = False, 1 = True.
CHAR	szUniqueID[16]	If not blank, find fax starting with this unique ID.

C Data Type	Member	Description
CHAR	szUserID[22]	If not blank, find fax starting with this user ID.
CHAR	szToFaxNum[32]	If not blank, find fax starting with this destination fax number.
CHAR	szToName[60]	If not blank, find fax starting with this destination name.
CHAR	SzToCompany[60]	If not blank, find fax starting with this destination company.
CHAR	reserved1[280]	Reserved space for future use.

SD_ATTACHITEM

The SD_ATTACHITEM structure.

C Data Type	Member	Description
ULONG	ulFlags	One of the SD_ATTACHFLAG_NATIVE_??? constants.
TCHAR	szFilename[256]	Attachment file name.
TCHAR	szDescription[256]	Attachment description.

SERVERINFO

The SERVERINFO structure.

C Data Type	Member	Description
ULONG	ulServerType	See SERVERTYPE_???.
ULONG	ulServerVersion	Hex value defines server version (e.g. 0x0350 = v3.50).
ULONG	ulServerSpecial	See SERVERSPECIAL_???.
CHAR	szImageLocServer[48]	Contains the computer name of the RightFax server.
CHAR	szImageLocVolume[16]	This is the share name in LANMAN/LANSERVER.
SHORT	sMaxUsers	Maximum users supported by the server.
SHORT	sVerifyCodes	Indicates whether or not billing codes are verified by the server. This information can be set via RFaxSaveServerInfo().
CHAR	szBillDesc1[16]	User-defined description of first billing code. This information can be set via RFaxSaveServerInfo().

C Data Type	Member	Description
CHAR	szBillDesc2[16]	User-defined description of second billing code. This information can be set via RFaxSaveServerInfo().
CHAR	acReqFields[14]	Each character indicates whether a field in the Fax Information dialog box has been set to required by the fax server admin. Bits are set within the character stating whether the requirement is for sent or received faxes (see REQFIELD_???). Fields are indexed in the following order: TO_NAME, TO_FAXNUM, TO_VOICENUM, TO_COMPANY, TO_CITYSTATE, FROM_NAME, FROM_VOICENUM, FROM_FAXNUM, FROM_OPERATORNUM, FROM_GENERALNUM, BILLCODE1, BILLCODE2. This information can be set via RFaxSaveServerInfo().
ULONG	ulBuildDate	Build date for the server code (0xYYYYMMDD).
CHAR	szImageDir[128]	Drive:path (guaranteed to end with a backslash) of RightFax image folder.
CHAR	szBaseRFDir[128]	Drive:path (guaranteed to end with a backslash) of RightFax base folder.
ULONG	ulFlags	Bit 0 = Server running in ANSI code page mode. This information can be set via RFaxSaveServerInfo().
FAXBOOL	fEnterprise	Set to True if this is an Enterprise server.
FAXBOOL	fDocsOnDemand	Set to True if Docs-on-Demand is licensed.
FAXBOOL	fTeleconnect	Set to True if TeleConnect is licensed.

SERVERINFO2

The SERVERINFO2 structure.

C Data Type	Member	Description
ULONG	ulServerType	See SERVERTYPE_???.
ULONG	ulServerVersion	Hex value defines server version (e.g., 0x0350 = v3.50).
ULONG	ulServerSpecial	See SERVERSPECIAL_???.
CHAR	szImageLocServer[48]	Contain the computer name of the RightFax server.

C Data Type	Member	Description
CHAR	szImageLocVolume[16]	This is the share name in LANMAN/LANSERVER.
SHORT	sMaxUsers	Maximum users supported by the server.
SHORT	sVerifyCodes	Whether or not billing codes are verified by the server.
CHAR	szBillDesc1[16]	User-defined description of first billing code.
CHAR	szBillDesc2[16]	User-defined description of second billing code.
CHAR	acReqFields[14]	Each character indicates whether a field in the Fax Information dialog box has been set to required by the fax server admin. Bits are set within the character stating whether the requirement is for sent or received faxes (see REQFIELD_???). Fields are indexed in the following order: TO_NAME, TO_FAXNUM, TO_VOICENUM, TO_COMPANY, TO_CITYSTATE, FROM_NAME, FROM_VOICENUM, FROM_FAXNUM, FROM_OPERATORNUM, FROM_GENERALNUM, BILLCODE1, BILLCODE2.
ULONG	ulBuildDate	Build date for the server code (0xYYYYMMDD).
CHAR	szImageDir[128]	Drive:path (guaranteed to end with a backslash) of RightFax image folder.
CHAR	szBaseRFDir[128]	Drive:path (guaranteed to end with a backslash) of RightFax base folder.
ULONG	ulFlags	See SERVERINFO2_FLAGS_???.
FAXBOOL	fEnterprise	Set to True if this is an Enterprise server.
FAXBOOL	fDocsOnDemand	Set to True if Docs-on-Demand is licensed.
FAXBOOL	fTeleconnect	Set to True if TeleConnect is licensed.
FAXBOOL	fSatellite	Set to True if the server is a Satellite license.
FAXBOOL	fSBS	Set to True is the server is a Small Business Server license.
FAXBOOL	fTimeZoneInfoValid	Indicates whether or not TimeZoneInfo has valid data. Version 7.x and later servers will set this to 3/2/0 (daylight/standard/invalid), where 3 indicates the server is operating under daylight savings, 2 indicates the server is operating under standard time.
TIME_ZONE_INFORMATION	TimeZoneInfo	As used/defined by Win32 GetTimeZoneInformation. Reflects time zone information about the fax server, which can be different from that of the client/caller.
CHAR	szANSICP[10]	ANSI code page in use at the server (in particular, the code page of the RightFax Database Server Service).

C Data Type	Member	Description
CHAR	szOEMCP[10]	OEM code page in use at the server (in particular, the code page of the RightFax Database Server Service.
FAXBOOL	fProdEnabled	Is the Production module licensed on the server?
FAXBOOL	fProdINL	Is the Production INL module licensed on the server?
FAXBOOL	fProdFilter	Is the Filter for Production module licensed on the server?
FAXBOOL	fProdNotifier	Is the Production Notifier licensed on the server?
FAXBOOL	fIPPlusConnector	Is IPPlus licensed on the server?
FAXBOOL	fOCRRouter	Is the OCR Routing module licensed on the server?
FAXBOOL	fOCRConv	Is the OCR Conversion module licensed on the server?
FAXBOOL	fPDFModule	Is the PDF module licensed on the server?
FAXBOOL	fSerialValid	Is the server's serial number valid?
CHAR	szSerialNumber[36]	Contains the server's serial number.
FAXBOOL	fOEMCPMode	Is the character set in the OEM character set mode?
SHORT	sLiscensedChans	Total number of licensed channels.
FAXBOOL	fSecureDoc	Is the SecureDocs module licensed on the server?
ULONG	ulBumpBits	Reflects bumps in a single bit flags field. Use BUMP_??? flags to mask out required bits.
LONG	lBumpDate	Date that the server was last bumped.
SHORT	fSMTPDisabled	Indicates whether SMTP features are enabled or disabled.
CHAR	szSystemIdentifier[40]	The System Unique Identifier (SUID) used when the server was licensed.
SHORT	sCollectiveServerListCount	Number of nodes in the Shared Services collective.
CHAR	szAllowedDialingDigits[128]	Dialing digits from the server.
SHORT	sLicensedFoipChans	Number of licensed FoIP channels
SHORT	fRequireStrongPassword	Server requires login with strong password.
SHORT	sFileCleanUpRetentionDays	When cleaning up older files, the number of days since their last modification date.
LONG	tMaintenanceTime	Time to perform above clean up.

C Data Type	Member	Description
ULONG	ulMinimumClientVersion	Minimum client version expressed in hex to match PIPEREQUEST.d.dwClientVersion.
ULONG	ulBumpBitsHigh	Reserved for internal use.
ULONG	ulServerFlags	Values for VERSIONINFO_0.ulFlags.
SHORT	sClientInactivityTimeout	Time in minutes to log out the user from client application after inactivity.
CHAR	reserved[8]	Reserved space.

SERVERINFO3

The SERVERINFO3 structure includes everything in the SERVERINFO2 structure in addition to the following fields.

C Data Type	Member	Description
CHAR	szQueueServerName[64]	MSMQ server address.
SHORT	sLicensedExclusiveGroups	
SHORT	sLicensedSecureFoipChans	
SHORT	sLicensedSAPConnectors	
CHAR	szSharedGuid[40]	GUID used to identify the Shared Services system.
CHAR	reserved2[454]	Reserved for future use.

SERVICEINFO_10

The SERVICEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	
CHAR	szHost[16]	The host that this service runs on. Not necessarily the fax server it belongs to.
CHAR	szDescription[128]	Description of the service.
RFSERVICETYPE	usType	The type of service. See *RFSERVICETYPE_??? on page 474*.
LONG	lUpdateUsn	USN for configuration changes. Used by service to detect when their configuration needs to be refreshed.
CHAR	reserved[102	Reserved for future use. Must be initialized to 0.

SETTINGINFO_10

The SETTINGINFO_10 structure.

C Data Type	Member	Description
TCHAR	szName [64]	Name for the setting.
TCHAR	szValue [1800]	Value for the setting.

SETTINGLISTELEMENT0

The SETTINGLISTELEMENT0 structure.

C Data Type	Member	Description
TCHAR	szName[64]	Name for the setting.

SIGINFO_10

The SIGINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of signature.
ULONG	flags	Reserved. Initialize to zero.
CHAR	signid[22]	Signature ID.
CHAR	signowner[22]	Owner ID.
CHAR	signfile[14]	Signature file.
CHAR	signdesc[36]	Signature description.
CHAR	signusers[3][22]	Users that can use this signature.
CHAR	reserved[72]	Reserved space.

SIGLISTELEMENT0

The SIGLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of signature.
CHAR	signid[22]	Signature ID.
CHAR	signowner[22]	Owner ID.
CHAR	signdesc[36]	Signature description.
CHAR	signusers[68]	Users that can use this signature.

STAMPINFO_10

Deprecated.

The STAMPINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of stamp.
FAXHANDLE	hOwner	Handle of owning user or NULL if no owner.
LONG	tLastUpdated	Last time the stamp was updated.
TCHAR	szDescription[64]	Description of stamp.
CHAR	reserved[180]	

STAMPLISTELEMENT0

Deprecated.

The STAMPLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of stamp.
FAXHANDLE	hOwner	Handle of owning user or NULL if no owner.
LONG	tLastUpdated	Last time the stamp was updated.
TCHAR	szDescription[64]	Description of stamp.

STATUSVALUEENTRY

The STATUSVALUEENTRY structure contains an unnamed union. The valid member of the union is indicated by the sDataType member.

C Data Type	Member	Description
ULONG	ulDataID	One of the SVE_??? constants from RFStat.h. Indicates the data to obtain.
ULONG	ulIndex1	Optional value. May be required depending on the SVE_??? constant used in ulDataID. For example, if ulDataID is SVE_BRD_CRT_CHNTYPE, then the ulIndex1 value indicates the channel from which to obtain the channel type (0–*n*).
ULONG	ulIndex2	Optional value. May be required depending on the SVE_??? constant used in ulDataID. For example, if ulDataID is SVE_WS_SECONDSUP, then ulIndex2 indicates the WorkServer (0–*n*) from which to obtain the up-time.
ULONG	ulError	Filled in upon return from the RFaxStatusGetValues(). Indicates whether the server could supply the value requested by the combination of ulDataID, ulIndex1, and ulIndex2. See RFAXSTS_??? constants for possible values.
SHORT	sDataType	Indicates the type of data returned. One of the SVE_DATATYPE_??? constants.
UCHAR	ucData	Valid if sDataType = SVE_DATATYPE_UCHAR. Contains value for ulDataID.
UCHAR	szData[256]	Valid if sDataType = SVE_DATATYPE_STRING. Contains value for ulDataID.
SHORT	sData	Valid if sDataType = SVE_DATATYPE_SHORT. Contains value for ulDataID.
LONG	lData	Valid if sDataType = SVE_DATATYPE_LONG. Contains value for ulDataID.
UNSIGNED SHORT	usData	Valid if sDataType = SVE_DATATYPE_USHORT. Contains value for ulDataID.
UNSIGNED LONG	ulData	Valid if sDataType = SVE_DATATYPE_ULONG. Contains value for ulDataID.
UNSIGNED LONG	tTime	Valid if sDataType = SVE_DATATYPE_DTTM. Contains value for ulDataID. Date/time values are stored as ctime values based from 1/1/1970. All times are expressed as GMT, i.e., no local offsets have been applied.
UCHAR	buffer[256]	Valid if sDataType == SVE_DATATYPE_STRUCT. Contains value for ulDataID. Structures are mapped into buffer storage area. Data must be copied or cast out of the buffer space.
UCHAR	aucData[256]	Valid if sDataType = SVE_DATATYPE_UCHARARY. Contains value for ulDataID.
SHORT	asData[128]	Valid if sDataType = SVE_DATATYPE_SHORTARY. Contains value for ulDataID.
LONG	alData[64]	Valid if sDataType = SVE_DATATYPE_LONGARY. Contains value for ulDataID.

C Data Type	Member	Description
UNSIGNED SHORT	ausData[128]	Valid if sDataType = SVE_DATATYPE_USHORTARY. Contains value for ulDataID.
UNSIGNED LONG	aulData[64]	Valid if sDataType = SVE_DATATYPE_ULONGARY. Contains value for ulDataID.
UNSIGNED LONG	atTime[64]	Valid if sDataType = SVE_DATATYPE_DTTMARY. Contains value for ulDataID.
COMPACTCHANNELSTAT1	ccs1	Valid if sDataType = SVE_DATATYPE_ CMPTCHNS1. Contains value for ulDataID.
COMPACTBRDSRVSTAT1	cbss1	Valid if sDataType = SVE_DATATYPE_ CMPTBSS1. Contains value for ulDataID.

SVCINFO_10

The SVCINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of service.
ULONG	ulFlags	One of the SVCFLAG_??? constants.
CHAR	szServiceID[22]	Service ID.
CHAR	szProvider[22]	One of: NONE, D1, E+, TELMI, SCALL, SKYPER, D2, QUIX.
CHAR	szTerminalNumber[64]	Phone number to dial for terminal access (or SMTP server address in the case of SMTP).
CHAR	szServicePassword[32]	Password for UCP or SMTP authentication.
SHORT	sMaxMessageLength	Maximum message length to send.
SHORT	sMaxMessages	Maximum messages to send in one call.
SHORT	sServiceType	One of the SVC_SERVICETYPE_??? constants.
CHAR	szModemConfig[10]	7E1, 8N1, etc.
CHAR	szModemPort[32]	Port COM1, COM2, etc., or the TAPI modem name.
CHAR	szATInitString[64]	Only used with non-TAPI.
CHAR	szDialPrefix[20]	Only used with non-TAPI.
USHORT	usModemFlowControl	0 = None 1 = XON/XOFF 2 = RTS/CTS (only used with non-TAPI)

C Data Type	Member	Description
USHORT	usConnectSpeed	1200, 2400, etc. (only used with non-TAPI).
USHORT	usConnectTimeout	Maximum seconds to wait for connect.
USHORT	usTransactionTimeout	Maximum seconds to complete message send transaction.
CHAR	szSMTPSender[64]	Sender e-mail address.
FAXHANDLE	hConnection	Get more connection details from connections API when the sServiceType is SVC_SERVICETYPE_SMTP.
CHAR	reserved1[1124]	Reserved space.

SVCLISTELEMENT0

The SVCLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of service.
ULONG	ulFlags	One of the SVCFLAG_??? constants.
CHAR	szServiceID[22]	Service ID.
SHORT	sServiceType	One of the SVC_SERVICETYPE_??? constants.
FAXHANDLE	hConnection	Get more connection details from connections API. For now, only when the sServiceType is SVC_SERVICETYPE_SMTP.
CHAR	szReserved [160]	Reserved space (set to 0).

TAGINFO_10

The TAGINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	hOwner	The object this tag is for.
TCHAR	szName[64]	Name for the tag
TCHAR	szValue[512]	Value for the tag.

TOKENINFO_10

The TOKENINFO_10 structure.

C Data Type	Member	Description
CHAR	szToken[1800]	Generally an OAuth access token.
RFAXTIME	expiresAt	"C" time szToken expires in UTC.
CHAR	reserved[126]	Reserved for future use. Initialize to zeros.

TOKENINFO_11

The TOKENINFO_11 structure.

C Data Type	Member	Description
CHAR	szToken[4096]	Generally an OAuth access token.
RFAXTIME	expiresAt	"C" time szToken expires in UTC.

USERINFO_10

The USERINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	Handle	Database handle for particular user.
USHORT	usNumFaxesOwned	Total number of faxes this user owns.
CHAR	groupid[22]	The ID of the group that this user belongs to.
ULONG	userflags	One or more of the USERFLAGS_??? constants.
CHAR	userid[22]	User ID.
CHAR	username[30]	Description or user name.
CHAR	password[12]	Password (always blank when getting the structure from the server).
ULONG	userdidnumber	Deprecated. Personal fax number (DID number). Use szUserdidnumber.
CHAR	fcsmodelname[14]	Default FCS file.

C Data Type	Member	Description
USHORT	notify_type	See NOTIFY_TYPE_???.
CHAR	st_didnum[32]	Default DID (personal fax) number.
CHAR	st_genfaxnum[32]	Default companyfax number.
SHORT	st_callback	Call back requested?
CHAR	st_opernum[32]	Main voice number.
CHAR	st_fromname[60]	Sender name and sent faxes.
CHAR	st_fromphone[32]	Personal voice number.
CHAR	st_to_faxnum[32]	Destination fax number.
CHAR	st_to_contactnum[32]	Destination contact number.
CHAR	st_to_name[60]	Destination name.
CHAR	st_to_company[60]	Destination company name.
CHAR	st_to_citystate[60]	Destination city/state.
CHAR	autoprint_netprint_id[10]	ID of the printer used for auto printing received faxes.
CHAR	default_printer[10]	Default printer for the user.
UCHAR	ucWebImageFormat	See USER_ROUTEFMT_??? constants.
UCHAR	ucLastUpgradedFolderID	The ID of the most recently added folder at the time the folders were last upgraded.
CHAR	reserved1[2]	Reserved space.
ULONG	ulRouteInfoParam	Usage varies with selected route type. For USER_ROUTETYPE_FTP, this indicates the handle of the connection to use.
FAXHANDLE	hWorkflow	Workflow to route to when route type is USER_ROUTETYPE_WORKFLOW.
CHAR	szAutoSentPrinter[10]	ID of the printer used for auto printing sent faxes.
USHORT	usAutoPrintFlags	See USERAPFLAGS_???.
ULONG	ulSubscriberID	Mail box subscriber ID.
ULONG	ulUserFlags2	See USERFLAGS2_???.
CHAR	reserved3[22]	Reserved. Initialize to zero.

C Data Type	Member	Description
UCHAR	ucHighestPriority	Highest priority allowed.
UCHAR	ucDefaultPriority	Default priority. See PRIORITY_???.
CHAR	newfax_notify_user[22]	Alternate notification of this user ID.
UCHAR	routetype	See USER_ROUTETYPE_??? constants.
UCHAR	routefmt	See USER_ROUTEFMT_??? constants.
CHAR	af_phone[32]	Auto forward phone number. See af_user.
USHORT	send_notify	See NOTIFY_SND_???.
USHORT	receive_notify	See NOTIFY_RCV_???.
USHORT	update_interval	Number in seconds before an update on the client.
USHORT	auto_delete	0 = Off 1 = Auto delete after successful send 2 = Always delete
USHORT	default_view_scale	Reserved. Initialize to zero.
USHORT	autoocr_flags	See OCRFLAGS_???.
UCHAR	autoocr_layout	See OCRLAYOUT_???.
UCHAR	autoocr_format	See OCRFORMAT_???.
CHAR	autoocr_ext[4]	File extension to use for OCR output files (i.e., .txt).
UCHAR	af_user	0=if af_phone[] specifies a list of phone numbers. 1=if af_phone[] specifies a list of RightFax users separated by commas. 2=if af_phone[] specifies a RightFax group.
UCHAR	route_deleteafter	Set to 1 if fax should be deleted after route.
ULONG	ulChangeTag	Serial number. Changed if the user's faxes are in any way modified.
CHAR	routeinfo[100]	Routing information field.
CHAR	nwdsname[80]	NetWare distinguished name.
CHAR	folders[256]	List of strings. Each string is followed by an ID byte.
CHAR	szOtherPBID[4][22]	Array or up to four other user IDs to monitor for phonebook entries.

C Data Type	Member	Description
CHAR	szOtherPBPass[4][12]	Passwords associated to szOtherPBID users.
SHORT	sRPO_StartPage	Receive print options. Starting page.
SHORT	sRPO_EndPage	Ending page.
UCHAR	ucRPO_Flags	0 = Print TRX 1 = Staple 2 = Collate 3 = NewAutoPrint (extra1 struct is valid instead of szUserID below)
UCHAR	ucRPO_Res	0 = High 1 = Medium 2 = Low
UCHAR	ucRPO_Size	0 = Default 1 = Letter 2 = Legal 3 = A4
UCHAR	ucRPO_Source	0 = Default 1 = Upper 2 = Lower 3 = Manual
UCHAR	ucRPO_OutputBin	See PRINTOUTPUT_???.
UCHAR	ucRPO_Duplex	See PRINTDUPLEX_???
USHORT	usRPO_SecurityCode	Four digit security code stored as a binary.
UCHAR	ucRPO_Copies	Number of copies.
UCHAR	ucEmailRouteForm	
UCHAR	aucRPO_AcctCode[8]	Flag indicating whether or not to print the cover page.
UCHAR	ucRPO_Priority;	0 = Print TRX 1 = Staple 2 = Collate 3 = NewAutoPrint (extra1 struct is valid instead of szUserID below)

C Data Type	Member	Description
UCHAR	ucRPO_Other	0 = High 1 = Medium 2 = Low
CHAR	szDefaultBI1[16]	Default billing information field 1 string.
CHAR	szDefaultBI2[16]	Default billing information field 2 string.
UCHAR	sid[88]	Security ID.
CHAR	szUserdidnumber[22]	String form of userdidnumber which can represent much larger (longer numbers.
INT64	biSubscriberID	
FAXHANDLE	hFaxNumber	Optional reference to a fax number (via FAXNUMBERLISTELEMENT.handle) corresponding to this users's szUserdidnumber.
ULONG	ulNumFaxesOwned	
CHAR	Reserved[34}	Reserved for future use.

USERINFO_11

The USERINFO_11 structure.

C Data Type	Member	Description
FAXHANDLE	Handle	Database handle for particular user.
USHORT	usNumFaxesOwned	Total number of faxes this user owns.
CHAR	groupid[22]	The ID of the group that this user belongs to.
ULONG	userflags	One or more of the USERFLAGS_??? constants.
CHAR	userid[22]	User ID.
CHAR	username[30]	Description or user name.
CHAR	password[12]	Password (always blank when getting the structure from the server).
ULONG	userdidnumber	Deprecated. Personal fax number (DID number). Use szUserdidnumber.
CHAR	fcsmodelname[14]	Default FCS file.
USHORT	notify_type	See NOTIFY_TYPE_???.

C Data Type	Member	Description
CHAR	st_didnum[32]	Default DID (personal fax) number.
CHAR	st_genfaxnum[32]	Default company fax number.
SHORT	st_callback	Call back requested?
CHAR	st_opernum[32]	Main voice number.
CHAR	st_fromname[60]	Sender name and sent faxes.
CHAR	st_fromphone[32]	Personal voice number.
CHAR	st_to_faxnum[32]	Destination fax number.
CHAR	st_to_contactnum[32]	Destination contact number.
CHAR	st_to_name[60]	Destination name.
CHAR	st_to_company[60]	Destination company name.
CHAR	st_to_citystate[60]	Destination city/state.
CHAR	autoprint_netprint_id[10]	ID of the printer used for auto printing received faxes.
CHAR	default_printer[10]	Default printer for the user.
UCHAR	ucWebImageFormat	See USER_ROUTEFMT_??? constants.
UCHAR	ucLastUpgradedFolderID	The ID of the most recently added folder at the time the folders were last upgraded.
CHAR	reserved1[2]	Reserved space.
ULONG	ulRouteInfoParam	Usage varies with the selected route type. For USER_ROUTETYPE_FTP, this indicates the handle of the connection to use.
FAXHANDLE	hWorkflow	Workflow to which to route when the route type is USER_ROUTETYPE_WORKFLOW.
CHAR	szAutoSentPrinter[10]	ID of the printer used for auto printing sent faxes.
USHORT	usAutoPrintFlags	See USERAPFLAGS_???.
ULONG	ulSubscriberID	Mail box subscriber ID.
ULONG	ulUserFlags2	See USERFLAGS2_???.
CHAR	reserved3[22]	Reserved. Initialize to zero.
UCHAR	ucHighestPriority	Highest priority allowed.

C Data Type	Member	Description
UCHAR	ucDefaultPriority	Default priority. See PRIORITY_???.
CHAR	newfax_notify_user[22]	Alternate notification of this user ID.
UCHAR	routetype	See USER_ROUTETYPE_??? constants.
UCHAR	routefmt	See USER_ROUTEFMT_??? constants.
CHAR	af_phone[32]	Auto forward phone number. See af_user.
USHORT	send_notify	See NOTIFY_SND_???.
USHORT	receive_notify	See NOTIFY_RCV_???.
USHORT	update_interval	Number in seconds before an update on the client.
USHORT	auto_delete	0 = Off 1 = Auto delete after successful send 2 = Always delete
USHORT	default_view_scale	Reserved. Initialize to zero.
USHORT	autoocr_flags	See OCRFLAGS_???.
UCHAR	autoocr_layout	See OCRLAYOUT_???.
UCHAR	autoocr_format	See OCRFORMAT_???.
CHAR	autoocr_ext[4]	File extension to use for OCR output files (i.e., .txt).
UCHAR	af_user	0=if af_phone[] specifies a list of phone numbers. 1=if af_phone[] specifies a list of RightFax users separated by commas. 2=if af_phone[] specifies a RightFax group.
UCHAR	route_deleteafter	Set to 1 if fax should be deleted after route.
ULONG	ulChangeTag	Serial number. Changed if the user's faxes are in any way modified.
CHAR	routeinfo[100]	Routing information field.
CHAR	nwdsname[80]	NetWare distinguished name.
CHAR	folders[256]	List of strings. Each string is followed by an ID byte.
CHAR	szOtherPBID[4][22]	Array or up to 4 other user IDs to monitor for phonebook entries.
CHAR	szOtherPBPass[4][12]	Passwords associated to szOtherPBID users.

C Data Type	Member	Description
SHORT	sRPO_StartPage	Receive print options. Starting page.
SHORT	sRPO_EndPage	Ending page.
UCHAR	ucRPO_Flags	0 = Print TRX 1 = Staple 2 = Collate 3 = NewAutoPrint (extra1 struct is valid instead of szUserID below)
UCHAR	ucRPO_Res	0 = High 1 = Medium 2 = Low
UCHAR	ucRPO_Size	0 = Default 1 = Letter 2 = Legal 3 = A4
UCHAR	ucRPO_Source	0 = Default 1 = Upper 2 = Lower 3 = Manual
UCHAR	ucRPO_OutputBin	See PRINTOUTPUT_???.
UCHAR	ucRPO_Duplex	See PRINTDUPLEX_???.
USHORT	usRPO_SecurityCode	Four digit security code stored as a binary.
UCHAR	ucRPO_Copies	Number of copies.
UCHAR	ucEmailRouteForm	
UCHAR	aucRPO_AcctCode[8]	Flag indicating whether or not to print the cover page.
UCHAR	ucRPO_Priority;	0 = Print TRX 1 = Staple 2 = Collate 3 = NewAutoPrint (extra1 struct is valid instead of szUserID below)

C Data Type	Member	Description
UCHAR	ucRPO_Other	0 = High 1 = Medium 2 = Low
CHAR	szDefaultBI1[16]	Default billing information 1 string.
CHAR	szDefaultBI2[16]	Default billing information 2 string.
UCHAR	sid[88]	Security ID.
CHAR	emailaddr[256]	User's email reply-to address.
CHAR	szSMSServiceID[22]	
CHAR	szMobileAddress[32]	SMS/Mobile phone address for notifications, etc.
CHAR	szNotifyInfo[100]	To differentiate Routing Info from Notification Info. See the routinfo field in USERINFO_10 and USERINFO_11. NOTE: Only valid for v900 servers. For older servers, Notification and routing share the routeinfo field.
CHAR	szUserdidnumber[22]	String form of userdidnumber which can represent much larger (longer) numbers.
UCHAR	ucIncludeThumbnail	Set to 1 to include a thumbnail in notifications.
UCHAR	ucThumbnailFileType	0 = TIFF 1 = PDF
UCHAR	ucThumbnailPages	0 = first page 1 = all pages
UCHAR	ucReserved	Space reserved for future use.
INT64	biSubscriberID	
FAXHANDLE	hFaxNumber	Optional reference to a fax number (via FAXNUMBERLISTELEMENT.handle) corresponding to this users's szUserdidnumber.
ULONG	ulNumFaxesOwned	
UCHAR	ucIncludeThumbnailRecv	Receive notification settings.
UCHAR	ucThumbnailFileTypeRecv	Receive notification settings.
UCHAR	ucThumbnailPagesRecv	Receive notification settings.
UCHAR	ucReserved2	

C Data Type	Member	Description
USHORT	usEDCArchiveFlags	Specify EDC Archive Module. See USER_EDCFLAGS_???.
ULONG	ulDefaultOptions	Holds bitwise options that need to be available across clients. See DEFAULT_OPTS_???.
ULONG	ulUserFlags3	
LONG	lastSignInTime	
CHAR	reserved2[10]	Space reserved for future use.

USERLISTELEMENT0

The USERLISTELEMENT0 structure is used with RFaxGetUserList().

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
CHAR	userid[22]	User ID.
CHAR	username[30]	User name.
ULONG	DIDNumber	Private DID fax number.
CHAR	groupid[22]	ID of the group that the user belongs to.
USHORT	usNumFaxesOwned	Total number of faxes that the user owns.
CHAR	reserved[44]	Reserved. Initialize to zero.

USERLISTELEMENT1

The USERLISTELEMENT1 structure is used with RFaxGetUserList().

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
CHAR	userid[22]	User ID.
CHAR	username[30]	User name.
ULONG	DIDNumber	Private DID fax number.
CHAR	groupid[22]	ID of the group that the user belongs to.

C Data Type	Member	Description
USHORT	usNumFaxesOwned	Total number of faxes that the user owns.
ULONG	ulSubscriberID	Caller subscriber ID.
UCHAR	routetype	See USER_ROUTETYPE_???.
UCHAR	routefmt	See USER_ROUTEFMT_???.
ULONG	userflags	See USERFLAGS_???.
USHORT	notify_type	See NOTIFY_TYPE_???.
UCHAR	ucHighestPriority	Highest priority available to the user.
UCHAR	ucDefaultPriority	Default priority.

USERLISTELEMENT2

The USERLISTELEMENT2 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
CHAR	userid[22]	User ID.
CHAR	username[30]	User name.
ULONG	DIDNumber	Private DID fax number.
CHAR	groupid[22]	ID of the group that the user belongs to.
USHORT	usNumFaxesOwned	Total number of faxes that the user owns. Upper limit is 65K.
ULONG	ulSubscriberID	Caller subscriber ID.
UCHAR	routetype	See USER_ROUTETYPE_???.
UCHAR	routefmt	See USER_ROUTEFMT_???.
ULONG	userflags	See USERFLAGS_???.
USHORT	notify_type	See NOTIFY_TYPE_???.
UCHAR	ucHighestPriority	Highest priority available to the user.
UCHAR	ucDefaultPriority	Default priority.

C Data Type	Member	Description
ULONG	ulUserFlags2	One or more of the USERFLAGS2_???.
ULONG	ulDisplayFlags	See UDFLAG_???.
CHAR	szDIDNumber[22]	Personal fax number (DID number). Replaces the old numeric DIDNumber.
INT64	biSubscriberID	BIG SubscriberID.
ULONG	ulNumFaxesOwned	Total number of faxes that the user owns. Upper limit is 4294967295.
ULONG	ulUserFlags3	
CHAR	reserved[62]	Reserved for future use.

USERMESSAGE

The USERMESSAGE structure contains the desktop notification message about a newly received fax that the user requested to receive from the server. Used in calls to RFaxGetUserMessage().

C Data Type	Member	Description
FAXHANDLE	userHandle	User handle.
LONG	filterTime	
FAXHANDLE	faxHandle	Fax handle.
CHAR	szMessage[256]	The message content.
CHAR	reserved[244]	Space for future use. Initialize to 0.

USERPROPERTIES

The USERPROPERTIES structure is used with the RFaxLoadUserProperties() and RFaxSaveUserProperties() functions.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle.
CHAR	szUserDescription[1024]	User description.
CHAR	szRouteFileName [100]	Adding ability to specify routed filename format.
CHAR	reserved[774]	Reserved. Initialize to zero.

VALIDATEREQUEST

The VALIDATEREQUEST structure.

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.
ULONG	ulRequestType	See REQUEST_???.
FAXHANDLE	handle	If non-Null, handle to fax this event is supposed to validate.
CHAR	billinfo1[16]	Copy of billinfo1 field from fax identified by “handle”.
CHAR	billinfo2[16]	Copy of billinfo2 field from fax identified by “handle”.
CHAR	szUserID[22]	Copy of user ID field from fax identified by “handle”.
CHAR	szPhoneNum[32]	Copy of to_faxnum field from fax identified by “handle”.

VERSIONINFO_0

The VERSIONINFO_0 structure.

C Data Type	Member	Description
LONG	lVersion	RightFax server version 0xMMmm, for example 0x2020 for 20.2.
ULONG	ulBuildDate	0xYYYYMMDD
LONG	lServerVersion	RightFax server version 0xMMmm, for example 0x2020 for 20.2.
LONG	lServerType	Either 0xCF1 or 0xCF2 (or 0x0 if older server).
SHORT	sBFTDisabled	Set to 1 indicating that BFT is disabled on server (only valid if ulBuildDate > 19970805).
SHORT	sEnterprise	1= Enterprise server
SHORT	sDocsOnDemand	1=Docs on Demand is licensed
SHORT	sTeleconnect	1=Teleconnect is licensed
SHORT	sSatellite	1=Satellite server
SHORT	sSBS	1=Small business server
SHORT	sProdEnabled	1=Product enabled.
SHORT	sProdINL	1=Internet link enabled.

C Data Type	Member	Description
SHORT	sProdFilter	1=Server is licensed for the Integration Filter module.
SHORT	sProdNotifier	1=Server is licensed for the Integration Notifier module.
SHORT	sIPPlusConnector	IP Plus Connector enabled.
SHORT	sOCRRouter	OCR Router enabled.
SHORT	sOCRConv	OCR Conversion enabled.
SHORT	sPDFModule	PDF Module enabled.
SHORT	sSerialValid	This should always be 1. The server's serial number is valid.
CHAR	szSerialNumber[36]	Serial number. (Serial numbers have been replaced by SUIDs)
SHORT	sLiscensedChans	Number of licensed channels.
SHORT	sSecureDoc	SecureDocs Module enabled.
ULONG	ulBumpBits	Bits representing licensed features. See BUMP_???.
SHORT	sMaxUsers	Maximum user count.
LONG	lBumpDate	Last time the server's license was updated.
SHORT	fSMTPDisabled	TRUE=SMTP features are disabled.
TCHAR	szSystemIdentifier[40]	System identifier.
TCHAR	szCollectiveServerList[200]	String containing names of servers in collective.
SHORT	sCollectiveServerListCount	Number of servers in collective.
CHAR	szBillDesc1[16]	User-defined description of first billing code.
CHAR	szBillDesc2[16]	User-defined description of second billing code.
CHAR	szAllowedDialingDigits[128]	Read the allowed dialing digits from the server.
SHORT	sLicensedFoipChans	Licensed FoIP channels.
ULONG	ulServerFeatures	See *SERVERFEATURE_??? on page 478*
CHAR	reserved[24]	Reserved.

WORKFLOW_FAX_COUNT

The WORKFLOW_FAX_COUNT structure.

C Data Type	Member	Description
FAXHANDLE	hWorkflow	A handle to a workflow.
USHORT	usFaxCount	The count of faxes currently in the workflow indicated by hWorkflow.

WORKFLOWDELEGATIONINFO_10

The WORKFLOWDELEGATIONINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	hWorkflow	Handle of the workflow.
FAXHANDLE	hDelegate	Handle of the delegated user or group. See usFlags.
USHORT	usFlags	See *WORKFLOWDELEGATIONFLAGS_???* on page 493.

WORKFLOWDELEGATIONLISTELEMENT0

The WORKFLOWDELEGATIONLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	hWorkflow	Handle of the workflow.
FAXHANDLE	hDelegate	Handle of the delegated user or group. See usFlags.
USHORT	usFlags	See *WORKFLOWDELEGATIONFLAGS_???* on page 493.
TCHAR	szDelegateID[22]	ID of the delegate.

WORKFLOWFIELDCHOICEINFO_10

The WORKFLOWFIELDCHOICEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	hField	The workflow field this choice is for.
TCHAR	szValue[64]	Value for this choice.
USHORT	usUIIndex	The index of the value in the UI list.
CHAR	reserved[58]	Reserved.

WORKFLOWFIELDINFO_10

The WORKFLOWFIELDINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
ULONG	ulFlags	Workflow field flags. See *WORKFLOWFIELDFLAG_???* on page 493.
TCHAR	hWorkflow	The workflow this field is for.
TCHAR	szName[32]	Name for the field.
TCHAR	szDefaultValue[64]	Default value for the field.
CHAR	reserved[148]	Reserved.

WORKFLOWINFO_10

The WORKFLOWINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
TCHAR	szName[22]	ID for the workflow mailbox.
TCHAR	szDescription[30]	Description for workflow mailbox.
UINT	reserved1	
FAXHANDLE	hCompletionUser	Optional user to whom to route when this workflow is complete.
LONG	lCompletionTime	Optional maximum expected completion time in seconds. 0 = no time limit.
WorkflowFlags	Flags	Workflow flags. See *WORKFLOWFLAGS_???* on page 493.
WorkflowAction	completionAction	Workflow completion action. See *WORKFLOWACTION_???* on page 492.
FAXHANDLE	hRejectionUser	Optional user to whom to route to when fax is rejected from this workflow.
WorkflowAction	rejectionAction	Workflow rejection action. See *WORKFLOWACTION_???* on page 492.
FAXHANDLE	hExceptionUser	Optional User to route to when fax is excepted from this workflow.
WorkflowAction	exceptionAction	Workflow exception action. See *WORKFLOWACTION_???* on page 492.

C Data Type	Member	Description
LONG	lReminderTime	Optional completion time remaining in seconds at which to send a reminder to notification users. 0 = no time limit.
LONG	lWarningTime	Optional completion time remaining in seconds at which to send a warning to notification users. 0 = no time limit.
FAXHANDLE	hNotificationUser	Optional handle to user to notify for remaining time reminders and warnings.
CHAR	reserved[158]	Reserved.

WORKFLOWINTELLIGENCEINFO_10

The WORKFLOWINTELLIGENCEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of this item.
FAXHANDLE	hWorkflow	Handle of the related workflow.
RFAXTIME	tCreated	Date created.
RFAXTIME	tModified	Date modified.
USHORT	usFlags	Reserved for future use.
TCHAR	szProcessingType[20]	Processing type string.
TCHAR	szCreatedBy[30]	ID of the user that created this item.
TCHAR	szModifiedBy[30]	ID of the user that last modified this item.
TCHAR	szSetupImageName[256]	Reference OCR setup image name.

WORKFLOWLISTELEMENT0

The WORKFLOWLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Database handle for particular list element.
TCHAR	szName[22]	ID for the workflow.

C Data Type	Member	Description
TCHAR	szDescription[30]	Description for workflow.
TCHAR	szCompletionWorkflow[22]	Workflow to enter when this workflow is complete.
TCHAR	szCompletionUser[22]	User to whom to route when this workflow is complete.
USHORT	usWaitingFaxes	Number of faxes in the workflow that are not locked by a user.
ULONG	ulAvgWaitingElapsed	For all locked faxes, the average time in seconds since faxes entered this workflow.
ULONG	ulMaxWaitingElapsed	For all locked faxes, the maximum time in seconds since faxes entered this workflow.
USHORT	usTotalFaxes	The number of faxes in the workflow.
ULONG	ulAvgTotalElapsed	For all faxes, the average time in seconds since faxes entered this workflow.
ULONG	ulMaxTotalElapsed	For all faxes, the maximum time in seconds since faxes entered this workflow.
WorkflowFlags	flags	Workflow flags. *WORKFLOWFLAGS_???* on page 493.
WorkflowAction	completionAction	Workflow completion action. See *WORKFLOWACTION_???* on page 492.
CHAR	reserved[130]	Reserved.

WORKFLOWRELATIONLISTELEMENT0

The WORKFLOWRELATIONLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	hWorkflowOwner	Handle of the workflow.
FAXHANDLE	hWorkflowTarget	Handle of the workflow being referred to.
USHORT	usType	Type of relationship. See *WORKFLOWRELATIONTYPE_???* on page 494.
TCHAR	szTargetID[22]	ID of target workflow.

WORKFLOWVALUEINFO_10

The WORKFLOWVALUEINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	hFax	The document this value is for.
TCHAR	szWorkflow[32]	The workflow this value is part of.
TCHAR	szField[32]	The field that this value corresponds to.
TCHAR	szValue[508]	The value.
ULONG	ulFlags	See *WORKFLOWVALUEFLAGS_???* on page 494.

WORKREQCOUNTS_0

The WORKREQCOUNTS_0 structure

C Data Type	Member	Description
ULONG	aulCountReqTypes[64]	An array of counts indexed by the bit number of the request type. The number of the bit that identifies a particular type of work request is indexed into this array to determine how many of these work requests exist.

WORKREQUEST

The structures AUTOPRINTREQ, MSMAIL1REQ, CCMAIL1REQ, WPOMAIL1REQ, and OCRREQ are all used with the WORKREQUEST union. The WORKREQUEST is a structure posted by the fax server when it requires “work” to be done. These requests are stored in a FIFO queue, and any process can query the server for the next event. Such queries can contain a mask of request types to filter out requests for processes it is not able to accomplish. For example, an email gateway only requests event types of email routing and email messages, but it would not query for request types of fax printing because it doesn’t do fax printing.

If multiple processes request the same event types, then events are dealt out on a first-come first-serve basis. If you write a process to handle archive events, you need to make sure that no other process, i.e., one of the RightFax WorkServers, is attempting to also service such events. In other words, if your process is not getting all the events you expected, another process may be taking them from the queue.

The second argument to the RFaxGetGenericEvent(), “dwMessageTypes,” is a 32-bit, unsigned long that is mapped into bit fields. The REQUEST_??? constants can be used to construct the dwMessageTypes argument by bitwise ORing together the event types to be requested. RFaxGetGenericEvent() will return 0 if an event was retrieved, else it will return RFAXERR_NOTFOUND (2). It is recommended that if a process calls RFaxGetGenericEvent() and 2 is returned, that the process sleep for five or more seconds before checking again. If an event is returned and acted upon, RFaxGetGenericEvent() can be called immediately after to possibly handle another event.

RFaxGetArchiveEvent() and RFaxGetMessageEvent() are simply wrappers around the RFaxGetGenericEvent() function. They are useful when the calling language cannot deal with the complex WORKREQUEST structure.

The WORKREQUEST structure.

C Data Type	Member	Description
ULONG	flags	Reserved. Initialize to zero.
ULONG	ulRequestType	See REQUEST_???.
FAXHANDLE	handle	Handle of the work request.
CHAR	d.convert.sourcedir[128]	Source folder.
CHAR	d.convert.sourcefile[14]	Source file name.
CHAR	d.convert.outputdir[128]	Output folder.
CHAR	d.convert.outputfile[14]	Output file name.
FAXBOOL	d.convert.finemode	True = Convert in fine mode.
FAXBOOL	d.convert.fCoverSheet	True = Convert cover sheet.
CHAR	d.convert.szUserID[22]	Owner's user ID.
FAXHANDLE	hRequiredLibDoc	Handle of the library document when a required library document has been selected by the admin.
USHORT	usMaxPages	Maximum number of pages that the user is allowed to send in one fax.
UCHAR	routefmt	
UCHAR	reserved1	
SHORT	d.merge.formnum	Used to accomplish form overlays.
CHAR	d.merge.szUserID[22]	User ID.
SHORT	d.merge.sMergeOpts	0=Normal Merge 1=Form Fixup w/Blank out TTI 2= Form Fixup w/Cut out TTI
SHORT	d.merge.sFormHeight	Only valid on sMergeOpts > 0; 0=Don't Adjust 1=Letter 2= Legal 3=A4
AUTOPRINTREQ	d.print	Used to autoprint received or sent faxes.

C Data Type	Member	Description
MSMAIL1REQ	d.msmail1	Used by MS-Mail gateway.
CCMAIL1REQ	d.ccmail1	Used by cc:Mail gateway.
WPOMAIL1REQ	d.wpomail1	Used by GroupWise gateway.
GENMAIL1REQ	d.genmail1	Used to queue notifications and inbound fax routing requests to gateways, e.g., SMTP, Lotus, Exchange, SAP, etc.
GENMAIL2REQ	genmail2	See theGENMAIL2REQ structure.
FILERTE1REQ	filerte1	See the FILERTE1REQ structure.
FAXBOOL	d.mail1.fDeleteFax	Delete after routing.
SHORT	d.mail1.sMsgType	See MSGTYPE_???.
CHAR	d.mail1.szMsgText[200]	Message to send to mail.
CHAR	d.mail1.szUserID[22]	User ID.
FAXBOOL	d.archivefax.fDeleteFax	Delete after archiving.
CHAR	d.archivefax.szUserID[22]	Archiver's user ID.
USHORT	d.archivefax.usNotifyChannel	Reserved. Initialize to zero.
FAXBOOL	d.archivefax.fProdFax	Did this fax originate from the Production environment?
SHORT	sAttempt	Limit the number of attempts to notify.
FAXBOOL	d.filesave1.fDeleteFax	Used when received faxes are being routed to a folder.
USHORT	d.filesave1.usFileFormat	One of the IMAGEFMT_??? constants.
CHAR	d.filesave1.szFilePath[72]	Route to network folder path.
AUTOPRINTREQ	d.filesave1.print	AUTOPRINTREQ structure.
CHAR	d.filesave1.szUserID[22]	User ID.
CHAR	d.makefcs.szUserID[22]	User ID.
CHAR	d.validate.billinfo1[16]	Used when external billing code validation is enabled.
CHAR	d.validate.billinfo2[16]	Bill code 2.
CHAR	d.validate.szUserID[22]	User ID for validation.

C Data Type	Member	Description
CHAR	d.validate.szPhoneNum[32]	To phone number.
OCRREQ	d.ocr	OCRREQ structure.
SHORT	d.filedelete.sourcedir[12]	Source folder to delete.
CHAR	d.filedelete.sourcefile[12][14]	Source files to delete. Can specify up to 14 files at one time.
CHAR	d.xroute.szUserID[22]	User ID for routing with Interconnect.
CHAR	d.xroute.szAltAddress[100]	
FAXBOOL	d.xroute.fDeleteFax	
SHORT	d.xroute.sRetryCount	
ULONG	d.xroute.ulWhenToTry	
FAXBOOL	d.cxreply.fDeleteFax	Delete fax after routing with CallXpress integrations.
SHORT	d.cxreply.sMsgType	
CHAR	d.cxreply.szUserID[22]	
USHORT	d.cxreply.usNotifyType	
SHORT	d.cxreply.sRetryCount	
ULONG	d.cxreply.ulWhenToTry	
AUTOPRINTREQ	d.cxreply.print	
USHORT	d.cxreply.usReplyType	0 = One-Call Send Ack, 1 = One-Call Receive Ack
CHAR	d.cxreply.szMessageID[16]	
CHAR	d.cxreply.szMsgText[200]	
BYTE	d.bRequestData[476]	Reserved. Initialize to zero.
LONG	lRequestTime	ctime of request event.
CHAR	szRequestKey1[10]	Reserved. Initialize to zero.

WPOMAIL1REQ

The WPOMAIL1REQ structure. See WORKREQUEST for more information on this structure

C Data Type	Member	Description
FAXBOOL	fDeleteFax	Delete fax after routing.
SHORT	sMsgType	One of the MSGTYPE_??? constants.
CHAR	szMsgText[200]	Message text.
CHAR	szUserID[22]	Owner's user ID.
CHAR	szMailAddr[60]	Routing info.
AUTOPRINTREQ	print	AUTOPRINTREQ structure.
USHORT	usFileFormat	One of the IMAGEFMT_??? constants.
SHORT	sOCRError	If the fax was OCRed, this will indicated whether it was successful or not.

Appendix B: Constants

ADMIN_PAGERNOTIFY_???

Constants for the following fields: EXTENSIONINFO_10.d.AdminPagerNotify.ulTypes1 and EXTENSIONINFO_10.d.AdminPagerNotify.ulTypes2.

Constant	Value	Description
ADMIN_PAGERNOTIFY_SERVICEFAILURE	1	Some service has failed.
ADMIN_PAGERNOTIFY_CRITICALLOWDISK	2	Disk space is getting low.
ADMIN_PAGERNOTIFY_LOWDISK	3	Disk space is getting too low. Certain operations suspended.
ADMIN_PAGERNOTIFY_QUEUEFULL	4	Fax server internal queue has reached 90% utilization.
ADMIN_PAGERNOTIFY_SENDQUEUEDEEP	5	Deprecated. Use alerting instead. Boardserver send queue too deep.
ADMIN_PAGERNOTIFY_BADLINE	6	Probable bad line detected.
ADMIN_PAGERNOTIFY_HUNGPORT	7	Hung port (off-hook too long).
ADMIN_PAGERNOTIFY_HEARTBEAT	8	Send status every X minutes.
ADMIN_PAGERNOTIFY_IMPROPERDOWN	9	Server was shutdown improperly.
ADMIN_PAGERNOTIFY_BRDSRVDOWN	10	A boardserver service has failed, but at least one is still running.
ADMIN_PAGERNOTIFY_ALLBRDSRVDOWN	11	A boardserver service has failed, and there are no others running.
ADMIN_PAGERNOTIFY_T1FAILURE	12	Probable T1 Failure detected.
ADMIN_PAGERNOTIFY_TOOFEWDIGITS	13	Received fax with too few DID digits.
ADMIN_PAGERNOTIFY_FAILEDFAXES	15	Failed received fax transmission.

ARCHIVEFAXOPT_???

Constants for the dwOptions of RFaxArchiveFax.

Constant	Value	Description
ARCHIVEFAXOPT_NONE	0x0000	

Constant	Value	Description
ARCHIVEFAXOPT_EDC	0x0001	
ARCHIVEFAXOPT_FAXARCHIVE	0x0002	

AUDITS_OPTIONS_???

Constants for usOptions of RFaxGetAuditsList().

Constant	Value	Description
AUDITS_OPTION_NONE	0x0000	
AUDITS_OPTION_CONSOLIDATE	0x0001	Consolidate entries from the same application instance and user.

AUDITS_USERCONNECTIONORDER_???

Constants for parameter usOrder in calls to RFaxGetAuditsList().

Constant	Value	Description
AUDITS_USERCONNECTIONORDER_DEFAULT	0	
AUDITS_USERCONNECTIONORDER_OWNER	1	
AUDITS_USERCONNECTIONORDER_TIME	2	
AUDITS_USERCONNECTIONORDER_MODULE	3	
AUDITS_USERCONNECTIONORDER_CLIENT	4	

BILLCODE_LOAD_???

Constants for the sWhich argument of function RFaxLoadBillCode().

Constant	Value	Description
BILLCODE_LOAD_HANDLE	0	Load by handle.
BILLCODE_LOAD_BOTH	1	Load by BI1 and BI2.
BILLCODE_LOAD_BI1	2	Load by BI1.
BILLCODE_LOAD_BI2	3	Load by BI2.

Constant	Value	Description
BILLCODE_LOAD_EITHER	4	Load by either.

BILLSEARCHKEY_???

Constants for the usKeyField argument of functions RFaxGetBillCodeList() and RFaxGetBillCodeList2().

Constant	Value	Description
BILLSEARCHKEY_CODE1	0	Billinfo_1.
BILLSEARCHKEY_CODE2	1	Billinfo_2.
BILLSEARCHKEY_DESC	2	Description.

BUMP_???

Bump bit field flags for masking required bits from SERVERINFO2.ulBumpBits and VERSIONINFO_0.ulBumpBits.

Constant	Value	Description
BUMP_SBS	0x00000001	Small Business Server.
BUMP_SATELLITE	0x00000002	Satellite Server.
BUMP_PRODUCTION	0x00000004	Enterprise Integration.
BUMP_IPPLUS	0x00000008	RightFax IP Plus.
BUMP_TELECONNECT	0x00000010	TeleConnect.
BUMP_DOCSONDEMAND	0x00000020	Docs-on-Demand.
BUMP_ENTERPRISE	0x00000040	Enterprise Server.
BUMP_PDFMODULE	0x00000080	PDF Module.
BUMP_PRODFILTER	0x00000100	Integration Filter.
BUMP_PRODNOTIFY	0x00000200	Integration Notify.
BUMP_SECUREDOC	0x00000400	SecureDocs.
BUMP_OCRCONVERTER	0x00000800	OCR Converter.
BUMP_INTERNETLINK	0x00001000	InternetLink.

Constant	Value	Description
BUMP_OCRROUTER	0x00002000	OCR Router.
BUMP_EXCHANGEGATE	0x00004000	Exchange Gateway.
BUMP_NOTESGATE	0x00008000	Notes Gateway.
BUMP_PRODLITE	0x00010000	Small Business Integration.
BUMP_WEBCLIENT	0x00020000	FaxUtil Web
BUMP_EDC	0x00040000	External Doc Connector.
BUMP_SNMP	0x00080000	SNMP Alerting Module.
BUMP_SAP	0x00100000	Connector for SAP.
BUMP_DEALER	0x00200000	NFR server.
BUMP_FINANCIAL	0x00400000	Financial server.
BUMP_PRODLITE2	0x00800000	
BUMP_ENCRYPTION	0x01000000	
BUMP_ALERTING	0x02000000	Alerting & Monitoring.
BUMP_SMS	0x04000000	SMS.
BUMP_DND	0x08000000	DND.
BUMP_COLLECTIVE	0x10000000	Allow multiple servers in the same database to share resources and work load.
BUMP_BARCODE	0x20000000	Barcode.
BUMP_SPDF	0x40000000	Searchable PDF.
BUMP_EDOCSCONNECTOR	0x80000000	EDocs connector.
BUMP_ANALYTICS	0x0000000100000000ULL	
BUMP_IMAGELOCKER	0x0000000200000000ULL	Image folder in SQL.
BUMP_FAXARCHIVE	0x0000000400000000ULL	RightFax Fax Archive.
BUMP_EXCLUSIVEGROUPS	0x0000000800000000ULL	Exclusive groups.
BUMP_SECUREFOIP	0x0000001000000000ULL	Secure FOIP channels.
BUMP_INTELLIFLOWS	0x0000002000000000ULL	RightFax Intelligent Workflows.

COLLECTIVELISTOPT_???

Constant	Value	Description
COLLECTIVELISTOPT_UPSERVERS	0x0000	Servers thought to be up and responsive.
COLLECTIVELISTOPT_DOWNSERVERS	0x0001	Servers thought to be down/unresponsive.
COLLECTIVELISTOPT_ALLSERVERS	0x0002	All servers.
COLLECTIVELISTOPT_NOREFRESH	0x0004	Don't refresh the list. Typically, this data is already known by the client. Use this flag to not call the server and instead use what is already known.

COMMPROTOCOL_???

Constants for the usCommProtocol argument of functions RFaxCreateServerHandle() and RFaxCreateServerHandle2().

Constant	Value	Description
COMMPROTOCOL_NONE	0	
COMMPROTOCOL_NAMEDPIPES	1	Must be used with RFWIN32.dll and RFOS2.DLL.
COMMPROTOCOL_IPXOS2	2	Only available for OS/2 fax servers. (deprecated)
COMMPROTOCOL_SPX	3	Uses RPCs over SPX.
COMMPROTOCOL_TCPIP	4	Uses RPCs over TCP/IP.
COMMPROTOCOL_IPX	5	Uses RPCs over IPX.
COMMPROTOCOL_SEC_TCPIP	6	Uses PRCs over secure TCP/IP.
COMMPROTOCOL_SEC_SPX	7	Uses RPCs over secure SPX.
COMMPROTOCOL_HTTP	8	Uses RPCs over HTTP.
COMMPROTOCOL_NAMEDPIPES_SINGLE	9	Connect using NamedPipes, but do not connect to other nodes in a collective.
COMMPROTOCOL_MAX	10	Maximum allowed COMM Protocol value.

CONNECTION_POP3_TYPEFLAGS_???

Constants for the ulTypeFlags field of *CONNECTIONINFO_10* on page 311 only when ulType is CONNECTIONTYPE_POP3.

Constant	Value	Description
CONNECTION_POP3_TYPEFLAGS_NONE	0x00000000	
CONNECTION_POP3_TYPEFLAGS_PASSWORD_CLEARONLY	0x00000001	Send password as clear text only. Absence of both PASSWORD_CLEARONLY and PASSWORD_ENCRYPTEDONLY uses automatic determination of how to send the password.
CONNECTION_POP3_TYPEFLAGS_PASSWORD_APOPONLY	0x00000002	Send password as encrypted text only. Absence of both PASSWORD_CLEARONLY and PASSWORD_APOPONLY uses automatic determination of how to send the password.

CONNECTIONFLAGS_???

Constants for the ulFlags field of *CONNECTIONINFO_10* on page 311

Constant	Value	Description
CONNECTIONFLAGS_NONE	0x00000000	
CONNECTIONFLAGS_OAUTH	0x00000001	Use user name and token rather than user name and password.

CONNECTIONTYPE_???

Constants for the ulType field of *CONNECTIONINFO_10* on page 311

Constant	Value	Description
CONNECTIONTYPE_NONE	0	
CONNECTIONTYPE_FTP	1	
CONNECTIONTYPE_FTPS	2	
CONNECTIONTYPE_SFTP	3	
CONNECTIONTYPE_SMTP	4	Connect plain text and automatically upgrade to TLS if supported by server.
CONNECTIONTYPE_SMTPTLS	5	Connect and upgrade to TLS. Fails if not supported by the server.
CONNECTIONTYPE_POP3	6	Connect plain text and automatically upgrade to TLS if supported by server.
CONNECTIONTYPE_POP3S	7	Connect and use SSL immediately. Will not connect to a non-encrypted port.

Masks for CONNECTIONTYPE

CONNECTIONTYPE_MASK_ALL (~MAKE_CONNECTIONTYPE_MASK(CONNECTIONTYPE_NONE))

CONNECTIONTYPE_MASK_FTP (MAKE_CONNECTIONTYPE_MASK(CONNECTIONTYPE_FTP) | MAKE_CONNECTIONTYPE_MASK (CONNECTIONTYPE_FTPS) | MAKE_CONNECTIONTYPE_MASK(CONNECTIONTYPE_SFTP))

CONNECTIONTYPE_MASK_SMTP (MAKE_CONNECTIONTYPE_MASK(CONNECTIONTYPE_SMTP) | MAKE_CONNECTIONTYPE_MASK (CONNECTIONTYPE_SMTPTLS))

CONNECTIONTYPE_MASK_POP (MAKE_CONNECTIONTYPE_MASK(CONNECTIONTYPE_POP3) | MAKE_CONNECTIONTYPE_MASK (CONNECTIONTYPE_POP3S))

CONTENTTYPE_???

Constants for the ucPreferredContentType field of FAXINFO_11.

Constant	Value	Description
CONTENTTYPE_G3	0	G3
CONTENTTYPE_TEXT	1	Text
CONTENTTYPE_PDF	2	PDF

COUNT_???

Constants for the usObjectType argument of function RFaxGetObjectCount().

Constant	Value	Description
COUNT_FAXES	0	Count the number of faxes.
COUNT_PHONES	1	Number of phonebook entries.
COUNT_USERS	2	Number of users.
COUNT_GROUPS	3	Number of groups.
COUNT_PRINTERS	4	Number of printers.
COUNT_FORMS	5	Number of forms.
COUNT_SIGS	6	Number of signatures.
COUNT_CODES	7	Number of billing codes on the server.
COUNT_GRAPHICS	8	Obsolete.
COUNT_LIBDOCS	8	Number of library documents stored on the server.
COUNT_HISTORY	9	Number of history items attached to a fax.

Constant	Value	Description
COUNT_DELETED_FAXES	10	Number of faxes that have been deleted.
COUNT_NEW_FAXES	11	Unviewed and unprinted faxes (no BFTs).
COUNT_USER_EXTENSIONS	12	Depending on the setting of RFaxGetObjectCount()returns the number of database extensions for a specific user or all users.
COUNT_RESERVED1	13	Reserved for use by another API.
COUNT_PENDING_CONV_RESULTS	14	Count of the pending conversion results. These are the results of conversions that have already taken place but have not yet been processed by the workflow.
COUNT_SERVERQUEUESIZE	15	Count of the number of events in the server's event queue.
COUNT_INQUEUEDOCS	16	Number of documents in queue on the server(s).
COUNT_DTSCHEDULED	17	Number of documents scheduled on DocTransport(s).
COUNT_CONVERSIONEVENTS	18	Number of documents in the conversion process.
COUNT_DTDONE	19	Number of documents sent on DocTransport(s) that have not yet been noticed by the faxserv.
COUNT_PENDINGSENDS	20	Number of documents waiting to be sent.
COUNT_PENDINGCHECKXMIT	21	Number of documents whose send status needs to be checked.
COUNT_SYSTEMMESSAGES	22	Number of system messages in the database.
COUNT_QUICKKICKS	23	Number of quick kicks in the shared memory segment.
COUNT_USERS_SEARCHTOKEN	24	Number of users whose username contains the specified search token.
COUNT_HIDDENLDGROUPS	25	Number of groups hidden from a given library doc.
COUNT_HIDDENGRAPHICS	26	Number of library docs hidden from a given group.
COUNT_AUDITS	27	Reserved for internal use.
COUNT_FAXNUMBERS	28	Number of fax numbers.
COUNT_DOCUMENTS_INPROGRESS	29	Number of faxes pending transmission, such as faxes waiting for or in conversion, scheduled, waiting to be sent, and sending.
COUNT_PHONEBOOKS	30	Number of phonebooks.
COUNT_DEVICETYPES	31	Number of device types.
COUNT_COVERSHEETS	32	Number of cover sheets.

Constant	Value	Description
COUNT_WORKFLOW_FAXES	33	Number of faxes in each workflow for a user.
COUNT_AVAILABLE_OAUTHPROVIDERS	34	

COVERSHEETFLAGS_???

Constants for the ulFlags fields of COVERSHEETINFO_10.

Constant	Value	Description
COVERSHEETFLAGS_UNSUPPORTEDFORMAT	0x00000001	

CREATECVLFLAG_???

Constants for the ulFlags argument of function RFaxCreateCVL().

Constant	Value	Description
CREATECVLFLAG_UPLOADFILE	0x00000001	Uploads the CVL and deletes the local copy. If fax is not saved, make sure to delete this file on the server.
CREATECVLFLAG_LOCALNAMEISCOMPLETE	0x00000002	The specified local file name is the complete file spec for the CVL file.

CREATELSTFLAG_???

Constants for the ulLSTCreationFlags argument of function RFaxCreateLST().

Constant	Value	Description
CREATELSTFLAG_UPLOADFILE	0x00000001	Uploads the LST and deletes local copy. If fax is not saved, make sure to delete this file on the server.
CREATELSTFLAG_LOCALNAMEISCOMPLETE	0x00000002	The specified local file name is the complete file spec for the LST file.
CREATELSTFLAG_KEEPFIRSTUNIQUEID	0x00000004	Sets first recipient's szUniqueID to the unique_ID in FAXINFO_10 or FAXINFO_11. (addional recipients get new unique ID's).

CVL_ATTACHFLAG_???

Constants for the ulFlags field of CVL_ATTACHITEM.

Constant	Value	Description
CVL_ATTACHFLAG_NATIVE_INLINE	0x00000001	Inline disposition.
CVL_ATTACHFLAG_NATIVE_BODY	0x00000002	Native doc for use in body of the message.
CVL_ATTACHFLAG_NATIVE_DOC	0x00000004	Simple native document.
CVL_ATTACHFLAG_NATIVE_MASK	0x00000007	All Native doc specific flags.
CVL_ATTACHFLAG_DELETE	0x00000008	Delete attachment after upload.

CVL_ATTACHTYPE_???

Constants for the type field of CVL_ATTACHITEM.

Constant	Value	Description
CVL_ATTACHTYPE_FILE	0	Attachment is a file.
CVL_ATTACHTYPE_FAX	1	Attachment is a fax.
CVL_ATTACHTYPE_LIBDOC	2	Attachment is a library document (TIF file)
CVL_ATTACHTYPE_NOTE	3	Attachment is a note file.
CVL_ATTACHTYPE_SCANNED_IMAGE	4	Attachment is a scanned image file.
CVL_ATTACHTYPE_DELETED	5	Attachment has been deleted.
CVL_ATTACHTYPE_CMDNATIVEFILES	6	Attachment is a retained native document file.

DEFAULT_OPTS_???

Constants for the ulDefaultOptions fields of USERINFO_11 and GROUPINFO_10.

Constant	Value	Description
DEFAULT_OPTS_CREATEPDF	0x00000001	Create PDF for email check box option.
DEFAULT_OPTS_DOCTYPE_NONE	0x00000002	Se the default conversion bias to None (FAXFLAG2_DOCTYPE_TEXT.)
DEFAULT_OPTS_DOCTYPE_IMAGES	0x00000004	Set default conversion bias to Optimize for Images (FAXFLAG2_DOCTYPE_PANDI.)
DEFAULT_OPTS_DOCTYPE_TEXT	0x00000008	Set default conversion bias to Optimize for Text (FAXFLAG2_DOCTYPE_LIGHT.)

Constant	Value	Description
DEFAULT_OPTS_DOCTYPE_HIGHCON	0x00000010	Set default conversion bias to High Contrast (FAXFLAG2_DOCTYPE_LIGHT and FAXFLAG2_DOCTYPE_PANDI.)
DEFAULT_OPTS_INCLUDEWEBDELIVERYLINK	0x00000020	Include a Web Delivery URI with inbound routed faxes.

DELETEAUDITS_OPTIONS_???

Constants for usOptions parameter of RFaxDeleteAudit().

Constant	Value	Description
DELETEAUDITS_OPTION_NONE	0x0000	
DELETEAUDITS_OPTION_CONSOLIDATE	0x0001	Delete all entries from the same application instance and user.
DELETEAUDITS_OPTION_DELETEALL	0x0002	Delete all entries from the database.

DELETEFAX_FLAGS_???

Constants for calls to function RFaxDeleteFax2().

Constant	Value	Description
DELETEFAX_FLAGS_NOCHECK	0x00000001	Deleting a fax implies that all of the flags on the fax will be checked and verified before it is deleted. On a received fax, if any required fields are not filled in, the fax will not be deleted.
DELETEFAX_FLAGS_NOFORWARD	0x00000002	Forward has been cancelled.
DELETEFAX_FLAGS_FAXESBYFOLDER	0x00000004	Deletes all faxes in the folder regardless of required fields.

DELETEUSER_FLAGS_???

Constants for calls to function RFaxDeleteUser2().

Constant	Value	Description
DELETEUSER_FLAGS_NOEMPTY	0x00000001	Do not delete user if user still has faxes, contacts, or distribution lists. In this case, RFaxDeleteUser2() will return FDB_ERR_USERNOTEMPTY.

DEVICE_ORDER_???

Constants for parameter usOrder in calls to RFaxGetPagedDeviceList().

Constant	Value	Description
DEVICE_ORDER_DEFAULT	0	
DEVICE_ORDER_NETWORKADDRESS	1	
DEVICE_ORDER_DESCRIPTION	2	
DEVICE_ORDER_LICENSETYPE	3	
DEVICE_ORDER_ISREGISTERED	4	

DLGFLAGS_???

Constants for the ulFlags fields of DELEGATEINFO_10 and DELEGATELISTELEMENT0.

Constant	Value	Description
DLGFLAGS_GROUPID	0x00000001	Delegatee is a group.
DLGFLAGS_PERMSSET	0x00000002	Reserved to indicate permissions have been upgraded via 8.0 client.
DLGFLAGS_MARKEDDEL	0x00000004	
DLGFLAGS_DISABLED	0x00000008	Delegate is currently a disabled user.
DLGFLAGS_RESERVED0	0x20000000	Reserved.
DLGFLAGS_RESERVED1	0x40000000	Reserved.
DLGFLAGS_RESERVED2	0x80000000	Reserved.

DLGPERM_???

These constants can be masked against the ulReadPermissions, .ulWritePermissions, .ulDeletePermissions, and ulMiscPermission permissions of DELEGATEINFO_10to determine which are restrictive or non-restrictive.

Constant	Value	Description
DLGPERM_RESTRICTIVE	0x00000080	Masks restrictive delegate permissions.
DLGPERM_NONRESTRICTIVE	0x00000100	Masks nonrestrictive delegate permissions.

DLGPERM_D_???

These constants represent the delete permissions that a delegator can grant to a delegate. These are flag values for the ulDeletePermissions fields of DELEGATEINFO_10 and DELEGATELISTELEMENT0.

Constant	Value	Description
DLGPERM_D_CANDELETEFAXES	0x00000001	The delegate can delete the delegator's faxes (non-restrictive).
DLGPERM_D_CANDELETEFOLDERS	0x00000002	The delegate can delete the delegator's folders (if empty) (non-restrictive).
DLGPERM_D_CANDELETEPHONEITEMS	0x00000004	The delegate can delete the delegator's phonebooks and phonebook entries (non-restrictive).

DLGPERM_M_???

These constants represent miscellaneous permissions that a delegator can grant to a delegate. These are flag values for the ulMiscPermissions fields of DELEGATEINFO_10 and DELEGATELISTELEMENT0.

Constant	Value	Description
DLGPERM_M_CANAPPROVEFAXES	0x00000001	The delegate can approve delegator's faxes (non-restrictive).
DLGPERM_M_CANMODIFYUSEROPTIONS	0x00000002	The delegate can change the delegator's FaxUtil user options (non-restrictive).
DLGPERM_M_CANMODIFYDELEGATES	0x00000004	The delegate can change the delegator's list of delegates (non-restrictive).
DLGPERM_M_CANTOGGLEREFUSEFAXDISTRIB	0x00000008	The delegate can turn the Refuse Fax Distributions option on and off (non-restrictive).
DLGPERM_M_CANSERVERPRINT	0x00000010	Deleted. The delegate can print to network printers configured on the fax server.
DLGPERM_M_FAXREQUIRESAPPROVAL	0x10000000	Faxes sent by the delegate from the delegator's mailbox must be approved (restrictive).

DLGPERM_R_???

These constants represent the read permissions that a delegator can grant to a delegate. These are flag values for the ulReadPermissions fields of DELEGATEINFO_10 and DELEGATELISTELEMENT0.

Constant	Value	Description
DLGPERM_R_CANVIEWFAX	0x00000001	The delegate can view delegator's faxes (non-restrictive).
DLGPERM_R_CANVIEWFIRSTPAGE	0x00000002	The delegate can view the first page of delegator's faxes (non-restrictive).

Constant	Value	Description
DLGPERM_R_CANPRINTFAXES	0x00000004	The delegate can print delegator's faxes (non-restrictive).
DLGPERM_R_CANVIEWFAXHISTORY	0x00000008	The delegate can view the fax history of delegator's faxes (non-restrictive).
DLGPERM_R_CANBROWSEALLFOLDERS	0x00000010	The delegate can view faxes in all of the delegator's folders (otherwise Main folder only) (non-restrictive).
DLGPERM_R_CANUSEPHONEENTRIES	0x00000020	The delegate can use delegator's phonebook entries (non-restrictive).
DLGPERM_R_CANEXPORTIMAGES	0x00000040	The delegate can export images (non-restrictive).
DLGPERM_R_CANMAILIMAGES	0x00000080	The delegate can e-mail images (non-restrictive).

DLGPERM_W_???

These constants represent the read permissions that a delegator can grant to a delegate. These are flag values for the ulWritePermissions fields of DELEGATEINFO_10 and DELEGATELISTELEMENT0.

Constant	Value	Description
DLGPERM_W_CANCREATEFAXES	0x00000001	The delegate can create and send new faxes from the delegator's mailbox (non-restrictive).
DLGPERM_W_CANEDITFAXES	0x00000002	The delegate can edit the delegator's faxes (non-restrictive).
DLGPERM_W_CANUPDATEFAXSTATUS	0x00000004	The delegate can update the delegator's fax status (non-restrictive).
DLGPERM_W_CANFORWARDFAXES	0x00000008	The delegate can forward faxes. (Was CANFORWARDROUTEFAXES.)
DLGPERM_W_CANFORWARDROUTEFAXES	0x00000008	Deprecated. The delegate can forward and route faxes from the delegator's mailbox (non-restrictive).
DLGPERM_W_CANOCRFAXES	0x00000010	The delegate can perform OCR (optical character recognition) on the delegator's faxes (non-restrictive).
DLGPERM_W_CANCREATEFOLDERS	0x00000020	The delegate can create new folders in the delegator's fax mailbox (non-restrictive).
DLGPERM_W_CANRENAMEFOLDERS	0x00000040	The delegate can re-name folders in the delegator's fax mailbox (non-restrictive).
DLGPERM_W_CANCREATEPHONEITEMS	0x00000080	The delegate can create new phonebook entries in the delegator's fax phonebook (non-restrictive).
DLGPERM_W_CANEDITPHONEITEMS	0x00000100	The delegate can edit phonebook entries in the delegator's fax phonebook (non-restrictive).
DLGPERM_W_CANANNOTATEFAXES	0x00000200	The delegate can add annotations to faxes in the delegator's fax mailbox (non-restrictive).

Constant	Value	Description
DLGPERM_W_CANMOVEFAXES	0x00000400	The delegate can move faxes between folders in the delegator's fax mailbox (non-restrictive).
DLGPERM_W_CANROUTEFAXES	0x00000800	The delegate can route faxes.
DLGPERM_W_CANUSEPERSONALSTAMP	0x00001000	The delegate can use the delegator's personal stamp.

EXTENSIONTYPE_???

Constants for the ulType field of EXTENSIONINFO_10 and the ulExtensionType argument of function RFaxFindExtension().

Constant	Value	Description
EXTENSIONTYPE_PHONEGRP	1	The phonegrp member of EXTENSIONINFO_10 is valid.
EXTENSIONTYPE_USERFOLDERS	2	The userfolder member of EXTENSIONINFO_10 is valid.
EXTENSIONTYPE_SMSDATA	3	The SMSInfo member of EXTENSIONINFO_10 is valid.
EXTENSIONTYPE_USERPAGERNOTIFY	4	The UserPagerNotify member of EXTENSIONINFO_10 is valid.
EXTENSIONTYPE_ADMINPAGERNOTIFY	5	The AdminPagerNotify member of EXTENSIONINFO_10 is valid.

FAXBODYTYPE_???

Constants for the faxtype fields of FAXINFO_10 and FAXINFO_11.

Constant	Value	Description
FAXBODYTYPE_G4	0x00000001	Reserved for future use.
FAXBODYTYPE_SPDF	0x00000002	Creates a searchable PDF if the fax job requested a PDF body (FFRFLAGS_NEEDS_PDF) and the server is licensed for searchable PDF (BUMP_SPDF.)

FAXEMAILTYPE_???

Constants for the emailtype fields of FAXINFO_10 and FAXINFO_11.

Constant	Value	Description	
FAXEMAILTYPE_CCMAIL	0	The fax originated from cc:Mail.	
FAXEMAILTYPE_MSMAIL	1	The fax originated from MS-Mail.	

Constant	Value	Description
FAXEMAILTYPE_WPO	2	The fax originated from GroupWise.
FAXEMAILTYPE_TRS	3	The fax originated from TRS.
FAXEMAILTYPE_CX3	4	The fax originated from CallXpress3.
FAXEMAILTYPE_PAGER	5	The fax originated a pager (not used).
FAXEMAILTYPE_NOTES	6	The fax originated from Notes.
FAXEMAILTYPE_EXCHANGE	7	The fax originated from Exchange.
FAXEMAILTYPE_SMTP	8	The fax originated from SMTP.
FAXEMAILTYPE_SAP1	9	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP2	10	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP3	11	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP4	12	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP5	13	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP6	14	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP7	15	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP8	16	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP9	17	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP10	18	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP11	19	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP12	20	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP13	21	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP14	22	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP15	23	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP16	24	Fax originated from an SAP gateway.
FAXEMAILTYPE_SMS	25	The fax originated from SMS.
FAXEMAILTYPE_SAP17	26	Fax originated from an SAP gateway.

Constant	Value	Description
FAXEMAILTYPE_SAP18	27	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP19	28	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP20	29	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP21	30	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP22	31	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP23	32	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP24	33	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP25	34	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP26	35	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP27	36	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP28	37	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP29	38	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP30	39	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP31	40	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP32	41	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP33	42	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP34	43	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP35	44	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP36	45	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP37	46	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP38	47	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP39	48	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP40	49	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP41	50	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP42	51	Fax originated from an SAP gateway.

Constant	Value	Description
FAXEMAILTYPE_SAP43	52	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP44	53	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP45	54	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP46	55	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP47	56	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP48	57	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP49	58	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP50	59	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP51	60	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP52	61	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP53	62	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP54	63	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP55	64	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP56	65	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP57	66	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP58	67	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP59	68	Fax originated from an SAP gateway.
FAXEMAILTYPE_SAP60	69	Fax originated from an SAP gateway.
FAXEMAILTYPE_CXINTF	200	Fax originated from CallXpress.
FAXEMAILTYPE_TUIFWD	201	The fax originated from TeleConnect Forward.

FAXERROR_???

Constants for the fax_error_code fields of FAXINFO_10, FAXINFO_11, FAXLISTELEMENT0, FAXLISTELEMENT1, and FAXLISTELEMENT2, and the usFaxErrorCode fields of RFaxStringFaxStatus2() and RFaxStringFaxStatus4().

Constant	Value	Description
FAXERROR_NONE	0	No error.
FAXERROR_BUSY	1	Fax number was busy.
FAXERROR_TRXERROR	2	Fax transmission/receive error.
FAXERROR_POORQUALITY	3	Fax board reports that the fax was received with poor quality.
FAXERROR_NOANSWER	4	No answer at fax number.
FAXERROR_BADFCS	5	Bad FCS information.
FAXERROR_BADCONVERT	6	Error occurred during conversion of body or FCS.
FAXERROR_MAKEFCS	7	Error occurred during creation of FCS text.
FAXERROR_CANTSCHD	8	Can't schedule for some reason.
FAXERROR_UNKNOWN	9	Unknown error reported by fax board.
FAXERROR_HUMAN	10	Human answered the phone.
FAXERROR_GROUP2	11	Reached a G2 fax machine.
FAXERROR_LOCALINUSE	12	Destination unable to receive the fax.
FAXERROR_LINEPROBLEM	13	Problem on the line.
FAXERROR_BADPAPER	14	Invalid form type specified.
FAXERROR_BADSIG	15	Invalid signature used.
FAXERROR_NOSIGAUTH	16	No authorization for attempted signature usage.
FAXERROR_UNDEF_17	17	Undefined.
FAXERROR_DISCARDED	18	Fax was discarded.
FAXERROR_BADPHONE	19	Invalid characters in phone number.
FAXERROR_UNDEF_20	20	Undefined.
FAXERROR_INVALIDCODE	21	User supplied an invalid billing code.
FAXERROR_BADCODE	22	Error in an embedded code.
FAXERROR_BADOCR	23	OCR operation failed.
FAXERROR_BADPRINT	24	Print operation failed.

Constant	Value	Description
FAXERROR_NOLIBAUTH	25	Unauthorized user attempted NEWLIB.
FAXERROR_VIEWSTAR1	26	ViewStar (Imaging system) import error.
FAXERROR_DISAPPROVED	27	Fax was disapproved.
FAXERROR_EMAILDELIVERR	28	Email delivery error.
FAXERROR_BLOCKEDNUMBER	29	Fax not sent because number was blocked.
FAXERROR__EMAILDELBEFOREREAD	30	Email deleted before read.
FAXERROR_BLOCKEDPROHIBITED	31	Fax blocked due to prohibited content.
FAXERROR_OCRROUTEFAILED	32	No OCR recipient.

FAXFILTER_???

Constants for the usFilter argument of functions RFaxGetFaxList(), RFaxGetFaxList2(), and RFaxGetFaxList3().

Constant	Value	Description
FAXFILTER_NONE	0	Don't filter.
FAXFILTER_RECEIVED	1	Filter on received faxes.
FAXFILTER_SENT	2	Filter on sent faxes.
FAXFILTER_INPROCESS	3	Filter on in-process faxes.
FAXFILTER_NEW	4	Received, unviewed, and unprinted.
FAXFILTER_NEEDSAPPROVAL	5	Filter on faxes that are waiting for approval.
FAXFILTER_SENTABANDONED	6	Filter on abandoned faxes.
FAXFILTER_SENTDONE	7	Outbound faxes that have sent successfully or have failed too many times and have been abandoned.
FAXFILTER_RCVDINPROCESS	8	
FAXFILTER_WASINTERCONNECTED	9	Changed from WINFAX

FAXFILTERFLAGS_???

Constants for the usFilter argument of the *FAXFILTERINFO_10* on page 318 function.

Constant	Value	Description
NONE	0	
ALLSOURCES	1 << 0	Indicates the client should facilitate searching both databases. Note that this cannot be done in one RFaxGetFaxList call so the client provides the mechanism.

FAXFLAG_???

Constants for the fr_flags fields of FAXINFO_10 and FAXINFO_11 the flags fields of FAXLISTELEMENT0, *FAXLISTELEMENT1* on page 327, and FAXLISTELEMENT2, and the ulFaxFlags fields of RFaxStringFaxStatus2() and RFaxStringFaxStatus4(). The constants are also used in the different flags fields of RFAXFAXFILTER.

Constant	Value	Description
FAXFLAG_AUTOFORWARDED	0x00000001	Set when the faxrec is auto-forwarded outside the system.
FAXFLAG_HELD	0x00000002	Set when the <HOLD> code is pre-scanned.
FAXFLAG_VIEWED	0x00000004	Set once some portion of the fax has been viewed. This flag is reset when there is a change in owNership.
FAXFLAG_DELETED	0x00000008	Set if this fax belongs to a user's delete fax set, i.e., it is deleted.
FAXFLAG_RECEIVED	0x00000010	Set if this is a received fax.
FAXFLAG_DONT_AUDIT	0x00000020	Set for faxes forwarded within the network or on faxes deleted before they were sent.
FAXFLAG_INITIALIZED	0x00000040	Set the first time we edit the FCS.
FAXFLAG_COMPLETEFCS	0x00000100	Set by server when FCS is considered complete.
FAXFLAG_WAS_FORWARDED	0x00000200	Set when fax is forwarded, cleared after first FCS edit.
FAXFLAG_DELAYSEND	0x00000400	Set when this is a delay send fax.
FAXFLAG_NOCOVER	0x00000800	Set when no cover page is desired.
FAXFLAG_AUTODELETE	0x00001000	Set when user wants fax deleted after successful send.
FAXFLAG_PRINTED	0x00002000	Set once the fax has been printed.
FAXFLAG_NEEDS_FCSCVT	0x00004000	Set if the fax needs a new FCS.
FAXFLAG_AUTODELETEALL	0x00010000	Set when user wants fax deleted after successful or unsuccessful send.
FAXFLAG_BILLCODES_VERIFIED	0x00020000	Set when the billing information has been validated.

Constant	Value	Description
FAXFLAG_FINECOVER	0x00040000	Set when the cover sheet should be imaged in fine mode.
FAXFLAG_GATEWAYFAX	0x00080000	Set when the fax was generated by the mail gateway.
FAXFLAG_MULTICODES	0x00100000	Set when an <NEWDEST> code was encountered or when generating a LST file.
FAXFLAG_NEEDPHONECHK	0x00200000	Set when the fax fields need be checked for phonebook expansion.
FAXFLAG_SMARTRESUME	0x00400000	Set when the fax should use partial-retries.
FAXFLAG_HASTRXNOTES	0x00800000	Set when a fax has notes in a Forward or Route Trx entry.
FAXFLAG_WASAPPROVED	0x01000000	Set when a fax has been approved for sending.
FAXFLAG_NEEDSAPPROVAL	0x02000000	Set when a fax needs approval to be sent.
FAXFLAG_GENERIC1	0x04000000	Application-specific flag.
FAXFLAG_WAS_DNDBLOCKED	0x08000000	Set when a document is blocked by the DND.
FAXFLAG_COMPLETEEVENT	0x10000000	Fire off a complete event when this fax completes successfully or is abandoned.
FAXFLAG_BROADCAST	0x20000000	Handle as a broadcast fax.
FAXFLAG_WASINTERCONNECTED	0x40000000	XRouted via Interconnect to new server. Was: Sent via the WINFAX HAL.
FAXFLAG_GENERIC2	0x80000000	Application-specific flag.

FAXFLAG2_???

Constants for the fr_flags2 fields of FAXINFO_10 and FAXINFO_11, the flags2 field of FAXLISTELEMENT2, and the ulFaxFlags2 field of RFaxStringFaxStatus4().

Constant	Value	Description
FAXFLAG2_LCRTIMEDELAYED	0x00000004	The fax was previously time-delayed by an LCR rule.
FAXFLAG2_SLOWCHECK	0x00000008	The fax status is updated less often.
FAXFLAG2_PRODFAX	0x00000010	Reserved. Do not use.
FAXFLAG2_INLJOB	0x00000020	Reserved. Do not use.
FAXFLAG2_PRINTONLY	0x00000040	The fax has an associated PDF file.
FAXFLAG2_HASPDF	0x00000080	Hot link to a PDF file is enabled.

Constant	Value	Description
FAXFLAG2_HASHOTLINK	0x00000100	Hide the fax from the SecureDocs Web box.
FAXFLAG2_HIDEFROMWEB	0x00000200	The fax will appear to be deleted from the SecureDocs Web site but will still be in the user's FaxUtil mailbox.
FAXFLAG2_USERFORWARDSRC	0x00000400	The source fax that was used to forward to a new Rightfax user. This constant is different from FAXFLAG_WAS_FORWARDED in that it is set on the source fax and it is set on a forward-to-user operation (as opposed to a forward-to-new-fax-number operation).
FAXFLAG2_FCSNOTESAREUNICODE	0x00000800	Indicates that the NOTEINFO_10.text field uses UNICODE characters.
FAXFLAG2_SENTGATEWAYFAX	0x00001000	Added flag for fax sent to an email gateway.
FAXFLAG2_AUTOFORWARDEDVIASFC	0x00002000	To differentiate for billing code validation.
FAXFLAG2_SMS	0x00004000	New SMS type document.
FAXFLAG2_RECIPIENTNOTIFY	0x00008000	Notify the recipient.
FAXFLAG2_SILENTDNDNOTIFY	0x00010000	Javelina silent notification for user on DnD restriction.
FAXFLAG2_DOCUMENTWASEDCED	0x00020000	Added ability to tell if document was EDC'ed.
FAXFLAG2_P2P	0x00040000	Added ablity to tell if fax was sent via P2P (ORCA).
FAXFLAG2_DOCTYPE_TEXT *	0x00080000	Use text (no optimization). Cannot be used with FAXFLAG2_DOCTYPE_LIGHT or FAXFLAG2_DOCTYPE_PANDI which are optimizations.
FAXFLAG2_DOCTYPE_LIGHT*	0x00100000	Set when user wants fax converted using settings generally effective for handling a light fax.
FAXFLAG2_FROMARCHIVE	0x00200000	Fax was retrieved from the archive database.
FAXFLAG2_GATEWAYDEFAULTUSER	0x00800000	Set when a gateway can't find a user for an outbound fax and falls back to a default user.
FAXFLAG2_DOCTYPE_PANDI*	0x01000000	Set when user wants fax converted using settings generally good for handling photos and images.
FAXFLAG2_VIEWED_FULLY	0x02000000	Indicates the fax was fully viewed under the current owner. This flag is reset when there is a change in ownership.
FAXFLAG2_NEEDS_FCS_OCR	0x04000000	Set when the FCS for the fax must be OCRed.
FAXFLAG2_HAS_FCS_OCR	0x08000000	Set when the FCS for the fax has been OCRed.
FAXFLAG2_HAS_WORKFLOWED	0x10000000	Indicates fax has been in a workflow and may have workflow values.

Constant	Value	Description
FAXFLAG2_EXCEPTION_COMPLETE	0x20000000	Indicates fax has been in an exception workflow, been processed, and has returned to its prior workflow. This flag is cleared after a user locks it.
* If none of the FAXFLAG2_DOCTYPE_??? flags are set, then the conversion bias set for the server is used.		

FAXFOLDER_???

Constants for the usFolder fields of FAXINFO_10 and FAXINFO_11, and the usFolder argument of functions RFaxGetFaxList(), RFaxGetFaxList2(), RFaxGetFaxList3(), and RFaxGetNewFaxCount().

Constant	Value	Description
FAXFOLDER_ALL	0	Get all of the faxes in all of the folders. When used as FAXINFO_10.usFolder and FAXINFO_11.usFolder structure members, a value of zero indicates the Main folder.
FAXFOLDER_WORKFLOW	0xFFFA	Get just the workflow folder.
FAXFOLDER_ARCHIVE	0xFFFB	Get just the archive folder.
FAXFOLDER_NONE	0xFFFC	
FAXFOLDER_ALLBUTTRASH	0xFFFD	Everything except the Trash folder.
FAXFOLDER_TRASH	0xFFFE	Get just the Trash folder.
FAXFOLDER_MAIN	0xFFFF	Get just the Main folder.

FAXGETLIST_???

Constants for optional flags (dwOptions) of functions RFaxGetFaxList3() and RFaxGetFaxList4().

Constant	Value	Description
FAXGETLIST_SETUSERID	0x0001	Set the UserID member of each FAXLISTELEMENT0/FAXLISTELEMENT1. Normally left blank by the server.
FAXGETLIST_SETSTATUSTYPE	0x0002	Set usStatusType and dwStatusID members of each FAXLISTELEMENT0/FAXLISTELEMENT1. Not provided by the server.

FAXNUMBERORDER_???

Constants for RFaxGetFaxNumberList().usOrder.

Constant	Value	Description
FAXNUMBERORDER_DEFAULT	0	
FAXNUMBERORDER_FAXNUMBER	1	
FAXNUMBERORDER_OWNER	2	

FAXORDER_???

Constants for the usFaxOrdering argument of function RFaxGetPagedFaxList().

Constant	Value	Description
FAXORDER_DEFAULT	0	
FAXORDER_DATETIME	1	
FAXORDER_TOFROM	2	
FAXORDER_DESTINATION	3	
FAXORDER_STATUS	4	
FAXORDER_BILLCODE1	5	
FAXORDER_BILLCODE2	6	
FAXORDER_NUMPAGES	7	
FAXORDER_UNIQUEID	8	
FAXORDER_COMMENT	9	
FAXORDER_FOLDER	10	
FAXORDER_BFTBYTES	11	
FAXORDER_HISTCHGTIME	12	
FAXORDER_COMPANY	13	
FAXORDER_SENTRECEIVED	14	
FAXORDER_VIEWED	15	
FAXORDER_PRINTED	16	
FAXORDER_HASBFT	17	

Constant	Value	Description
FAXORDER_HASPDF	18	
FAXORDER_DOCTYPE	19	
FAXORDER_OWNER	20	
FAXORDER_NEEDSPDF	21	
FAXORDER_MSGFROMTRANSPORT	22	
FAXORDER_GATEWAYFAX	23	
FAXORDER_HASTRXNOTES	24	
FAXORDER_FINECOVER	25	
FAXORDER_COMPLETEEVENT	26	
FAXORDER_NUMTRX	27	
FAXORDER_HANDLE	28	
FAXORDER_DNIS	29	Show DNIS info.
FAXORDER_COMPLETIONTIME	30	Show completion time.
FAXORDER_JOBID	31	JobId (RFConnect).
FAXORDER_ANI	32	ANI.
FAXORDER_LOCKINGUSERID	33	ID of locking user.

FAXSTAT_???

Constants for the fax_status fields of FAXINFO_10FAXINFO_11, FAXLISTELEMENT0, FAXLISTELEMENT1, and FAXLISTELEMENT2, and the usFaxStatus fields of RFaxStringFaxStatus2() and RFaxStringFaxStatus4().

Constant	Value	Description
FAXSTAT_UNBORN	0	The fax has not yet been created.
FAXSTAT_NEEDS_FCS	1	The fax requires a FCS to be generated and none was specified.
FAXSTAT_NEEDS_CONVERSION	2	The fax is waiting for conversion.
FAXSTAT_NEEDS_TOBESENT	3	The fax is waiting to be sent.

Constant	Value	Description
FAXSTAT_IN_CONVERSION	4	The fax is in conversion.
FAXSTAT_IN_SEND	5	The fax is currently sending. usPagesInFront indicates the percentage of pages sent so far.
FAXSTAT_DONE_OK	6	The fax was successfully sent or received.
FAXSTAT_MANUALFCS	7	The fax record was entered manually. Sometimes referred to as a thermal fax. These are often used to note that some transaction occurred when it did not go through RightFax.
FAXSTAT_IN_SCHEDULE	8	The fax is scheduled for send. usPagesInFront indicates the number of pages are ahead of this fax in the queue.
FAXSTAT_DONE_ERROR	9	Too many errors; will not be retried.
FAXSTAT_DUPLICATE	10	Set on a network forwarded fax.
FAXSTAT_ERROR	11	Had an error (see fax_error_code). Will be retried.
FAXSTAT_ATTN	12	Fax needs attention.
FAXSTAT_NEEDS_ATTACH	13	Fax needs an attachment.
FAXSTAT_HELD	14	Fax is held for preview.
FAXSTAT_INOCR	15	Deprecated status. See OCRSTAT_INOCR and FAXINFO_11/LISTELEMENT4.body_ocr_status/fcs_ocr_status.
FAXSTAT_INPRINT	16	Deprecated status. See PRINTSTAT_INPRINT and FAXINFO_11/LISTELEMENT4.print_status.
FAXSTAT_QUEPRINT	17	Deprecated status. See PRINTSTAT_QUEUED and FAXINFO_11/LISTELEMENT4.print_status.
FAXSTAT_QUEOCR	18	Deprecated status. See OCRSTAT_QUEUED and FAXINFO_11/LISTELEMENT4.body_ocr_status/fcs_ocr_status.
FAXSTAT_IN_VALIDATE	19	Fax is waiting for validation.
FAXSTAT_IN_APPROVAL	20	Fax is waiting for approval.

FAXSTRINGSTAT_???

Constants for the dwOptions argument of functions RFaxStringFaxStatus3() and RFaxStringFaxStatus5().

Constant	Value	Description
FAXSTRINGSTAT_SETSTATUSTYPE	0x00000001	Sets the usStatusType member of FAXLISTELEMENT0 and FAXLISTELEMENT1.
FAXSTRINGSTAT_SETSTATUSID	0x00000002	Sets the dwStatusID member of FAXLISTELEMENT0 and FAXLISTELEMENT1.
FAXSTRINGSTAT_UTF8	0x00000004	Returns string in UTF8 format.
FAXSTRINGSTAT_RESERVED	0x80000000	Reserved for future use.

FDFLAG_???

Constants for the displayflags fields of FAXLISTELEMENT0 and FAXLISTELEMENT1.

Constant	Value	Description
FDFLAG_HASFCS	0x0001	Fax has a cover sheet.
FDFLAG_HASBODY	0x0002	Fax has a body.
FDFLAG_HASBFT	0x0004	BFT (binary file attachment, like OCRed text).
FDFLAG_ONLYBFT	0x0008	Object is a binary file.

FFRFLAGS_???

Constants for the ffr_flags fields of FAXINFO_10, FAXINFO_11, and FAXLISTELEMENT2.

Constant	Value	Description
FFRFLAGS_NONE	0x00000000	Nothing needs to be done.
FFRFLAGS_NEEDS_PRESCAN	0x00000001	Pre-scan of fax is necessary (searches for embedded codes).
FFRFLAGS_NEEDS_BODYCVT	0x00000002	The body of the fax needs to be converted.
FFRFLAGS_NEEDS_PAPERCVT	0x00000004	An image needs to be overlayed.
FFRFLAGS_NEEDS_SIGCVT	0x00000008	A signature needs to be added to the document.
FFRFLAGS_NEEDS_ATTACH	0x00000010	An attachment needs to be added.
FFRFLAGS_NEEDS_CONVERT	0x00000003	Combination of FFRFLAGS_NEEDS_PRESCAN and FFRFLAGS_NEEDS_BODYCVT.

Constant	Value	Description
FFRFLAGS_CONVERT_ALL	0x0000001F	Combination of FFRFLAGS_NEEDS_PRESCAN, FFRFLAGS_NEEDS_BODYCVT, FFRFLAGS_NEEDS_PAPERCVT, FFRFLAGS_NEEDS_SIGCVT, and FFRFLAGS_NEEDS_ATTACH.
FFRFLAGS_LIBDOC	0x00000020	A library document needs to be converted.
FFRFLAGS_HASBFT	0x00000040	Set when szBFTFile is present.
FFRFLAGS_NEWLIB	0x00000080	This fax had a NEWLIB code.
FFRFLAGS_BODYHASPRINTED	0x00000100	Body has been autoprinted.
FFRFLAGS_FOUNDCOVERCODE	0x00000200	A cover sheet embedded code was found.
FFRFLAGS_DELETELASTPAGE	0x00000400	When conversion is complete, should the last page be deleted?
FFRFLAGS_COVERONLYCODE	0x00000800	Allow only the cover sheet to go through.
FFRFLAGS_FAXJOBIMAGE	0x00001000	faxfilename[] is an image for a fax job and shouldn't be deleted.
FFRFLAGS_DELETEFIRSTPAGE	0x00002000	When conversion is complete, should the first page be deleted?
FFRFLAGS_NEEDS_PDF	0x00004000	Need to convert a PDF.
FFRFLAGS_NEEDS_NOTESTXT	0x00008000	Directs the server to create a text file from the Notes record associated. This was started for SMS.
FFRFLAGS_PDF_LANDSCAPE	0x00010000	Convert PDF in landscape layout when requested.
FFRFLAGS_NEEDS_OCR	0x00020000	OCR when requested.

FILECOPY_TYPE__???

Constants for the .usDestFormat argument of function RFaxFileCopyType().

Constant	Value	Description
FILECOPY_TYPE_G3	0x00000001	Raw group III.
FILECOPY_TYPE_TIFFG3	0x00000002	Group III TIFF.
FILECOPY_TYPE_TIFFG4	0x00000004	Group IV TIFF.
FILECOPY_TYPE_GIF	0x00000008	GIF.
FILECOPY_TYPE_RFG3	0x00000010	RightFax Group III.

FILEFILTER__???

Used by RFaxGetGroupFilteredCoversList()

Constant	Value	Description
FILEFILTER_NONE	0X000	
FILEFILTER_GROUP	0x0001	

FOLDERFLAGS_???

Constants for the ulFlags field of *FOLDERLISTELEMENT1* on page 334.

Constant	Value	Description
FOLDERFLAGS_NONE	0x00000000	
FOLDERFLAGS_DELETED	(1 << 0)	

FORMFLAG_???

Constants for the flags field of FORMLISTELEMENT0 and ft_flags field of FORMINFO_10.

Constant	Value	Description
FORMFLAG_APPEND	1	Set if this form type specifies that pages are to be appended to a fax.

FSMSG_???

Constants for the usMsgType argument of function RFaxSendFSMsg().

Constant	Value	Description
FSMSG_KICKEVT	3	The fax server will check the event queue and, if an event for the fax specified in the "handle" argument of the RFaxSendFSMsg() function is found, move the event to the front of the event queue. If no event for the specified fax is found, look up the fax in the database and reschedule it. If you do not specify a user ID using the "lpUserID" argument, the server will look up the owner of the fax and use that ID to re-schedule the fax. This message has the same behavior as the RFaxKickFax() function.

Constant	Value	Description
FSMSG_NEWROUTE	4	A route event is posted to the server using the “handle” argument of the RFaxSendFSMsg() function as the handle to the routed fax and the “lpUserID” argument as the user to whom the fax was routed.

GENHISTTYPE_???

Constants for the d.physical.tr_rectype field of HISTLISTELEMENT0 and the dwGenHistType argument of function RFaxInitHistoryItem().

The constants indicate the record types in the generic history database.

Constant	Value	Description
GENHISTTYPE_VIEWED	100	Viewed.
GENHISTTYPE_FORWARDED	101	Forwarded.
GENHISTTYPE_COMBINED	102	Combined.
GENHISTTYPE_SPLIT	103	Split.
GENHISTTYPE_SMS	104	SMS.
GENHISTTYPE_FOIP	105	FoIP.
GENHISTTYPE_EICON	106	Eicon.
GENHISTTYPE_INTEL	107	Intel.
GENHISTTYPE_P2P	108	P2P.
GENHISTTYPE_EDOCS	109	eDocs
GENHISTTYPE_FAXSCHED	110	Scheduled.
GENHISTTYPE_ROUTED	111	Routed.
GENHISTTYPE_ROUTEDFAILED	112	Failed routing.
GENHISTTYPE_ANNOTATED	113	Annotated.
GENHISTTYPE_ANNOTATED_SOLID	114	
GENHISTTYPE_FAX_IMPORTED	115	Imported.
GENHISTTYPE_FAX_DELETED	116	Deleted.

Constant	Value	Description
GENHISTTYPE_SEND_VIA_EMAIL	117	Sent via email.
GENHISTTYPE_SEND_VIA_EMAIL_DELEGATE	118	Sent via email by delegate.
GENHISTTYPE_EDC_ARCHIVE	119	Archived.
GENHISTTYPE_EDC_ARCHIVE_DELEGATE	120	Archived by delegate.
GENHISTTYPE_CONFIRMATIONRECEIPT	121	
GENHISTTYPE_CONFIRMATIONRECEIPT_ DELEGATE	122	
GENHISTTYPE_COMBINED_DELEGATE	123	
GENHISTTYPE_ROUTE_DELEGATE	124	
GENHISTTYPE_NETFORWARD_DELEGATE	125	
GENHISTTYPE_FORWARDED_DELEGATE	126	
GENHISTTYPE_EDC_FAX_SUBMITTED_FROM_ DEVICE	127	
GENHISTTYPE_ARCHIVED	128	
GENHISTTYPE_MANUAL_ROUTE	129	
GENHISTTYPE_UNVIEWED	130	
GENHISTTYPE_MANUAL_ROUTE_DELEGATE	131	
GENHISTTYPE_RELEASED	132	
GENHISTTYPE_WORKREQUEST_FAILED	135	
GENHISTTYPE_OT_CAPTURE	136	History record for zip receives.
GENHISTTYPE_FAX_SAVEAS	137	
GENHISTTYPE_FAX_SAVEAS_DELEGATE	138	
GENHISTTYPE_MOVE_TO_TRASH	139	Fax moved to trash.
GENHISTTYPE_MOVE_TO_TRASH_DELEGATE	140	Fax moved to trash by a delegate.
GENHISTTYPE_EDIT_RECORD	141	

Constant	Value	Description
GENHISTTYPE_EDIT_RECORD_DELEGATE	142	
GENHISTTYPE_INCEPTION_GENERAL	139	General fax creation.
GENHISTTYPE_INCEPTION_GENERAL_DELEGATE	151	General fax creation history record when created by a delegate.
GENHISTTYPE_INCEPTION_COMBINESPLIT	152	Fax was created from combine or split function.
GENHISTTYPE_INCEPTION_COMBINESPLIT_ DELEGATE	153	Fax was created by a delegate from combine or split function.
GENHISTTYPE_INCEPTION_FORWARD	154	Fax was created from forward function.
GENHISTTYPE_INCEPTION_FORWARD_DELEGATE	155	Fax was created by a delegate from forward function.
GENHISTTYPE_WORKFLOW_ENTERED	156	A workflow is initiated.
GENHISTTYPE_WORKFLOW_VIEWED	157	User locked the workflow fax.
GENHISTTYPE_WORKFLOW_EDITMETADATA	158	User entered metadata to a fax that is in a workflow.
GENHISTTYPE_WORKFLOW_COMPLETED	159	User completed the workflow.
GENHISTTYPE_BLOCKED_PROHIBITED	160	Fax was blocked due to prohibited content.
GENHISTTYPE_ROUTE_FTP	161	Fax was routed to an FTP destination.
GENHISTTYPE_ROUTE_FTP_FAILURE	162	Fax failed to route to an FTP destination.
GENHISTTYPE_FULLY_VIEWED	163	All pages of the fax have been viewed.
GENHISTTYPE_PARTLY_VIEWED	164	Not all of the pages of the fax have been viewed.
GENHISTTYPE_WORKFLOW_REJECTED	165	User rejected fax from its workflow.
GENHISTTYPE_WORKFLOW_EXCEPTED	166	User excepted fax from its workflow.
GENHISTTYPE_WORKFLOW_RESUMED	167	User sent a fax back to its prior workflow after an exception.
GENHISTTYPE_AUTOMATIC_FORWARD	168	The server automatically forwarded the fax to the specified user.
GENHISTTYPE_AUTOMATIC_FORWARD_FAILURE	169	The server failed to automatically forward the fax to the specified user.
GENHISTTYPE_INCEPTION_AUTOMATIC_ FORWARD	170	The fax is the result of a fax automatically forwarded to the specified user.
GENHISTTYPE_SEND_VIA_EMAIL_DEST	171	Email initiated by ~u
to ~s2~s3~s4~s5~s6.

Constant	Value	Description
GENHISTTYPE_SEND_VIA_EMAIL_DELEGATE_DEST	172	Send via Email initiated by ~u on behalf of ~s1
;to ~s2~s3~s4~s5~s6.
GENHISTTYPE_POSTFAXPROCESSED	173	Exported file ~s2 via ~s1.
GENHISTTYPE_SHAREPOINT_IMPORTED_FAX	174	Imported from SharePoint.
GENHISTTYPE_SHAREPOINT_EXPORTED_FAX	175	Exported to SharePoint.
GENHISTTYPE_AUTOROUTED	176	Automatically routed faxes have a new record.
GENHISTTYPE_POSTFAXPROCESSED_MANUAL	177	Exported file ~s2 via ~s1 by ~u.

GENMAILTYPE_???

Constants for the ucGenMailType field of GENMAIL1REQGENMAIL2REQ, and GENMAILPREVIEWREQ, and the usMailType argument of function RFaxGetGenericMailEvent().

Constant	Value	Description
GENMAILTYPE_TRS	1	Route to TRS.
GENMAILTYPE_CX3	2	CallXpress3.
GENMAILTYPE_NOTES	3	Lotus Notes.
GENMAILTYPE_EXCHANGE	4	Exchange.
GENMAILTYPE_CXINTF	5	CallXpress.
GENMAILTYPE_SMTP	6	SMTP.
GENMAILTYPE_SAP1	7	SAP.
GENMAILTYPE_SAP2	8	SAP.
GENMAILTYPE_SAP3	9	SAP.
GENMAILTYPE_SAP4	10	SAP.
GENMAILTYPE_INL	11	InternetLink.
GENMAILTYPE_INL2	12	InternetLink.
GENMAILTYPE_SAP5	13	SAP.

Constant	Value	Description
GENMAILTYPE_SAP6	14	SAP.
GENMAILTYPE_SAP7	15	SAP.
GENMAILTYPE_SAP8	16	SAP.
GENMAILTYPE_SAP9	17	SAP.
GENMAILTYPE_SAP10	18	SAP.
GENMAILTYPE_SAP11	19	SAP.
GENMAILTYPE_SAP12	20	SAP.
GENMAILTYPE_SAP13	21	SAP.
GENMAILTYPE_SAP14	22	SAP.
GENMAILTYPE_SAP15	23	SAP.
GENMAILTYPE_INLDEL	24	Delete email generated by GENMAILTYPE_INL.
GENMAILTYPE_AUTOREPLY	25	AutoReply messages for AR fax processing.
GENMAILTYPE_SMS	26	SMS.
GENMAILTYPE_SAP16	27	SAP.
GENMAILTYPE_SAP17	28	SAP.
GENMAILTYPE_SAP18	29	SAP.
GENMAILTYPE_SAP19	30	SAP.
GENMAILTYPE_SAP20	31	SAP.
GENMAILTYPE_SAP21	32	SAP.
GENMAILTYPE_SAP22	33	SAP.
GENMAILTYPE_SAP23	34	SAP.
GENMAILTYPE_SAP24	35	SAP.
GENMAILTYPE_SAP25	36	SAP.
GENMAILTYPE_SAP26	37	SAP.
GENMAILTYPE_SAP27	38	SAP.

Constant	Value	Description
GENMAILTYPE_SAP28	39	SAP.
GENMAILTYPE_SAP29	40	SAP.
GENMAILTYPE_SAP30	41	SAP.
GENMAILTYPE_SAP31	42	SAP.
GENMAILTYPE_SAP32	43	SAP.
GENMAILTYPE_SAP33	44	SAP.
GENMAILTYPE_SAP34	45	SAP.
GENMAILTYPE_SAP35	46	SAP.
GENMAILTYPE_SAP36	47	SAP.
GENMAILTYPE_SAP37	48	SAP.
GENMAILTYPE_SAP38	49	SAP.
GENMAILTYPE_SAP39	50	SAP.
GENMAILTYPE_SAP40	51	SAP.
GENMAILTYPE_SAP41	52	SAP.
GENMAILTYPE_SAP42	53	SAP.
GENMAILTYPE_SAP43	54	SAP.
GENMAILTYPE_SAP44	55	SAP.
GENMAILTYPE_SAP45	56	SAP.
GENMAILTYPE_SAP46	57	SAP.
GENMAILTYPE_SAP47	58	SAP.
GENMAILTYPE_SAP48	59	SAP.
GENMAILTYPE_SAP49	60	SAP.
GENMAILTYPE_SAP50	61	SAP.
GENMAILTYPE_SAP51	62	SAP.
GENMAILTYPE_SAP52	63	SAP.

Constant	Value	Description
GENMAILTYPE_SAP53	64	SAP.
GENMAILTYPE_SAP54	65	SAP.
GENMAILTYPE_SAP55	66	SAP.
GENMAILTYPE_SAP56	67	SAP.
GENMAILTYPE_SAP57	68	SAP.
GENMAILTYPE_SAP58	69	SAP.
GENMAILTYPE_SAP59	70	SAP.
GENMAILTYPE_SAP60	71	SAP.

GET_GROUP__???

Constant	Value	Description
GET_GROUP_LIB_DOCS	1	Used by RFaxGetLibDocList4().
GET_GROUP_LIBDOC_LIST	2	Used by RFaxGetGroupExcludedLibDocList()

Constant	Value	Description
GET_GROUP_PHONEBOOKS	1	Used by RFaxGetPhonebooksList().
GET_GROUP_PHONEBOOKS_LIST	2	

GET_PHONEBOOK_GROUPS__???

Constant	Value	Description
GET_PHONEBOOKS_GROUPS	1	

GETFOLDERS_???

Key for usGetOpt parameter to *RFaxGetFolderList3()* on page 239.

Constant	Value	Description
GETFOLDERS_NONE	0x0000	None.
GETFOLDERS_INCLUDEBUILTIN	0x0001	Include the permanent (or built-in) folders.
GETFOLDERS_INCLUDEALLCHILDREN	0x0002	Ignore the usParent parameter.
GETFOLDERS_INCLUDEDELETED	0x0004	Include deleted folders.

GETGROUPS_???

Constants for the usGetOptions argument of function RFaxGetGroupList2().

Constant	Value	Description
GETGROUPS_QUICKER	0x0000	Gets a list of groups with no extra work that can cause slowdowns on servers with heavy usage. It is recommended that this option be used when possible.
GETGROUPS_USERCOUNT	0x0001	Get the count of users for each group. This can impact performance on servers with lots of groups and/or lots of users.
GETGROUPS_ADMINIDS	0x0002	Causes the server to fill in the GROUPINFO_10.admin and alt_admin fields. This can impact performance on servers with many groups and/or admins.
GETGROUPS_SEARCH	0x0004	Causes the server to use the groupid in the keydata for searching.
GETGROUPS_MINIONS	0x0008	Gets only groups that the logged in user has some administrative or monitoring dominion over.

GETHISTORYLISTOPT_???

Constants used in the usOptions parameter to *RFaxGetHistoryList4()* on page 141.

Constant	Value	Description
GETHISTORYLISTOPT_NONE	0x00	
GETHISTORYLISTOPT_INLINE_METADATA	0x01	Return history metadata in text.

GETPHONES_???

Constants for the usGetOptions parameter of *RFaxGetPhoneList3()* on page 250 and *RFaxGetPhoneList4()* on page 249.

Constant	Value	Description
GETPHONES_NONE	0x0000	
GETPHONES_NOGROUPS	0x0001	Gets single items only. No phone groups.
GETPHONES_OTHERBOOKS	0x0002	Search the phonebooks configured for the user.

GETSEMAPHOREFLAGS_???

Constants for the usFlags parameter of RFaxGetSemaphore().

Constant	Value	Description
GETSEMAPHOREFLAGS_AUTORELEASE	0x0001	Tells Windows the underlying semaphore file is a temporary file. With this option, Windows deletes the file automatically even if the process crashes; however, other processes cannot access the file so the user will not be able to read the contents. **Note** This does not affect the life of the lock--just the life of the file. Without this flag, the existence of the file does not mean the lock is necessarily still held.
GETSEMAPHOREFLAGS_HIDDEN	0x0002	Mark the semaphore file as hidden.

GETUSERS_???

Constants for the usGetOptions argument of function RFaxGetUserList4().

Constant	Value	Description
GETUSERS_QUICKER	0x0000	Gets a list of users with no extra work that can cause slowdowns on servers with heavy usage. It is recommended that this option be used when possible.
GETUSERS_GROUPID	0x0001	Get the ID of the group each user belongs to. This has a mild performance impact on servers with lots of users.
GETUSERS_DOCCOUNT	0x0002	Get the count of documents for each user. This has a huge performance impact on servers with lots of users and/or lots of documents.
GETUSERS_MINIONS	0x0004	Only get users this user has some administrative or delegated domain over.

GETWORKFLOWDELEGATIONSOPT_???

Constants used in the usOptions parameter of RFaxGetWorkflowDelegationsList.

Constant	Value	Description
GETWORKFLOWDELEGATIONSOPT_NONE	0x00	

GROUP_EDCFLAGS_???

Constants for GROUPINFO_10.usEDCArchiveFlags.

Constant	Value	Description
GROUP_EDCFLAGS_XMLGEN	0x0001	XML Generator should process.
GROUP_EDCFLAGS_VAULT	0x0002	Vault should process.
GROUP_EDCFLAGS_FNET	0x0004	Filenet should process.

GROUPORDER_???

Filters for the usOrder parameters of function *RFaxGetPagedGroupList()* on page 179.

Constant	Value	Description
GROUPORDER_DEFAULT	0	
GROUPORDER_GROUPID	1	
GROUPORDER_MEMBERS	2	
GROUPORDER_ADMINS	3	
GROUPORDER_ALTADMINS	4	
GROUPORDER_ROUTECODE	5	
GROUPORDER_FCSMODEL	6	
GROUPORDER_VIEWEDAGE	7	
GROUPORDER_SENTAGE	8	
GROUPORDER_DELETEAGE	9	
GROUPORDER_NEWAGE	10	
GROUPORDER_FAILEDAGE	11	
GROUPORDER_NOTCOMPAGE	12	

GRP_???

Constant used for the ulArchiveFlags field of GROUPINFO_10.

Constant	Value	Description
GRP_ARCHIVEFLAG	0x00000001	

GRP_ATTACHPAGES_???

Constants for the ucNotifyAttachmentPages field of GROUPINFO_10.

Constant	Value	Description
GRP_ATTACHPAGES_FIRST	0x00	Attach first page only.
GRP_ATTACHPAGES_RX_ALL	0x01	Attach all pages.
GRP_ATTACHPAGES_TX_ALL	0x02	

GRP_ATTACHTYPE_???

Constants for the ucNotifyAttachmentType field of GROUPINFO_10.

Constant	Value	Description
GRP_ATTACHTYPE_FAX	0x00	Attachment is a fax image.
GRP_ATTACHTYPE_RX_PDF	0x01	Received attachment is a PDF file.
GRP_ATTACHTYPE_TX_PDF	0x02	Sent attachment is a PDF file

GRP_FCSDLGCTRL1_???

Constants for the ulFCSDlgCtrl1 field of GROUPINFO_10.

Constant	Value	Description
GRP_FCSDLGCTRL1_NONE	0x00000000	
GRP_FCSDLGCTRL1_NOVOICENUM	0x00000001	If true, hide the **Voice number** field.
GRP_FCSDLGCTRL1_NOCOMPANY	0x00000002	If true, hide the **Company** field.

Constant	Value	Description
GRP_FCSDLGCTRL1_NOCITYSTATE	0x00000004	If true, hide the **City/State** field.
GRP_FCSDLGCTRL1_NOPBBUTTON	0x00000008	If true, hide the **Phonebook** button.
GRP_FCSDLGCTRL1_NOADDPBENTRYBUTTON	0x00000010	If true, hide the **Add Phonebook Entry** button.
GRP_FCSDLGCTRL1_NOFORMTYPE	0x00000020	If true, hide the **Form** field.
GRP_FCSDLGCTRL1_NOPRIORITY	0x00000040	If true, hide the **Priority** field.
GRP_FCSDLGCTRL1_NOPREVIEW	0x00000080	If true, hide the **Hold for Preview** field.
GRP_FCSDLGCTRL1_NOCOVERSHEETTOGGLE	0x00000100	If true, hide the **Use Cover Sheet** field.
GRP_FCSDLGCTRL1_NOCOVERSHEETSELECT	0x00000200	If true, hide the **Select Cover Sheet** field.
GRP_FCSDLGCTRL1_NOSMARTRESUME	0x00000400	If true, hide the **Use Smart Resume** field.
GRP_FCSDLGCTRL1_NOFROMNAME	0x00000800	If true, hide the **From Name** field.
GRP_FCSDLGCTRL1_NOFROMPHONENUM	0x00001000	If true, hide the **From Phone Number** field.
GRP_FCSDLGCTRL1_NOCALLBACK	0x00002000	If true, hide the **Callback** field.
GRP_FCSDLGCTRL1_NODELAYSEND	0x00004000	If true, hide the **Delay Send** field.
GRP_FCSDLGCTRL1_NOBILLINFO1	0x00008000	If true, hide the **BillInfo1** field.
GRP_FCSDLGCTRL1_NOBILLINFO2	0x00010000	If true, hide the **BillInfo2** field.
GRP_FCSDLGCTRL1_NOBILLINFOLOOKUP	0x00020000	If true, hide the **Billing Code Lookup** button.
GRP_FCSDLGCTRL1_NOMOREBUTTON	0x00040000	If true, hide the **Billing Code Lookup More** button.
GRP_FCSDLGCTRL1_NONOTESBUTTON	0x00080000	If true, hide the **Notes** button.
GRP_FCSDLGCTRL1_NOUSEPDF	0x00100000	If true, hide the **Use PDF** field.
GRP_FCSDLGCTRL1_NOPDFOPTIONS	0x00200000	If true, hide the **PDF Options** field.
GRP_FCSDLGCTRL1_NOCERTIFIED	0x00400000	If true, hide the **Use Certified Delivery** field.
GRP_FCSDLGCTRL1_NORECIPIENTYPE	0x00800000	If true, hide the **Recipient Type** button.
GRP_FCSDLGCTRL1_NONATIVEDOCATTACH	0x01000000	If true, hide the **Attach Native Document** field.
GRP_FCSDLGCTRL1_NOALTBODYATTACH	0x02000000	If true, hide the **Alternate Body Attach** field.
GRP_FCSDLGCTRL1_NOSCANATTACH	0x04000000	If true, hide the **From Scanner Attach** button.

Constant	Value	Description
GRP_FCSDLGCTRL1_NOFILEATTACH	0x08000000	If true, hide the **File Attach** button.
GRP_FCSDLGCTRL1_NONOTES	0x10000000	If true, hide the **Notes** field.
GRP_FCSDLGCTRL1_NOCOMMENTS	0x20000000	If true, hide the **Comments** field.

GRP_FCSDLGCTRL2_???

Constants for the ulFCSDlgCtrl2 field of GROUPINFO_10.

Constant	Value	Description
GRP_FCSDLGCTRL2_NONE	0x00000000	
GRP_FCSDLGCTRL2_NOPRIVATEFAX	0x00000001	If true, hide the **Fax Number** field.
GRP_FCSDLGCTRL2_NOGENERALFAX	0x00000002	If true, hide the **Company Fax Number** field.
GRP_FCSDLGCTRL2_NOGENERALVOICE	0x00000004	If true, hide the **Company Voice Number** field.
GRP_FCSDLGCTRL2_NOLIBDOCSELECT	0x00000008	If true, hide the **Select Library Document** field.
GRP_FCSDLGCTRL2_NOSECURESEND	0x00000010	If true, hide the **Send Via SecureDocs** field.
GRP_FCSDLGCTRL2_NOAUTODELETESETTING	0x00000020	If true, hide the **Auto Delete** field.
GRP_FCSDLGCTRL2_NOTRANSMITQUALITY	0x00000040	If true, hide the **Fine Mode** field.
GRP_FCSDLGCTRL2_NOALTFAXNUM	0x00000080	If true, hide the **Alternate Fax Number** field.
GRP_FCSDLGCTRL2_NOPHONEBOOKIMPORT	0x00000100	

GFLAG_???

Constants for the flags field of LIBDOCLISTELEMENT0, LIBDOCLISTELEMENT1, and LIBDOCLISTELEMENT2.

Constant	Value	Description
GFLAG_NORMMODE	0x00000000	Normal mode.
GFLAG_LIBDOC	0x00000001	Only older servers will have graphic objects without the library document bit set.
GFLAG_FINEMODE	0x00000002	Library document image is fine mode (200 x 200).
GFLAG_IBEXDOC	0x00000004	Ibex document.

Constant	Value	Description
GFLAG_PUBFOD	0x00000008	Published to Fax-on-Demand.
GFLAG_PUBWEB	0x00000010	Published to the Web.
GFLAG_FOLDER	0x00000020	Reserved.
GFLAG_CATALOGFOD	0x00000040	Is included in Fax-on-Demand catalog.
GFLAG_EMBARGOED	0x00000080	Document is embargoed until a specific date/time.
GFLAG_EXPIRES	0x00000100	Document expires on a specific date/time.
GFLAG_PUBLAN	0x00000200	Is published to LAN.
GFLAG_ISAUTOCAT	0x00000400	Set if it is an auto-generated catalog.
GFLAG_RESERVED2	0x80000000	If lib doc is set to be deleted (in GroupLibDoc context).

GRPFLAG_???

Constants for the flags field of GROUPINFO_10.

Constant	Value	Description
GRPFLAG_CONCURRENTLIMIT	0x00000001	The group has concurrent sending limits enabled.
GRPFLAG_MUSTHAVEFCS	0x00000002	Users in the group sending faxes will have cover sheets enabled automatically.
GRPFLAG_MUSTHOLDPREVIEW	0x00000004	Forces all users in the group to have faxes held for preview.
GRPFLAG_AGEALLFOLDERS	0x00000008	Automatically ages faxes in the user's folders.
GRPFLAG_LOCKDELEGATES	0x00000010	Prohibit changes to Delegates.
GRPFLAG_NEEDAPPROVALNOTIFY	0x00000020	Notify admin and alt admin of faxes needing approval.
GRPFLAG_NOTIFYINCLUDEFAX_RX	0x00000040	Include fax with email notifications of received faxes.
GRPFLAG_EXCLUSIVE	0x00000080	The group is an exclusive group.
GRPFLAG_SET_CONVERT_BIAS	0x00000100	Allow group members to set their own default conversion bias in FaxUtil.
GRPFLAG_EXCLUDEFROMARCHIVE	0x00000200	Exclude this group from archiving.
GRPFLAG_CANUPLOADAVATAR	0x00000400	If set, user can upload a user image to represent them in some user interfaces.
GRPFLAG_NOTIFYINCLUDEFAX_TX	0x00000800	Include the fax with sent notifications.

Constant	Value	Description
GRPFLAG_PURGE_RFARCHIVE	0x00001000	Purge Archive for this group.
GRPFLAG_MUSTVIEWALLPAGES	0x00002000	Group requirement to view all pages.
GRPFLAG_RESERVED1	0x40000000	Reserved.
GRPFLAG_RESERVED2	0x80000000	Reserved.

GRPSFDFLAG_???

Constants for the usSFDFlags field of GROUPINFO_10.

Constant	Value	Description
GRPSFDFLAG_ENABLED	0x0001	Smart fax distribution is enabled.
GRPSFDFLAG_NEEDSLOGIN	0x0002	User needs to log in to get faxes.

GRPSFDTYPE_???

Constants for the .usSFDType field of GROUPINFO_10.

Constant	Value	Description
GRPSFDTYPE_LINEAR	0x0000	Smart fax distribution is set to linear.
GRPSFDTYPE_BALANCED	0x0001	Smart fax distribution is set to balanced.

HISTDESC_???

Constants for the usOptions argument of the RFaxGetHistoryElementDescription () function.

Constant	Value	Description
HISTDESC_NONE	0x0000	
HISTDESC_SHOWSECONDS	0x0001	Display seconds in history transaction time.
HISTDESC_TIMEFIRST	0x0002	Display the time before the date.
HISTDESC_NOTIME	0x0004	Do not include the time stamp.

HISTPRINTFLAG_???

Constant	Value	Description
HISTPRINTFLAG_RETRY	0x0001	Used when a print attempt needs to be retried.

HISTROUTEFLAG_???

Constants for HISTLISTELEMENT0.d.physical.tr_rectype.

Constant	Value	Description
HISTROUTEFLAG_AUTOROUTE	0x0001	Fax was autorouted.
HISTXROUTEFLAG_SENTTOSERVER	0x0001	0 - From server listed 1 - To server listed

HISTTRXFLAG_???

Constants for usFlags d.physical.tr_rectype of HISTLISTELEMENT0.

Constant	Value	Description
HISTTRXFLAG_PARTIALRETRIES	0x0001	The send attempt had partial retries enabled. Correlates to ACTLOG_FLAG_PARTIALRETRY.
HISTTRXFLAG_SENTREMOTELY	0x0002	Fax was scheduled on another DocTransport via a Least Cost Routing rule. Correlates to ACTLOG_FLAG_SCHEDREMOTE.
HISTTRXFLAG_ANIVALID	0x0004	The received fax included ANI info that is recorded in the history record HISTLISTELEMENT0.d.trx.szANI field. Correlates to ACTLOG_HAS_ANI.
HISTTRXFLAG_AOCVALID	0x0008	The received fax included advice of charge (AOC) data that is recorded in the history record HISTLISTELEMENT0.d.trx.aulAOCData field. Correlates to ACTLOG_HAS_AOC.
HISTTRXFLAG_ECMVALID	0x0200	Error correction mode (ECM) was enabled for the call. Correlates to ACTLOG_FLAG2_HASECM.

HISTTYPE_???

Constants for the d.physical.tr_rectype field of HISTLISTELEMENT0 and the tr_rectype field of HISTLISTELEMENT2.

Constant	Value	Description
HISTTYPE_TXRXINFO	0	Contains transmission receive information.
HISTTYPE_ROUTE	1	Routing information.
HISTTYPE_PRINT	2	Printing information.
HISTTYPE_CONVERR	3	Conversion error.
HISTTYPE_OCR	4	OCR information.
HISTTYPE_NETFORWARD	5	Network forwarding information.
HISTTYPE_APPROVED	6	Fax is approved.
HISTTYPE_DENIED	7	Fax was denied.
HISTTYPE_FILEROUTED	8	Where the file was routed to.
HISTTYPE_SECUREDOCS	9	Fax was sent via SecureDocs.
HISTTYPE_INTERCONNECT	10	Interconnect source/destination.
HISTTYPE_GENERIC	11	Generic (default value).

ICONFLAG_???

Constants for the usFlags field of ICONINFO_10.

Constant	Value	Description
ICONFLAG_STAMP	0x0001	The image to use for stamping faxes.
ICONFLAG_AVATAR	0x0002	The image to use for the user or profile image of the owner.

ICONLISTELEMENT0

The ICONLISTELEMENT0 structure.

C Data Type	Member	Description
FAXHANDLE	handle	Handle of icon.
FAXHANDLE	hOwner	Handle of owning user or NULL if no owner.
LONG	tLastUpdated	Last time the icon was updated.
TCHAR	szDescription[64]	Description of icon.

IMAGEFMT_???

Constants for the usFileFormat fields of MSMAIL1REQ, CCMAIL1REQ, WPOMAIL1REQ, and GENMAIL1REQ, and the d.filesave1.usFileFormat field of WORKREQUEST.

Constant	Value	Description
IMAGEFMT_DCX	0	Format is type DCX.
IMAGEFMT_PCX	1	Format is type PCX.
IMAGEFMT_TIFF1	2	Format is type TIFF-G3.

LIBDOC_USAGE_???

Constants for the usUsageType argument of function RFaxAdjustLibDocUsage()the sIntendedUse argument of functionRFaxLoadLibDoc3(), and the ulLibDocUsage field of CVL_ATTACHITEM.

Constant	Value	Description
LIBDOC_USAGE_NONE	0x0000	Library document is not used.
LIBDOC_USAGE_FOD	0x0001	Library document is used in Docs-on-Demand.
LIBDOC_USAGE_WEB	0x0002	Library document is published to the Web.
LIBDOC_USAGE_LAN	0x0003	Library document is published to the LAN.
LIBDOC_USAGE_RESETALL	0x0100	Only used with ADJUSTLIBDOC command. Resets all counts.
LIBDOC_USAGE_RESETLUD	0x0200	Only used with ADJUSTLIBDOC command. Reset last used date as well.

LIBENUM_ADMINLOAD_???

Constants for the usAdminFlags argument of function RFaxGetLibDocList4().

Constant	Value	Description
LIBENUM_ADMINLOAD_IGNOREDATES	0x0001	Ignore expiration and embargo dates in search.
LIBENUM_ADMINLOAD_IGNOREUSE	0x0002	Ignore usage in search.

LIBENUMFLAGS_???

Constants for the ulLibEnumFlags argument of functions RFaxGetLibDocList3() and RFaxGetLibDocList4().

Constant	Value	Description
LIBENUMFLAGS_SEARCHONID	0x00000001	Perform search on the ID.
LIBENUMFLAGS_SEARCHONDESC	0x00000002	Perform search on the description.
LIBENUMFLAGS_SEARCHONIDANDDESC	0x00000004	Perform search on both the description and ID.
LIBENUMFLAGS_SEARCHONGROUP	0x00000008	Perform search on the group.

LOADCONN_KEY_???

Key for sWhichKey parameter to *RFaxLoadConnectionByKey()* on page 44.

Constant	Value	Description
LLOADCONN_KEY_NAME	1	Load connection with matching szName field.
LOADCONN_KEY_SERVICE	2	Load connection used by the specified service (for example, "eTransport").

LOADFAX_???

Constants for the usLoadImageOpts argument of functions RFaxLoadFax2(), RFaxLoadFax3(), and RFaxLoadFax4(), and the usOpts argument of function RFaxLoadFax4()

Constant	Value	Description
LOADFAX_LOADALL	0	Load both the image and the cover sheet.
LOADFAX_NOCOVER	1	Don't load the cover.
LOADFAX_NOBODY	2	Don't load the body.
LOADFAX_ANNOTATIONS	4	Load annotation data.

Constant	Value	Description
LOADFAX_CMDDATA	8	Loads command data file.
LOADFAX_FIRSTPAGEONLY	16	Forces load of first body page only.
LOADFAX_PDF	0x0020	Loads PDF body, if available.
LOADFAX_TXT	0x0040	Loads TXT body, if available. For SMS.
LOADFAX_ENHANCED	0x0080	Loads TIF body, if available. It is the Enhanced Image.

LOADGROUP_???

Constants for the usLoadOpt argument of function RFaxLoadGroup2().

Constant	Value	Description
LOADGROUP_QUICKER	0x0000	No extra data loaded.
LOADGROUP_USERCOUNT	0x0001	Causes the server to fill in the GROUPINFO_10.ulMembers field. This can impact performance on servers with many groups and/or users.
LOADGROUP_ADMINIDS	0x0002	Causes the server to fill in the GROUPINFO_10.admin and alt_admin fields. This can impact performance on servers with many groups and/or group monitors.

LOADLIB_???

Constants used in the usLoadOpt parameter to *RFaxLoadLibDoc4()* on page 215.

Constant	Value	Description
LOADLIB_NONE	0x00	
LOADLIB_ENFORCE_USAGE	0x01	When set, if the intended use does not match the allowed usage of the library document, then RFAXERR_LIBDOCINACCESSIBLE will be returned.
LOADLIB_ENFORCE_EMBARGO	0x02	When set, if the current time is outside of the embargo date of the library document, then RFAXERR_RFAXERR_LIBDOCEMBARGOED will be returned.
LOADLIB_ENFORCE_EXPIRATION	0x04	When set, if the current date is past the expiration date of the library document, then RFAXERR_LIBDOCEXPIRED will be returned.

Masks for LOADLIB

LOADLIB_ENFORCE_ALL (LOADLIB_ENFORCE_USAGE | LOADLIB_ENFORCE_EMBARGO | LOADLIB_ENFORCE_EXPIRATION)

LOADICONOPT_???

Constants for the usOptions argument of function RFaxLoadIcon().

Constant	Value	Description
LOADICONOPT_NONE	0x00	
LOADICONOPT_STAMP_OWNER	0x01	Handle passed to RFaxLoadIcon() is the owner of the personal stamp to load.
LOADICONOPT_AVATAR_OWNER	0x02	Handle passed to RFaxLoadIcon is the owner of the icon to load for displaying as their avatar.

LOADPHONEOPT_???

Constants for the usOptions argument of function RFaxLoadPhoneByName2().

Constant	Value	Description
LOADPHONEOPT_NONE	0x00	
LOADPHONEOPT_PARTIALMATCH	0x01	Allow partial matches. Returns first match.
LOADPHONEOPT_MATCHNAME	0x02	Allow matches by name in addition to ID.

LOADSTAMPOPT_???

Deprecated. Use *LOADICONOPT_??? above* instead.

Constants for the usOptions argument of function RFaxLoadStamp().

Constant	Value	Description
LOADSTAMPOPT_NONE	0x00	
LOADSTAMPOPT_OWNER	0x01	Handle passed to RFaxLoadStamp() is the owner of the stamp to load.

LOADUSER_???

Constants for the usLoadOpt argument of functions RFaxLoadUser4(), RFaxLoadUser5(), and RFaxLoadUser6().

Constant	Value	Description
LOADUSER_QUICKER	0x0000	No extra data loaded. usNumFaxesOwned and groupid will be empty,

Constant	Value	Description
LOADUSER_QUICK	0x0001	This is an alias for backward compatibility. Use LOADUSER_GROUPID instead. If you do NOT need the group id, use LOADUSER_QUICKER.
LOADUSER_GROUPID	0x0001	Loads a user aggregation via a view with groupid filled in. Only use when this field is needed . Default when you use RFaxLoadUser 2 or 3. This field is less of a performance issue than LOADUSER_FAXESOWNED.
LOADUSER_FAXESOWNED	0x0002	Loads a user aggregation via a view with usNumFaxesOwned filled in. This is a performance hit and should not be used when these fields are not needed.

LOCKFAXOPTION_???

Constants for optional flags of function *RFaxLockFax()* on page 99.

Constant	Value	Description
LOCKFAXOPTION_NONE	0x0000	
LOCKFAXOPTION_OVERRIDE	0x0001	Lock fax even if another user has it locked. Requires logged in user be full admin.

LOGDIR_???

Constants for the usLogicalServerDir argument of functions RFaxGetServerInfo2(), RFaxFileRead(), RFaxFileWrite(), RFaxFileList(), and RFaxFileDelete(), and the usLogServerSrcDir and usLogServerDstDir arguments of function RFaxFileCopy().

Constant	Value	Description
LOGDIR_NONE	0	Root of RightFax directories.
LOGDIR_IMAGE	1	RightFax\Image folder.
LOGDIR_SIG	2	RightFax\Sig folder.
LOGDIR_PAPERS	3	RightFax\Papers folder.
LOGDIR_FCS	4	RightFax\FCS folder.
LOGDIR_BIN	6	RightFax\Bin folder.
LOGDIR_OUTGOING	7	RightFax\Outgoing folder.
LOGDIR_BFT	8	RightFax\BFT folder.

Constant	Value	Description
LOGDIR_BOARD	9	RightFax\RFBoard folder.
LOGDIR_CMDDATA	10	RightFax\CMDDATA folder.
LOGDIR_DOCTRANSPORT	11	RightFax\DocTransport folder.
LOGDIR_DATABASE	12	RightFax\Database folder.
LOGDIR_IMAGE_ENHANCED	13	Enhanced image folder.
LOGDIR_IMAGE_ORIGINAL	14	Original image folder.
LOGDIR_CATALOG	15	RightFax\Catalog folder.
LOGDIR_CONFIGURATION	16	Configuration folder for configuration files common between collective servers and remote hosts.
LOGDIR_IMAGE_DISK	17	Only for the RFImageTool to be able to move files out of SQL and back onto disk if the admin enables and then backs out of SQL folders. Not for any other use.
LOGDIR_EDCARCHIVE	18	EDC Archive directory
LOGDIR_SCHED_REPORTS	19	Specify the directory to output the scheduled reports to.
LOGDIR_AUTO_UPDATES	20	Specify the folder to watch for automatic updates.
LOGDIR_WORKFLOWINTELLIGENCE	21	
LOGDIR_MAX	LOGDIR_ WORKFLOWINTELLIGENCE	

LOGINFLAGS_???

Constants for the ulFlags field of LOGININFO.

Constant	Value	Description
NONE	0x0000	For comparisons.
AUTHEN_REQ	0x0001	User requires NT authentication.
ISGROUPADMIN	0x0002	User is a group monitor.
ISADMINREADONLY	0x0004	User is a read-only administrator.
ISUSERMANAGEMENTADMIN	0x0008	User is a user management administrator.

Constant	Value	Description
ISADMIN	0x0010	User is a full administrator.

Masks for LOGINFLAGS

MASK_ISANYADMIN = (ISADMIN|ISUSERMANAGEMENTADMIN|ISADMINREADONLY|ISGROUPADMIN), User is any type of admin including Group Monitor.

MASK_ISANYNONGROUPADMIN = (ISADMIN|ISUSERMANAGEMENTADMIN|ISADMINREADONLY), Admin only.

LSTFLAG__

Constants for the ulLstFlags field of RECIPIENTLISTELEMENT0.

Constant	Value	Description
LSTFLAG_HIDEDESTFROMCC	0x00000001	Hide destination info on carbon-copied faxes.
LSTFLAG_MIMESENDDOC	0x00000002	The recipient is a MIME(or email) recipient.
LSTFLAG_CERTIFIEDDOC	0x00000004	The recipient is to receive a certified delivery document.
LSTFLAG_SMSDOC	0x00000008	The recipient is to receive an SMS message.
LSTFLAG_DONOTPROCESS	0x00000010	Mark a list element as unused.

MARKFAX_???

Constants for the usMarkCmd argument of functions RFaxMarkFax() and RFaxMarkFax2().

Constant	Value	Description
MARKFAX_VIEWED	0	Fax is viewed.
MARKFAX_UNVIEWED	1	Fax is not viewed.
MARKFAX_PRINTED	3	Fax is printed.
MARKFAX_NOTPRINTED	4	Fax is not printed,
MARKFAX_RELEASED	5	Fax is released (it was held for preview),
MARKFAX_HELD	6	Fax is held for preview,
MARKFAX_BODYPRINTED	7	The fax body was printed,

Constant	Value	Description
MARKFAX_APPROVED	8	The fax is approved,
MARKFAX_DISAPPROVED	9	The fax is not approved,
MARKFAX_GENERIC1	10	Generic user define flag,
MARKFAX_NOTGENERIC1	11	Generic user define flag,
MARKFAX_GENERIC2	12	Generic user define flag,
MARKFAX_NOTGENERIC2	13	Generic user define flag,
MARKFAX_FULLY_VIEWED	14	All pages of the fax have been viewed.
MARKFAX_PARTLY_VIEWED	15	Not all pages of the fax have been viewed.
MARKFAX_ATOMIC	0x4000	OR-on to MARKFAX_??? constant to force RFaxMarkFax() to work in atomic mode,

MSGCLASS_???

Constants for the msgclass.

Constant	Value	Description
MSGCLASS_SYSDEF	0	
MSGCLASS_NONE	1	
MSGCLASS_ADV	3	

MSGTYPE_???

Constants for the sMsgType fields of MESSAGEREQUEST, MSMAIL1REQ, CCMAIL1REQ, WPOMAIL1REQ, GENMAIL1REQ, GENMAIL2REQ, and GENMAILPREVIEWREQ, and the d.mail1.sMsgType field of WORKREQUEST.

Constant	Value	Description
MSGTYPE_SENDOK	0	Fax was sent successfully,
MSGTYPE_CHKLOOK2	1	The message has been created but has not yet been processed,
MSGTYPE_SENDERR1	2	Fax received an error,
MSGTYPE_SENDERR2	3	Fax received an error,

Constant	Value	Description
MSGTYPE_SENDING1	4	Fax is sending,
MSGTYPE_SENDING2	5	Fax is sending,
MSGTYPE_NEWFAX1	6	A new fax has arrived,
MSGTYPE_NEWFAX2	7	A new fax has arrived,
MSGTYPE_ABANDON1	8	Fax was abandoned after too many tries,
MSGTYPE_MESSAGE1	9	The message has been converted but the cover sheet has yet to be created,
MSGTYPE_CODEERR1	10	The fax reported an error in billing code validation,
MSGTYPE_PREVIEW1	11	Fax is held for preview,
MSGTYPE_OCRDONE1	12	OCR is completed,
MSGTYPE_PAPERERR1	13	Overlaying failed,
MSGTYPE_PHONEERR1	14	Phonebook error,
MSGTYPE_TO_RECIPIENT	17	

NOTIFY_RCV_???

Constants for the receive_notify fields of USERINFO_10 and USERINFO_11.

Constant	Value	Description
NOTIFY_RCV_NEW1	0x0001	Send notification message only once when a new fax is received.
NOTIFY_RCV_NEW2	0x0002	Send chklook message periodically,
NOTIFY_RCV_NEW3	0x0004	Send notification message when a new fax is received, and send chklook message periodically.

NOTIFY_SND_???

Constants for the send_notify fields of USERINFO_10 and USERINFO_11.

Constant	Value	Description
NOTIFY_SND_NEEDS_FCS1	0x0001	Send message only for first time needs_fcs,
NOTIFY_SND_NEEDS_FCS2	0x0002	Send needs_fcs message periodically,

Constant	Value	Description
NOTIFY_SND_SENDING1	0x0004	Send message about sending first time only,
NOTIFY_SND_SENDING2	0x0008	Send message about sending periodically,
NOTIFY_SND_SENDRETRY	0x0010	Send message about sending errors and retry,
NOTIFY_SND_SENDOK	0x0020	Send message about sending OK,
NOTIFY_SND_SENDERR	0x0040	Send message about sending errors & abandon,
NOTIFY_SND_NOPREVIEW	0x0080	Do not send message about faxes held for preview,
NOTIFY_SND_APPROVAL	0x0100	Send message about held for approval faxes,

NOTIFY_TYPE_???

Constants for the notify_type fields of GROUPINFO_10, USERINFO_10, USERLISTELEMENT1, and USERLISTELEMENT2, the ucRecipNotifyType field of RECIPIENTLISTELEMENT0, and the usNotifyType argument of function RFaxGetMessageEvent().

Note To use custom notifications, Network Messaging must be turned off in the Services list on all WorkServers.

Constant	Value	Description
NOTIFY_TYPE_NETBROADCAST	0	Notification is set to network broadcast,
NOTIFY_TYPE_GROUPMETHOD	0	
NOTIFY_TYPE_CUSTOM1	1	Custom 1,
NOTIFY_TYPE_CUSTOM2	2	Custom 2,
NOTIFY_TYPE_CUSTOM3	3	Custom 3,
NOTIFY_TYPE_CUSTOM4	4	Custom 4,
NOTIFY_TYPE_CUSTOM5	5	Custom 5,
NOTIFY_TYPE_CUSTOM6	6	Custom 6,
NOTIFY_TYPE_CUSTOM7	7	Custom 7,
NOTIFY_TYPE_CUSTOM8	8	Custom 8,
NOTIFY_TYPE_CUSTOM9	9	Custom 9,
NOTIFY_TYPE_CCMAIL	10	cc:Mail,

Constant	Value	Description
NOTIFY_TYPE_MSMAIL	11	MS-Mail,
NOTIFY_TYPE_GROUPWISE	12	Groupwise,
NOTIFY_TYPE_TRS	13	TRS,
NOTIFY_TYPE_CX3	14	CallXPressFax 3,
NOTIFY_TYPE_PAGER	15	Pager,
NOTIFY_TYPE_NOTES	16	Lotus Notes,
NOTIFY_TYPE_EXCHANGE	17	Microsoft Exchange,
NOTIFY_TYPE_SMTP	18	SMTP,
NOTIFY_TYPE_SAP	19	SAP,
NOTIFY_TYPE_SMS	20	SMS,
NOTIFY_TYPE_ANY	0xFFFF	Used only in calls to RFaxGetMessageEvent(). Signifies that the function should return any type of message event rather than one or more specific types,

NQFLAGS_???

Constants for the.ulNotifyQueueFlags field in FAXINFO_11.

Constant	Value	Description
NQFLAGS_NONE	0x00000000	
NQFLAGS_ONSUCCESS	0x00000001	Post a message to the queue at FAXINFO_1?.szNotifyQueue on success of an outbound fax.
NQFLAGS_ONFAILURE	0x00000002	Post a message to the queue at FAXINFO_1?.szNotifyQueue on final failure of an outbound fax.

OCRFLAGS_???

Constants for the autoocr_flags fields of USERINFO_10 and USERINFO_11, and the usOCRFlags field of OCRREQ.

Constant	Value	Description
OCRFLAGS_ROUTEMSMAIL	0x0001	Route the fax to MS-Mail after OCR is done.
OCRFLAGS_ROUTECCMAIL	0x0002	Route the fax to cc:Mail after OCR is done.

Constant	Value	Description
OCRFLAGS_CLEANUP	0x0004	Image should be cleaned up before OCR.
OCRFLAGS_MESSAGEUSER	0x0008	Deprecated. Request a notification message that the OCR was done.
OCRFLAGS_ROUTEWPOMAIL	0x0010	Route the fax to GroupWise after OCR is done.
OCRFLAGS_PRINTFAX	0x0020	Print the fax after OCR is done.
OCRFLAGS_ROUTEGENMAIL	0x0040	Route the fax to generic email after OCR is done.
OCRFLAGS_ROUTEFILE	0x0080	Auto OCR FileRoute.
OCRFLAGS_FCS	0x0100	OCR the FCS.
OCRFLAGS_KEEPSTATUS	0x0200	Don't update the fax status with the OCR operation result.
OCRFLAGS_SENDOCR	0x0400	This is a blocking outbound OCR. The fax will not send until the OCR operation has completed.

OCRFORMAT_???

Constants for the autoocr_format fields of USERINFO_10 and USERINFO_11, for the sFormat field of OCRREQ and the sFormat argument of function RFaxOCRFax().

Constant	Value	Description
OCRFORMAT_SMARTASCII	0	Approximates the original format in ASCII.
OCRFORMAT_ASCII	1	Produce a plain unformatted ASCII text file.
OCRFORMAT_RTF	2	Preserve fonts and formatting in a Windows RTF file. Only available for Windows clients.

OCRLAYOUT_???

Constants for the autoocr_layout fields of USERINFO_10 and USERINFO_11, for the sLayout field of OCRREQ and the sLayout argument of function RFaxOCRFax().

Constant	Value	Description
OCRLAYOUT_WYSIWYG	0	OCR will attempt to interpret the text in the format it appears in. For example, if the document contains three columns, the text will be interpreted in column form.
OCRLAYOUT_LEFTJUSTIFIED	1	The text will be interpreted as left-justified text.

OCRSTAT_???

Constants for the body_ocr_status and fcs_ocr_status fields in FAXINFO_# and FAXLISTELEMENT#.

Constant	Value	Description
OCRSTAT_NONE	0	
OCRSTAT_QUEUED	1	
OCRSTAT_INPRINT	2	
OCRSTAT_DONE	3	

PAPERSIZE_???

Constants for the sDefSize fields of PRINTERINFO_10 andPRINTERINFO_10.

Constant	Value	Description
PAPERSIZE_FITLETTER	0	Default paper size is to fit US letter.
PAPERSIZE_FITLEGAL	1	US legal.
PAPERSIZE_FITA4	2	European A4.
PAPERSIZE_FITLETLEG	3	Fit letter or legal (the best fit will be used).
PAPERSIZE_FITA4LEG	4	Fit A4 or legal.
PAPERSIZE_NOSCALE	5	Do not scale the fax image.

PAPERSOURCE_???

Constants for the sDefSource fields of PRINTERINFO_10 and PRINTERLISTELEMENT0.

Constant	Value	Description
PAPERSOURCE_NONE	0	No default paper source.
PAPERSOURCE_UPPERTRAY	1	Tray 1.
PAPERSOURCE_LOWERTRAY	2	Tray 2.
PAPERSOURCE_MANUAL	3	Manual feed.

Constant	Value	Description
PAPERSOURCE_TRAY3	4	Tray 3.
PAPERSOURCE_MPTRAY	5	Multi-purpose tray.
PAPERSOURCE_HIGHCAP	6	High capacity tray.

PHNGRPEXT_???

Constants for the PHONEGRP extension in EXTENSIONINFO_10

Constant	Value	Description
PHNGRPEXT_KEYCOUNT	77	Number of phone book entries in a PHONEGRP Extension.

PHONEFLAG_???

Constants for the flags fields of PHONEITEM, PHONELISTELEMENT0, and PHONELISTELEMENT1.

Constant	Value	Description
PHONEFLAG_LIST	0x00000001	Set if this a list of other phone_items.
PHONEFLAG_PUBLISHED	0x00000002	Set if this entry is published for use throughout the network.
PHONEFLAG_EXTERNALPUB	0x00000004	Set if entry should be shadowed to external system.
PHONEFLAG_READONLY	0x00000008	If set, only owner can modify.
PHONEFLAG_RESERVED2	0x00000040	Reserved.
PHONEFLAG_RESERVED1	0x00000080	Reserved.
PHONEFLAG_HASEXTENSIONS	0x00000100	Phone book entry has extensions.
PHONEFLAG_HIDEFROMCCLIST	0x00000200	This entry should never be shown on a CC-List.
PHONEFLAG_CERTIFIED	0x00000400	Use certified delivery for this recipient.
PHONEFLAG_EMAIL	0x00000800	This is an email recipient.
PHONEFLAG_SMS	0x00001000	SMS type.

PRINTDUPLEX_???

Constants for the ucRPO_Duplex fields of USERINFO_10 and USERINFO_11, and for the d.extra1.ucDuplex field of AUTOPRINTREQ.

Constant	Value	Description
PRINTDUPLEX_DEFAULT	0	Use default duplexing option.
PRINTDUPLEX_SINGLE	1	Print single-sided.
PRINTDUPLEX_LONGEDGE	2	Flip page on long edge.
PRINTDUPLEX_SHORTEDGE	3	Flip page on short edge.

PRINTERFLAGS_???

Constants for the flags fieldF of PRINTERINFO_10 and PRINTERLISTELEMENT0.

Constant	Value	Description
PRINTERFLAGS_DIRECTIP	0x00000002	Direct IP connection to the printer.

PRINTERTYPE_???

Constants for the printertype fields of PRINTERINFO_10 and PRINTERLISTELEMENT0.

Constant	Value	Description
PRINTERTYPE_NONE	0	Printer type is set to None.
PRINTERTYPE_LJ	1	LaserJet.
PRINTERTYPE_LJII	2	LaserJet II.
PRINTERTYPE_DJ	3	DeskJet.
PRINTERTYPE_LJIII	4	LaserJet III.
PRINTERTYPE_DJ500	5	DeskJet 500.
PRINTERTYPE_PS1	6	PostScript level 1.
PRINTERTYPE_LJ4	7	LaserJet 4.
PRINTERTYPE_PCL	7	LJ4 as the general PCL choice. Effectively, this is a synonym for PRINTERTYPE_LJ4.
PRINTERTYPE_PPDS	8	IBM 4019/4029.
PRINTERTYPE_XIP	9	Xionics XipPrint.
PRINTERTYPE_DCS20	10	Xerox DCS 20.

Constant	Value	Description
PRINTERTYPE_DCS35	11	Xerox DCS 35.
PRINTERTYPE_LJ3SI	12	LaserJet IIIsi.
PRINTERTYPE_LJ4SI	13	LaserJet 4si.
PRINTERTYPE_LJ5SI	14	LaserJet 5si.
PRINTERTYPE_ATI	15	ATI printers.
PRINTERTYPE_LJ5SI_MOPIER	16	HP LI5SI Mopier.
PRINTERTYPE_XRXN24	17	Xerox N24.
PRINTERTYPE_DCS220	18	Xerox DCS 220.
PRINTERTYPE_DCS230	19	Xerox DCS 230.
PRINTERTYPE_DCS240	20	Xerox DCS 240.
PRINTERTYPE_DCS255	21	Xerox DCS 255.
PRINTERTYPE_DCS265	22	Xerox DCS 265.
PRINTERTYPE_RICOH250	23	Ricoh Aficio 250.
PRINTERTYPE_LJ8000	24	HP LaserJet 8000.
PRINTERTYPE_GDI	25	GDI printers.
PRINTERTYPE_DCS460	26	Xerox DCS 460.
PRINTERTYPE_DCS470	27	Xerox DCS 470.
PRINTERTYPE_DCS480	28	Xerox DCS 480.
PRINTERTYPE_DCS490	29	Xerox DCS 490.
PRINTERTYPE_MAXNUM	29	

PRINTFLAG_???

Constants for the flags field of AUTOPRINTREQ. Also see the synonymous PRINTFLAGS_???.

Constant	Value	Description
PRINTFLAG_TRX	0x01	Print transmission history.

Constant	Value	Description
PRINTFLAG_STAPLE	0x02	Staple output.
PRINTFLAG_COLLATE	0x04	Collate output.
PRINTFLAG_NEW	0x08	New style autoprint. Use extra1 data instead of szUserID in union.
PRINTFLAG_AUTOPRINT_SENT	0x10	Autoprinting a sent fax.
PRINTFLAG_NO_STATUS_CHANGE	0x20	Do not change the fax status to reflect when printing.
PRINTFLAG_PRINTTIFFHDR	0x40	Print TIF file header field values.
PRINTFLAG_NOPRINTSCALING	0x80	Image not scaled when printing,

PRINTFLAGS_???

These constants are synonymous with PRINTFLAG_???, but those constants were inconsistently named. Server-side components were using these names.

Constant	Value	Description
PRINTFLAGS_TRX	0x0001	Print transmission history.
PRINTFLAGS_STAPLE	0x0002	Staple output.
PRINTFLAGS_COLLATE	0x0004	Collate output.
PRINTFLAGS_NEW	0x0008	New style autoprint. Use extra1 data instead of szUserID in union.
PRINTFLAGS_AUTOPRINT_SENT	0x0010	Autoprinting a sent fax.
PRINTFLAGS_NO_STATUS_CHANGE	0x0020	Do not change the fax status to reflect when printing.
PRINTFLAGS_PRINTTIFFHDR	0x0040	Print TIF file header field values.
PRINTFLAGS_NOPRINTSCALING	0x0080	Image not scaled when printing,

PRINTJOBTYPE_???

Constants for the printjobtype fields of FAXINFO_10 and FAXINFO_11.

Constant	Value	Description
PRINTJOBTYPE_EPSON	0	The type of print job is EPSON.

Constant	Value	Description
PRINTJOBTYPE_PCL	1	The type of print job is PCL-5.
PRINTJOBTYPE_PS	2	The type of print job is PostScript.
PRINTJOBTYPE_PCL2	3	The type of print job is PCL-5 (alt).
PRINTJOBTYPE_PS2	4	The type of print job is PScript (alt).
PRINTJOBTYPE_CVL	5	The type of print job is CVL (conversion list).
PRINTJOBTYPE_TIF	6	The type of print job is TIFF.

PRINTOUTPUT_???

Constants for the d.extra1.ucOutputBin field of AUTOPRINTREQ and the ucRPO_OutputBin field of USERINFO_10.

Constant	Value	Description
PRINTOUTPUT_DEFAULT	0	Print to default tray.
PRINTOUTPUT_UPPER	1	Print to upper tray.
PRINTOUTPUT_LOWER	2	Print to lower tray.
PRINTOUTPUT_FINISHER	3	Print to Finisher.
PRINTOUTPUT_MAILBOXFIRST	4	Values between and including MAILBOXFIRST and MAILBOXLAST are valid.
PRINTOUTPUT_MAILBOXLAST	27	Values between and including MAILBOXFIRST and MAILBOXLAST are valid.

PRINTRES_???

Constants for the res field of AUTOPRINTREQ.

Constant	Value	Description
PRINTRES_HIGH	0	Print high resolution.
PRINTRES_MEDIUM	1	Print medium resolution.
PRINTRES_LOW	2	Print low resolution.

PRINTSIZE_???

Constants for the size field of AUTOPRINTREQ, indicating the paper size.

Constant	Value	Description
PRINTSIZE_DEFAULT	0	Printer default.
PRINTSIZE_LETTER	1	Letter.
PRINTSIZE_LEGAL	2	Legal.
PRINTSIZE_A4	3	A4.
PRINTSIZE_LETTERLEGAL	4	Best fit between letter and legal.
PRINTSIZE_A4LEGAL	5	Best fit between A4 and legal.

PRINTSOURCE_???

Constants for the source field of AUTOPRINTREQ.

Constant	Value	Description
PRINTSOURCE_DEFAULT	0	Default paper source.
PRINTSOURCE_UPPER	1	Tray 1.
PRINTSOURCE_LOWER	2	Tray 2.
PRINTSOURCE_MANUAL	3	Manual feed.
PRINTSOURCE_TRAY3	4	Tray 3.
PRINTSOURCE_MPTRAY	5	Tray 4.
PRINTSOURCE_HIGHCAPACITY	6	Tray 5.

PRINTSTAT_???

Constants for the print_status fields in FAXINFO_# and FAXLISTELEMENT#.

Constant	Value	Description
PRINTSTAT_NONE	0	
PRINTSTAT_QUEUED	1	
PRINTSTAT_INPRINT	2	
PRINTSTAT_DONE	3	

PRIORITY_???

Constants for the ucPriority fields of FAXINFO_10, FAXINFO_11, and RECIPIENTLISTELEMENT0, and the ucDefaultPriority fields of USERINFO_10 and USERINFO_11.

Constant	Value	Description
PRIORITY_NORMAL	0	The priority of the fax is normal.
PRIORITY_LOW	1	The priority of the fax is low.
PRIORITY_HIGH	2	The priority of the fax is high.

REQFIELD_???

The REQFIELD_IDX_??? flags are used to index into acReqFields fields of SERVERINFO and SERVERINFO2.

The REQFIELD_??? flags are used to set the appropriate behavior. For example, acReqFields[REQFIELD_IDX_TO_NAME] = REQFIELD_SEND;]

Constant	Value	Description
REQFIELD_IDX_NONE	-1	See RFaxFcsDlg1ToReqFieldIndex() and RFaxFcsDlg2ToReqFieldIndex().
REQFIELD_IDX_TO_NAME	0	
REQFIELD_IDX_TO_FAXNUM	1	
REQFIELD_IDX_TO_VOICENUM	2	
REQFIELD_IDX_TO_COMPANY	3	
REQFIELD_IDX_TO_CITYSTATE	4	
REQFIELD_IDX_FROM_NAME	5	
REQFIELD_IDX_FROM_VOICENUM 6	6	
REQFIELD_IDX_FROM_FAXNUM	7	
REQFIELD_IDX_FROM_OPERATORNUM	8	
REQFIELD_IDX_FROM_GENERALNUM	9	
REQFIELD_IDX_BILLCODE1	10	
REQFIELD_IDX_BILLCODE2	11	

Constant	Value	Description
REQFIELD_NEITHER	0x0	Field is not required on sent or received faxes.
REQFIELD_SEND	0x1	Required on sending faxes.
REQFIELD_RECEIVE	0x2	Required on received faxes.
REQFIELD_BOTH	REQFIELD_SEND REQFIELD_RECEIVE	Required on both sending and received faxes.

REQUEST_???

Constants for the ulRequestType field of ARCHIVEREQUEST, COMPLETEREQUEST, MESSAGEREQUEST, VALIDATEREQUEST, and WORKREQUEST, the dwMessageTypes arguments of functions RFaxGetGenericEvent() and RFaxPurgeEvents(), the ulRequestType members of ARCHIVEREQUEST, COMPLETEREQUEST, and COMPLETEREQUEST2, and the ulReqFilter argument of function RFaxGetWorkRequestCounts ().

Constant	Value	Description
REQUEST_CONVERTPCL5	0x00000001	Convert to PCL-5 request.
REQUEST_DELETEFILES	0x00000002	Delete file request.
REQUEST_PRINTFAX	0x00000004	Print fax request.
REQUEST_OVERLAY	0x00000008	Overlay request.
REQUEST_CONVERTPS	0x00000010	Convert to PostScript request.
REQUEST_MSMAILROUTE	0x00000020	Route to MS-Mail request.
REQUEST_MSMAILMSG	0x00000040	MS-Mail notification request.
REQUEST_CCMAILROUTE	0x00000080	Route to cc:Mail.
REQUEST_CCMAILMSG	0x00000100	cc:Mail notification request.
REQUEST_ARCHIVEFAX	0x00000200	Archive sent fax request.
REQUEST_FILEROUTE	0x00000400	Route file to network folder.
REQUEST_MAKEFCS	0x00000800	Make a cover sheet.
REQUEST_FODCAT	0x00001000	Request FOD (fax on demand) catalog.
REQUEST_OCRFAX1	0x00004000	OCR fax.

Constant	Value	Description
REQUEST_OCRROUTE	0x00008000	OCR route request.
REQUEST_CONVERTCVL	0x00010000	Convert a CVL (conversion list).
REQUEST_WPOMAILROUTE	0x00020000	Groupwise route request.
REQUEST_WPOMAILMSG	0x00040000	Groupwise notification request.
REQUEST_LIBSYNC	0x00080000	Sync library document.
REQUEST_VALIDATE	0x00100000	Validate fax against the billing codes.
REQUEST_NETMSG	0x00200000	Send a network message.
REQUEST_XROUTE	0x00800000	Route the fax to a different RightFax server.
REQUEST_GENMAILRTE	0x01000000	General mail routing request.
REQUEST_GENMAILMSG	0x02000000	General mail notification request.
REQUEST_CXREPLY	0x04000000	A reply message to be sent back to a CallXpress server.
REQUEST_COMPLETEEVENT	0x08000000	Completion event request.
REQUEST_EMAILPREVIEW	0x10000000	Email preview request.
REQUEST_XDCSUBMIT	0x20000000	
REQUEST_CONVERT	0x40000000	

REQUESTFLAGS_???

Bump bit field flags for masking required bits from SERVERINFO2.ulBumpBits.

Constant	Value	Description
REQUESTFLAGS_IDMASK	0xFF000000	Used to mask off other bits in order to determine if REQUESTFLAGS_ID is valid.
REQUESTFLAGS_ID	0xCF000000	Valid request to workserver.

RFAXERR_???

Return values from RightFax API functions.

Constant	Value	Description
RFAX_SUCCESS	0	The function completed successfully.
RFAXERR_NOTFOUND	2	File or object not found.
RFAXERR_PATHNOTFOUND	3	Path not found.
RFAXERR_INVALID_HANDLE	6	Invalid RFAXSERVERHANDLE passed to function.
RFAXERR_NOT_ENOUGH_MEMORY	8	Insufficient memory to complete the requested command.
RFAXERR_INVALID_PARAMETER	87	An invalid parameter was specified.
RFAXERR_FILE_TOO_LARGE	223	The server rejected the file because it was too large. Currently used for stamps, icons, and avatars but could be used for any uploaded file in the future.
RFAXERR_INVALID_OWNERID	10000	An invalid owner ID was specified.
RFAXERR_CANNOTCREATEFILE	10001	Cannot create the file. Check rights and disk space.
RFAXERR_CANNOTOPENIMAGE	10002	Cannot open image file specified. Possibly corrupt.
RFAXERR_BADRFG3FORMAT	10003	Bad RightFax Group-3 image format. Check image format.
RFAXERR_IOERROR	10004	General I/O error.
RFAXERR_LIST_EMPTY	10005	The specified list contains for information.
RFAXERR_LIST_END	10006	The end of the list was specified.
RFAXERR_INVALID_USERID	10007	The user ID specified is not valid.
RFAXERR_USER_ABORTED	10008	The user aborted the function.
RFAXERR_IPCINITFAILURE	10009	RFIPC???.NDR initialization failed.
RFAXERR_IPCSETNAMEFAILURE	10010	RFIPC???.NDR SetServerName failed.
RFAXERR_NDRNOTFOUND	10011	The RFIPC???.NDR cannot be found or loaded (dependent DLL missing?).
RFAXERR_NOTSUPPORTED	10012	The API call made is not supported on the current platform.
RFAXERR_IPXOS2NOTSUPPORTED	10013	The IPX-OS/2 protocol is only supported with the Windows 16-bit DLL.
RFAXERR_BADTIFFFORMAT	10014	The TIFF format is bad or invalid.
RFAXERR_INVALID_GROUPID	10015	The group ID specified is invalid.
RFAXERR_INVALID_SIGNID	10016	The signid field is invalid.

Constant	Value	Description
RFAXERR_INVALID_SIGNOWNER	10017	The signowner field is invalid.
RFAXERR_INVALID_FORMID	10018	The code field is invalid.
RFAXERR_INVALID_PRINTERID	10019	The NetPrint_ID field is invalid.
RFAXERR_INVALID_OBJECTID	10020	The ObjectID field is invalid.
RFAXERR_INVALID_SERVICEID	10021	The ServiceID field is invalid.
RFAXERR_INVALID_LEVEL	10022	The level of data supplied to function is not compatible with other supplied arguments.
RFAXERR_NDROUTDATED	10023	The NDR library loaded does not support the requested protocol or option.
RFAXERR_UNEQUAL_LISTS	10024	Operation cannot be performed because lists are of unequal size/type.
RFAXERR_PERMANENTFOLDER	10025	The operation cannot be performed on a permanent folder.
RFAXERR_BADFOLDERID	10026	No such folder belongs to this user.
RFAXERR_FOLDERNOTEMPTY	10027	Cannot perform this operation on a folder with faxes in it.
RFAXERR_NOMOREFOLDERS	10028	The maximum number of folders has been reached.
RFAXERR_NOFOLDERSPACE	10029	There is not enough free folder name storage space left for this operation.
RFAXERR_FOLDERALREADYEXISTS	10030	The folder already exists.
RFAXERR_FOLDERHASFOLDERS	10031	Cannot perform this operation on a folder with subfolders.
RFAXERR_RFSTATNOTFOUND	10032	RFStat.dll not found.
RFAXERR_RFSTATOUTDATED	10033	RFStat.dll found does not support the requested function.
RFAXERR_DUPLIBDOC	10034	Duplicates lib doc name.
RFAXERR_BADHANDLE	-6	Handle does not point to valid object.
RFAXERR_USERNOTEMPTY	-1012	Cannot delete user, because the user has phonebook entries or faxes.
RFAXERR_NOUSERS	-1013	No users available.
RFAXERR_CANTDELETE	-1014	Cannot delete administrator or supervisor.
RFAXERR_NOTCOMPLETE	-1015	Cannot delete received fax, because it's not complete.
RFAXERR_MFP_GUEST	-1016	Cannot delete user linked as MFP guest account.
RFAXERR_GROUPNOTEMPTY	-1021	Cannot delete group, because it has members.

Constant	Value	Description
RFAXERR_CANTDELETESELF	-1022	Cannot delete yourself when logged in as an admin.
RFAXERR_CANTMODIFYSELF	-1023	Cannot change your ADMIN status yourself when logged in as an admin.
RFAXERR_DISABLEDACCT	-1024	Account disabled, connection denied.
RFAXERR_NOSUCHFILES	-1025	The fax had none of the requested files. For example, the cover sheet was requested but it did not have a cover. This is different from RFAXERR_NOTFOUND where an attempt was made to download the cover but it was missing.
RFAXERR_NOTPHONEENT	-1032	Specified phonebook entry is not an entry.
RFAXERR_NOTPHONELIST	-1033	Specified phonebook list is not a list.
RFAXERR_BLANKUSERID	-1034	Can't save a phone entry with an empty user ID.
RFAXERR_BLANKENTRYID	-1035	Can't save a phone entry with an empty entry ID.
RFAXERR_CANTFORWARD	-1040	Unable to forward the fax.
RFAXERR_FAILLOADRFLDLL	-1041	Unable to load
RFAXERR_BADENUM	-9900	A bad fax number was specified.
RFAXERR_ACCESSDENIED	-9901	Non-administrative user attempting to log in to FaxAdmin.
RFAXERR_BADPASSWORD	-9902	User ID valid, password invalid.
RFAXERR_BADKEY	-9903	User ID, printer ID, etc., not found.
RFAXERR_NullHANDLE	-9904	Invalid object handle (FAXHANDLE), cannot be Null.
RFAXERR_DUPKEY	-9905	The entry specified is a duplicate.
RFAXERR_BADPARM	-9906	A bad argument was specified.
RFAXERR_FAXDELETED	-9907	The fax is deleted.
RFAXERR_COPYFAILED	-9908	The copy failed.
RFAXERR_OVERUSERLIMIT	-9909	The maximum number of users has been reached.
RFAXERR_LOCKED	-9910	The specified semaphore is already locked.
RFAXERR_COMMANDNOTSUPT	-9912	Server doesn't support the command (possibly an old server version).
RFAXERR_ALREADYMARKED	-9913	The fax was already marked.
RFAXERR_REGWRITEERROR	-9914	Error writing to the registry.

Constant	Value	Description
RFAXERR_REGREADERROR	-9915	Error reading from the registry.
RFAXERR_PRODLICVIOLATION	-9916	Attempt to use a production feature without proper license.
RFAXERR_AUTHENTICATE_FAILURE	-9917	Tried and failed to discover the NT identity of client on server.
RFAXERR_AUTHENTICATION_REQ	-9918	NT authentication was set but user ID was provided. User ID should not be specified when using NT authentication.
RFAXERR_OCRCLICVIOLATION	-9919	Tried to use OCR without license.
RFAXERR_OCRRLICVIOLATION	-9920	Tried to use OCR routing without license.
RFAXERR_PDFLICVIOLATION	-9921	Tried to use PDF/PS functionality without license.
RFAXERR_SECUREAPIVIOLATION	-9922	Security is enabled, but this user has not yet been authenticated via a VerifyUser call.
RFAXERR_SOAP_PROXY_INIT	-9923	Unable to initialize the Soap proxy.
RFAXERR_SOAP_PROXY_ERROR	-9924	There was an error in the soap proxy layer.
RFAXERR_GENLICVIOLATION	-9925	Tried to use a feature without license.
RFAXERR_DBALREADYLOCKED	-9926	The database is already locked.
RFAXERR_DBLOCKED	-9927	The database is currently locked.
RFAXERR_BADUSERID	-9928	Bad user ID.
RFAXERR_NOTENABLED	-9929	Indicates a feature is not enabled on the server.
RFAXERR_NOSUCHAUTOFORWARDUSER	-9930	Bad autoforward user ID.
RFAXERR_STRONGPASSWORDLEN	-9931	Password does not satisfy length requirement for strong password.
RFAXERR_STRONGPASSWORDSPECIAL	-9932	Password does not satisfy special character requirement for strong password.
RFAXERR_STRONGPASSWORDCASE	-9934	Password does not satisfy case requirement for strong password.
RFAXERR_FAXNUMBERRESTRICTED	-9935	The fax number cannot be used by the members of this group.
RFAXERR_UNPROTECTEDUSER_VIOLATION	-9936	An unprotected user attempted to modify a protected entity.
RFAXERR_NOTCONFIGUREDFORNTAUTH	-9937	An attempt was made to log in using NT auth for a user not configured for NT authentication.
RFAXERR_CANTDISABLELASTADMIN	-9938	Cannot disable the last administrator.
RFAXERR_CANTDELETELASTADMIN	-9939	Cannot delete the last administrator.

Constant	Value	Description
RFAXERR_UNSUPPORTEDCLIENTVERSION	-9940	Caller is below minimum client version that is allowed to talk to RFDB.
RFAXERR_DATABASEDOWN	-9941	Persistent issues accessing the database. API operations may be stalled.
RFAXERR_MUSTBEADMINFORPERMISSION	-9942	Must be an administrator for this permission.
RFAXERR_LIBDOCEXPIRED	-9943	Library document cannot be used because it is expired.
RFAXERR_LIBDOCEMBARGOED	-9944	Library document cannot be used because it is embargoed.
RFAXERR_LIBDOCINACCESSIBLE	-9945	Library document cannot be used because the specified usage is invalid.
RFAXERR_FAXLOCKED	-9946	The fax is currently locked by a user.
RFAXERR_FAXNOTLOCKED	-9947	The fax is not currently locked by a user.
RFAXERR_WORKFLOWMAILBOXALREADYINUSE	-9948	The workflow is trying to use mailbox that is already used by another workflow.
RFAXERR_LIBDOCREQUIREDBYGROUP	-9949	The libdoc is being referenced by a group as a required libdoc.
RFAXERR_SERVERUNAVAILABLE	-9950	The server is currently unavailable.
RFAXERR_ERR_NOTCONFIGUREDFOROAUTH	-9951	
RFAXERR_NOTSUPPORTEDINARCHIVE	-9952	
RFAXERR_LOWMEM	-9998	Insufficient memory.
RFAXERR_UNKNOWN	-9999	Unknown error was encountered.

RFAXFAXFILTERERRORCODEFLAGS_???

Use in RFAXFAXFILTER.errorCodeFlags.

Constant	Value	Description
FAXFILTER_ERROR_PRINT_ERROR	0x00000001	Print operation failed.

RFAXFAXFILTERFLAGS

Constants for the searchFlags field of RFAXFAXFILTER.

General Filters

Constant	Value	Description
FAXFILTER_INCLUDE_DELEGATORS	0x00000001	Include delegators and any users that the user performing the search has admin access to.
FAXFILTER_INCLUDE_SELF	0x00000002	Include documents that belong to the user performing the search.
FAXFILTER_INCLUSIVE_STRINGS	0x00000004	Strings search. When set, any match is accepted for items such as csid, comment and destination. When not set, the entire item must match.

Status Filters

Constant	Value	Description
FAXFILTER_STATUS_NEEDS_FCS	0x00000008	Needs fax cover sheet.
FAXFILTER_STATUS_NEEDS_CONVERSION	0x00000010	Needs conversion.
FAXFILTER_STATUS_IN_CONVERSION	0x00000020	In conversion.
FAXFILTER_STATUS_WAITING_SEND	0x00000040	Waiting to send.
FAXFILTER_STATUS_MANUAL	0x00000080	Manually recorded.
FAXFILTER_STATUS_DUPLICATE	0x00000100	Duplicate status.
FAXFILTER_STATUS_NEEDS_ATTENTION	0x00000200	Needs attention.
FAXFILTER_STATUS_WAIT_ATTACHMENT	0x00000400	Waiting for attachment.
FAXFILTER_STATUS_IN_SEND	0x00000800	Sending.
FAXFILTER_STATUS_SCHEDULED_SEND	0x00001000	Scheduled to send.
FAXFILTER_STATUS_DONE_OK	0x00002000	Done OK.
FAXFILTER_STATUS_DONE_WITH_ERRORS	0x00004000	Done, but with errors.
FAXFILTER_STATUS_ERROR_AFTER_RETRY	0x00008000	Retry errors.
FAXFILTER_STATUS_HELD	0x00010000	Held for preview.
FAXFILTER_STATUS_IN_OCR	0x00020000	Fax is being OCR'd
FAXFILTER_STATUS_IN_PRINT	0x00040000	Fax is being printed.
FAXFILTER_STATUS_QUEUED_OCR	0x00080000	Fax is queued for OCR.

Constant	Value	Description
FAXFILTER_STATUS_QUEUED_PRINT	0x00100000	Fax is queued for printing.
FAXFILTER_STATUS_IN_VALIDATE	0x00200000	Faxes whose status is 'in validation.'
FAXFILTER_STATUS_IN_APPROVAL	0x00400000	Faxes waiting for approval.
FAXFILTER_STATUS_ALL	0x007FFFF8	Any status.

Delivery Filters

Constant	Value	Description
FAXFILTER_DELIVERY_FAX	0x00800000	Include faxes.
FAXFILTER_DELIVERY_EMAIL	0x01000000	Include emails.
FAXFILTER_DELIVERY_SMS	0x02000000	Include SMS messages.
FAXFILTER_DELIVERY_CERTIFIED	0x04000000	Include Certified Delivery documents.
FAXFILTER_DELIVERY_ALL	0x07800000	Include all.

Usergroup Filters

Constant	Value	Description
FAXFILTER_USERGROUP_HANDLES	0x80000000	Specify this bit to use RFAXFAXFILTER::userHandleList and RFAXFAXFILTER::groupHandleList. Clear this bit to use RFAXFAXFILTER::userIdList and RFAXFAXFILTER::groupHandleList.

Behavioral Filters

Constant	Value	Description
FAXFILTER_INCLUSIVE_STRINGS	0x00000004	Inclusive search. When set, any match is accepted for items such as CSID, comment and destination. When not set, the entire item must match.

RFAXSTS_???

Error values returned by function RFaxCreateStatusHandle() and used in the ulError member of STATUSVALUEENTRY.

Constant	Value	Description
RFAXSTS_SUCCESS	0	Success execution/return.
RFAXSTS_NOTFOUND	2	Value or service not found.
RFAXSTS_PATHNOTFOUND	3	Specified path not found.
RFAXSTS_INVALID_HANDLE	6	Invalid RFAXSTATUSHANDLE passed to function.
RFAXSTS_NOT_ENOUGH_MEMORY	8	Not enough memory or other system resources available to complete requested operation.
RFAXSTS_INVALID_ARGUMENT	87	Argument passed to a function is invalid. One of the values in ulDataID, ulIndex1, or ulIndex2 of a STATUSVALUEENTRY is invalid.
RFAXSTS_SERVICE_NOT_AVAILABLE	10000	The service represented by the ulDataID of a STATUSVALUEENTRY structure is not available. The service is not running for some reason.
RFAXSTS_INVALID_VERSION	10001	A version mismatch has occurred between the RFSTAT.DLL, theRightFax RPC Server Module, and/or on of the other RightFax modules.
RFAXSTS_UNKNOWN	-9999	Unknown error occurred.

RFAXTHREADING_??? (ApiThreadResolution)

The API can now detect concurrent usage of a single connection (RFAXSERVERHANDLE) by multiple threads. This function can be used to specify the method to handle these situations.

Constant	Value	Description
RFAXTHREADING_IGNORE		Ignore concurrent use of the same connection. This can cause problems. Default method for backward compatibility.
RFAXTHREADING_BLOCK		Block when another thread is using the connection.
RFAXTHREADING_FAIL		Fail when another thread is using the connection. Call will return ERROR_BUSY.
RFAXTHREADING_FAULT		Generate a fault when another thread is using the connection.

RFSERVICETYPE_???

Type of RightFax service. See *SERVICEINFO_10* on page 369.

Constant	Value	Description
UNKNOWN		
EMAILGATEWAY		
DOCTRANSPORT	4	
WORKSERVER	7	

RFSTATPROTOCOL_???

Constants for the usCommProtocol argument of function RFaxCreateStatusHandle().

Constant	Value	Description
RFSTATPROTOCOL_SPX	1	RPC over SPX.
RFSTATPROTOCOL_TCPIP	2	RPC over TCP/IP. This is the recommended protocol.

SAVEFAX_???

Constants for the usSaveOpt1 argument when saving a fax.

Constant	Value	Description
SAVEFAX_NOKICK	0	Save the fax without motivating it.
SAVEFAX_KICK	1	Update the fax information and motivate the fax dependent upon whether an attempt has already been made to send this fax. See RFaxKickFax().
SAVEFAX_NEWROUTE	2	Update the fax information and tell the server the fax will be following new routing information to a user.

SAVEFAX2_???

Constants for the usSaveOpt2 argument when saving a fax.

Constant	Value	Description
SSAVEFAX2_SAVEFAX	0	Save the whole FAXINFO_10 structure with the exception of print and OCR status fields.
AVEFAX2_ONLY_FAXREC	1<<0	Only save the FaxRec portion of the FAXINFO_10 structure.

Constant	Value	Description
SSAVEFAX2_PRESERVELC	1<<1	This flag should not be used. Preserve link count. Causes the link count field of the FAXINFO_10 structure to be ignored.
SAVEFAX2_ONLY_FAXSTATUS	1<<2	Only save the fax_status and fax_error_code fields.
SAVEFAX2_ONLY_BODYOCRSTATUS	1<<3	Save the body_ocr_status and body_error_code fields. Only use this when processing body OCR results.
SAVEFAX2_ONLY_BODYOCRDATA	1<<4	Save the fields modified by body OCR: ffr_flags, ulBFTBytes, szBFTFile fields. Only use this when processing body OCR results.
SAVEFAX2_ONLY_FCSOCRSTATUS	1<<5	Save the fcs_ocr_status and fcs_error_code fields. Only use this when processing FCS OCR results.
SAVEFAX2_ONLY_FCSOCRDATA	1<<6	Save the fr_flags2 field. Only use this when processing FCS OCR results.
SAVEFAX2_ONLY_PRINTSTATUS	1<<7	Save the print_status and print_error_code fields. Only use this when processing print results.

SAVEFAX3_???

Constants for the usSaveOpt3 argument when saving a fax.

Constant	Value	Description
SAVEFAX_NOCHECKCOMPLETE	1	Don't set or clear the FCS complete flag.

SAVEFAX5_???

Constants for the usSaveOpt5 argument when saving a fax.

Constant	Value	Description
SAVEFAX5_FORWARDED	1	Fax being saved was forwarded from another fax (so privacy history item can be generated).

SAVEFAXNUMBEREXCLUSIONS_???

Constants for the usOptions parameter of function RFaxSaveFaxNumberExclusions().

Constant	Value	Description
SAVEFAXNUMBEREXCLUSIONS_CLEAR	1	Indicates to clear all existing exclusions before saving the new ones.

SEARCHPHONEOPTS_???

Constants for the usSearchOptions parameter of *RFaxGetPhoneList4()* on page 249.

Constant	Value	Description
SEARCHPHONEOPTS_ID	0x0001	Search for the ID of the entry.
SEARCHPHONEOPTS_NAME	0x0002	Search the name field.

SEARCHUSEROPTS_???

Constants for the usSearchOptions parameters of function RFaxGetUserList5().

Constant	Value	Description
SEARCHUSEROPTS_NONE	0x0000	
SEARCHUSEROPTS_USERNAME	0x0001	Search in the user name field.
SEARCHUSEROPTS_USERID	0x0002	Search in the user ID field.

SECURE_DOCS_???

Constants for the flagsSecureDocs argument of the RFaxSendSecureDocSingleDest() function.

Constant	Value	Description
SECURE_DOCS_PDF	0x1	
SECURE_DOCS_CERTIFIED	0x2	

SD_ATTACHFLAG_NATIVE_???

Constants for the ulFlags field of SD_ATTACHITEM.

Constant	Value	Description
SD_ATTACHFLAG_NATIVE_INLINE	0x00000001	Inline Disposition.
SD_ATTACHFLAG_NATIVE_BODY	0x00000002	Native doc for use in body of the message
SD_ATTACHFLAG_NATIVE_DOC	0x00000004	Simple native document.
SD_ATTACHFLAG_NATIVE_MASK	0x00000007	All Native doc specific flags.

SERVERFEATURE_???

Constants for the ulServerFeatures field of VERSIONINFO_0.

Constant	Value	Description
SERVERFEATURE_SSCLIENTFAILOVER_ENABLED	0x00000001	

SERVERINFO2_FLAGS_???

Constants for the ulServerFlags field of SERVERINFO2.

Constant	Value	Description
SERVERINFO2_FLAGS_ANSIMODE	0x00000001	
SERVERINFO2_FLAGS_BILLDESCFIXED	0x00000002	
SERVERINFO2_FLAGS_ENCRYPTIMAGES	0x00000010	The encryption module is enabled. Images are stored encrypted.
SERVERINFO2_FLAGS_SQLIMAGES	0x00000020	Storage of images in SQL is enabled.
SERVERINFO2_FLAGS_ARCHIVEFAXES	0x00000040	RightFax Archiving is enabled. Aged faxes will be stored in RightFax Archive.
SERVERINFO2_FLAGS_ISPATCH	0x80000000	A beta SR has been installed on this server.

SERVERSPECIAL_???

Constants for the ulServerSpecial fields of SERVERINFO and SERVERINFO2.

Constant	Value	Description
SERVERSPECIAL_DEALER	9999	The server is a demonstration copy.
SERVERSPECIAL_TAMPERED	9998	The server authorization has been tampered with.

SERVERTYPE_???

Constants for the ulServerType fields of SERVERINFO and SERVERINFO2.

Constant	Value	Description
SERVERTYPE_OS2	0x0CF1	Server is an OS2 server.

Constant	Value	Description
SERVERTYPE_NT	0x0CF2	Server is a NT server.
SERVERTYPE_DTA	0x0CF3	Server is a Remote DocTransport server.

STATUS_TYPE_???

Constants for the pusStatusType argument of function RFaxStringFaxStatus() and the usStatusType filed of FAXLISTELEMENT0, FAXLISTELEMENT1, and FAXLISTELEMENT2.

Constant	Value	Description
STATUS_TYPE_FAIL	0	The fax has failed to send and will not be tried again.
STATUS_TYPE_RETRY	1	The fax has recently failed, and will be retried.
STATUS_TYPE_OK	2	The fax sent successfully.
STATUS_TYPE_INPROGRESS	3	The fax is currently sending.

SVC_SERVICETYPE_???

Constants for the sServiceType field of SVCINFO_10.

Constant	Value	Description
SVC_SERVICETYPE_SMTP	0	SMTP.
SVC_SERVICETYPE_TAP	1	TAPI.
SVC_SERVICETYPE_UCP	2	UCP.
SVC_SERVICETYPE_SMS	3	SMS.

SVCFLAG_???

Constants for the ulFlags field of SVCINFO_10.

Constant	Value	Description
SVCFLAG_TAPIDIRECTADDRESS	0x00000001	Use TAPI direct addressing.
SVCFLAG_DIALPULSE	0x00000002	Use dial pulse.

TERMSTAT_???

Constants for the fax_termstat fields of FAXINFO_10 and FAXINFO_11, and the d.trx.termstat field of HISTLISTELEMENT0.

Constant	Value	Description
TERMSTAT_SUCCESSRECV	0x0	Success for a received fax.
TERMSTAT_BUSY1	0x1	Busy tone detected.
TERMSTAT_BUSY2	0x2	Busy tone detected.
TERMSTAT_ROBUSY	0x3	Reorder busy tone.
TERMSTAT_RECALL	0x4	Same as the Brooktrout code FCP_RECALL.
TERMSTAT_CONFIRM	0x5	Same as the Brooktrout code FCP_CONFIRM.
TERMSTAT_BLOCKED	0x6	Fax was blocked due to a dialing rule.
TERMSTAT_RING1	0x8	No pick up.
TERMSTAT_RING2	0x9	No pick up.
TERMSTAT_HUMAN	0x10	Human answered.
TERMSTAT_ANSWER	0x11	Answered but no fax tone detected.
TERMSTAT_DIALTON	0x12	Dial tone detected.
TERMSTAT_SITDETECT	0x13	Same as the Brooktrout code FCP_SITINTC.
TERMSTAT_SITNOCIR	0x14	Same as the Brooktrout code FCP_SITNOCIR.
TERMSTAT_TXRXERROR	0x15	Transmission or receive error.
TERMSTAT_HNGNOLOOPCUR	0x16	No loop current.
TERMSTAT_CALLCOLLIDE	0x17	Call collided.
TERMSTAT_DIALNOLOOPCUR	0x18	No loop current on dial.
TERMSTAT_RNGNOANS	0x19	Line rang but was not answered.
TERMSTAT_G2DET	0x1a	Group 2 fax machine detected.
TERMSTAT_QUIET	0x1c	Line is quiet.
TERMSTAT_FAXMACHINE	0x20	Success for a sent fax.
TERMSTAT_LOCALINUSE	0x21	Line in use.

Constant	Value	Description
TERMSTAT_SILENCE	0x22	Line is quiet.
TERMSTAT_UNKNOWN	0x23	Unknown error code.

TRXREC_BOARDTYPE_???

Constants for the d.trx.usBoardType field of HISTLISTELEMENT0.

Constant	Value	Description
TRXREC_BOARDTYPE_BROOK	1	For histories generated by RightFax version 5.00 and older servers with Brooktrout boards.
TRXREC_BOARDTYPE_GAMMA	2	For histories generated by any server with GammaLink boards.
TRXREC_BOARDTYPE_BROOKNEW	3	For histories generated by RightFax version 5.20 and later servers with Brooktrout boards.
TRXREC_BOARDTYPE_SMS	4	For histories generated by server using SMS.
TRXREC_BOARDTYPE_FOIP	5	For histories generated by server using FOIP.
TRXREC_BOARDTYPE_EICON	6	For histories generated by any server with Eicon boards.
TRXREC_BOARDTYPE_INTEL	7	For histories generated by any server with Intel boards.
TRXREC_BOARDTYPE_P2P	8	For histories generated by server using P2P boards.

TRXREC_FLAGS_???

Constants for the .d.trx.usBoardType field of HISTLISTELEMENT0.

Constant	Value	Description
TRXREC_FLAGS_PARTIALRETRY	0x0001	Fax was a partial retry.
TRXREC_FLAGS_SENT_REMOTE	0x0002	Fax was a remote send.
TRXREC_FLAGS_ANI_VALID	0x0004	ANI was valid.
TRXREC_FLAGS_AOC_VALID	0x0008	AOC was valid.
TRXREC_FLAGS_ISDN_VALID	0x0010	ISDN cause value is valid
TRXREC_FLAGS_NO_RETRY	0x0020	Fax will not be retried.
TRXREC_FLAGS_CANCELED_HAF	0x0040	Canceled due to HAF (human answered fax).

Constant	Value	Description
TRXREC_FLAGS_BLOCKED_NUMBER	0x0080	Allow the transport to decide if a fax should be blocked.
TRXREC_FLAGS_RFC_NO_NOTIFY	0x0100	
TRXREC_FLAGS_HASECM	0x0200	Fax was sent with error correction mode (ECM) enabled for the call.

UDFLAG_???

Constants for the ulDisplayFlags field of USERLISTELEMENT2.

Constant	Value	Description
UDFLAG_BADSID	0x00000001	The user's SID field is not a valid SID.

USER_EDCFLAGS_???

Constants for USERINFO_11.usEDCArchiveFlags.

Constant	Value	Description
USER_EDCFLAGS_XMLGEN	0x0001	XML Generator should process.
USER_EDCFLAGS_VAULT	0x0002	Vault should process.
USER_EDCFLAGS_FNET	0x0004	Filenet should process.

USER_PAGERNOTIFY_???

Constants for the following fields: EXTENSIONINFO_10.d.UserPagerNotify.ulTypes1 and EXTENSIONINFO_10.d.UserPagerNotify.ulTypes2.

Constant	Value	Description
USER_PAGERNOTIFY_NEWFAXRECEIVED	1	Notify the user via pager, when a new fax is received.
USER_PAGERNOTIFY_NEWFAXFROMCSID	2	Notify the user via pager, when a new fax is received whose CSID matches an entry in the szCSIDList field.
USER_PAGERNOTIFY_OUTBOUND_ABANDONED	3	Notify the user via pager, when an outbound fax fails finally.

Masks

MAKE_PAGERNOTIFY_MASK(connType) (1 << (connType - 1))

USER_ROUTEFMT_???

Constants for the routefmt fields of USERINFO_10, USERINFO_11, USERLISTELEMENT1, and USERLISTELEMENT2, and the ucWebImageFormat fields of USERINFO_10 and USERINFO_11.

Constant	Value	Description
USER_ROUTEFMT_DCX	0	DCX.
USER_ROUTEFMT_PCX	1	PCX.
USER_ROUTEFMT_TIFF1	2	TIFF-G3.
USER_ROUTEFMT_TIFF2	3	TIFF-G4.
USER_ROUTEFMT_GIF	4	GIF.
USER_ROUTEFMT_PDF	5	PDF.
USER_ROUTEFMT_PDFWTMB	6	PDF with thumbnails.
USER_ROUTEFMT_CPC	7	CPC.
USER_ROUTEFMT_RFX	8	RFX.
USER_ROUTEFMT_PNG	9	PNG.
USER_ROUTEFMT_TXT	10	txt.
USER_ROUTEFMT_PDFSRCH	11	Searchable PDF.
USER_ROUTEFMT_ENHANCED	12	
USER_ROUTEFMT_WEBDEV	13	WebDelivery option.
USER_ROUTEFMT_JPG	14	
USER_ROUTEFMT_JSON	15	For Workflow, this is automated field capture output.

USER_ROUTETYPE_???

Constants for the routetype fields of USERINFO_10USERINFO_11, USERLISTELEMENT1, and USERLISTELEMENT2.

Constant	Value	Description
USER_ROUTETYPE_RF	0	Route to RightFax mailbox.
USER_ROUTETYPE_CCMAIL	1	Route to cc:Mail.

Constant	Value	Description
USER_ROUTETYPE_MSMAIL	2	Route to MS-Mail.
USER_ROUTETYPE_MHS	3	Not used.
USER_ROUTETYPE_FILE	4	Route to file.
USER_ROUTETYPE_OCR	5	Do OCR routing.
USER_ROUTETYPE_WPOFF	6	Route to Groupwise.
USER_ROUTETYPE_WPOMAIL	6	Same as USER_ROUTETYPE_WPOFF.
USER_ROUTETYPE_NOTES	7	Route to Lotus Notes.
USER_ROUTETYPE_XROUTE	8	Route the fax to a different RightFax server.
USER_ROUTETYPE_TRS	9	Route to TRS.
USER_ROUTETYPE_CX3	10	Route to CallXPressFax.
USER_ROUTETYPE_EXCHANGE	11	Route to Exchange.
USER_ROUTETYPE_SMTP	12	Route to SMTP.
USER_ROUTETYPE_ANI	13	Route to ANI.
USER_ROUTETYPE_SAP1	14	Route to an SAP gateway.
USER_ROUTETYPE_SAP2	15	Route to an SAP gateway.
USER_ROUTETYPE_SAP3	16	Route to an SAP gateway.
USER_ROUTETYPE_SAP4	17	Route to an SAP gateway.
USER_ROUTETYPE_SAP5	18	Route to an SAP gateway.
USER_ROUTETYPE_SAP6	19	Route to an SAP gateway.
USER_ROUTETYPE_SAP7	20	Route to an SAP gateway.
USER_ROUTETYPE_SAP8	21	Route to an SAP gateway.
USER_ROUTETYPE_SAP9	22	Route to an SAP gateway.
USER_ROUTETYPE_SAP10	23	Route to an SAP gateway.
USER_ROUTETYPE_SAP11	24	Route to an SAP gateway.
USER_ROUTETYPE_SAP12	25	Route to an SAP gateway.

Constant	Value	Description
USER_ROUTETYPE_SAP13	26	Route to an SAP gateway.
USER_ROUTETYPE_SAP14	27	Route to an SAP gateway.
USER_ROUTETYPE_SAP15	28	Route to an SAP gateway.
USER_ROUTETYPE_SAP16	29	Route to an SAP gateway.
USER_ROUTETYPE_SAP17	30	Route to an SAP gateway.
USER_ROUTETYPE_SAP18	31	Route to an SAP gateway.
USER_ROUTETYPE_SAP19	32	Route to an SAP gateway.
USER_ROUTETYPE_SAP20	33	Route to an SAP gateway.
USER_ROUTETYPE_SAP21	34	Route to an SAP gateway.
USER_ROUTETYPE_SAP22	35	Route to an SAP gateway.
USER_ROUTETYPE_SAP23	36	Route to an SAP gateway.
USER_ROUTETYPE_SAP24	37	Route to an SAP gateway.
USER_ROUTETYPE_SAP25	38	Route to an SAP gateway.
USER_ROUTETYPE_SAP26	39	Route to an SAP gateway.
USER_ROUTETYPE_SAP27	40	Route to an SAP gateway.
USER_ROUTETYPE_SAP28	41	Route to an SAP gateway.
USER_ROUTETYPE_SAP29	42	Route to an SAP gateway.
USER_ROUTETYPE_SAP30	43	Route to an SAP gateway.
USER_ROUTETYPE_SAP31	44	Route to an SAP gateway.
USER_ROUTETYPE_SAP32	45	Route to an SAP gateway.
USER_ROUTETYPE_SAP33	46	Route to an SAP gateway.
USER_ROUTETYPE_SAP34	47	Route to an SAP gateway.
USER_ROUTETYPE_SAP35	48	Route to an SAP gateway.
USER_ROUTETYPE_SAP36	49	Route to an SAP gateway.
USER_ROUTETYPE_SAP37	50	Route to an SAP gateway.

Constant	Value	Description
USER_ROUTETYPE_SAP38	51	Route to an SAP gateway.
USER_ROUTETYPE_SAP39	52	Route to an SAP gateway.
USER_ROUTETYPE_SAP40	53	Route to an SAP gateway.
USER_ROUTETYPE_SAP41	54	Route to an SAP gateway.
USER_ROUTETYPE_SAP42	55	Route to an SAP gateway.
USER_ROUTETYPE_SAP43	56	Route to an SAP gateway.
USER_ROUTETYPE_SAP44	57	Route to an SAP gateway.
USER_ROUTETYPE_SAP45	58	Route to an SAP gateway.
USER_ROUTETYPE_SAP46	59	Route to an SAP gateway.
USER_ROUTETYPE_SAP47	60	Route to an SAP gateway.
USER_ROUTETYPE_SAP48	61	Route to an SAP gateway.
USER_ROUTETYPE_SAP49	62	Route to an SAP gateway.
USER_ROUTETYPE_SAP50	63	Route to an SAP gateway.
USER_ROUTETYPE_SAP51	64	Route to an SAP gateway.
USER_ROUTETYPE_SAP52	65	Route to an SAP gateway.
USER_ROUTETYPE_SAP53	66	Route to an SAP gateway.
USER_ROUTETYPE_SAP54	67	Route to an SAP gateway.
USER_ROUTETYPE_SAP55	68	Route to an SAP gateway.
USER_ROUTETYPE_SAP56	69	Route to an SAP gateway.
USER_ROUTETYPE_SAP57	70	Route to an SAP gateway.
USER_ROUTETYPE_SAP58	71	Route to an SAP gateway.
USER_ROUTETYPE_SAP59	72	Route to an SAP gateway.
USER_ROUTETYPE_SAP60	73	Route to an SAP gateway.
USER_ROUTETYPE_WORKFLOW	74	Route to a workflow.
USER_ROUTETYPE_FTP	75	Route to SFTP/FTP.

Constant	Value	Description
USER_ROUTETYPE_CXINTF	200	Reserved.

USERAPFLAGS_???

Constants for the usAutoPrintFlags fields of USERINFO_10and USERINFO_11.

Constant	Value	Description
USERAPFLAGS_PRINTSENT	0x0001	Auto-print of sent faxes enabled to printer ID SVCINFO_10.szAutoSentPrinter.
USERAPFLAGS_SENTFCS	0x0002	Print cover sheet of sent faxes.
USERAPFLAGS_SENTBODY	0x0004	Print body of sent faxes.
USERAPFLAGS_SENTTRX	0x0008	Print transmission history of sent faxes.
USERAPFLAGS_RECVFCS	0x0010	Print cover sheet (first page) of received faxes.
USERAPFLAGS_RECVBODY	0x0020	Print body (pages 2 and on) of received faxes.
USERAPFLAGS_RECVTRX	0x0040	Print transmission history of received faxes.
USERAPFLAGS_SENTFAILED	0x0080	Print when send fails and goes to ED:xxx state.
USERAPFLAGS_SENTSUCCESS	0x0100	Print when send succeeds.
USERAPFLAGS_SENTBODY_FIRSTPAGEONLY	0x0200	Print first page of body of sent faxes only.
USERAPFLAGS_SENTTHUMBNAIL	0x0400	Print with thumbnail of first page.
USERAPFLAGS_SKIPCOVER	0x0800	If USERAPFLAGS_SENTTHUMBNAIL is set, use first page of fax body.

USERFLAGS_???

Constants for the userflags fields of USERINFO_10, USERINFO_11, USERLISTELEMENT1, and USERLISTELEMENT2.

Constant	Value	Description
USERFLAGS_ADMIN	0x00000001	User had administrative rights.
USERFLAGS_CANNEWLIB	0x00000002	Can use the NEWLIB embedded code.
USERFLAGS_CANNEWFORM	0x00000004	Can create new forms.
USERFLAGS_CANUSEHIGH	0x00000010	Can use high priority.

Constant	Value	Description
USERFLAGS_MARKED	0x00000020	Reserved. Initialize to zero.
USERFLAGS_UNPROTECTED	0x00000040	Indicates that mailbox is unprotected.
USERFLAGS_DONTVERIFY	0x00000080	Indicates that user is exempt from code verification.
USERFLAGS_DEFAULTFINE	0x00000100	Set to indicate that the default mode for this user is fine mode.
USERFLAGS_VIEWFIRSTONLY	0x00000200	Set to restrict viewing to only the first page of received faxes.
USERFLAGS_AUTOPRINT	0x00000400	Set to auto-print received faxes (autoprint_netprint_id contains printer ID to which to print).
USERFLAGS_COVERPAGE	0x00000800	Default setting of cover sheet generation.
USERFLAGS_AUTOFORWARD	0x00001000	Set to auto-forward received faxes.
USERFLAGS_AUTOUPDATE	0x00002000	Automatically refresh fax list.
USERFLAGS_ALTNOTIFY	0x00004000	Alternate notification of received faxes enabled?
USERFLAGS_FINECOVER	0x00008000	Set to indicate that cover sheets are created in fine mode.
USERFLAGS_AUTOPURGE	0x00010000	Set to indicate that this user will not have a fax_deleted_set.
USERFLAGS_DISABLED	0x00020000	Set to indicate that this is a disabled account.
USERFLAGS_MUSTHAVEPASS	0x00040000	Set to indicate that user must have non-blank password.
USERFLAGS_ARCHIVEON	0x00080000	Set to turn on archive.
USERFLAGS_CANCHGCOVERS	0x00100000	Set to indicate that user can change the cover sheet.
USERFLAGS_AUTOOCR	0x00200000	Set to automatically OCR all received faxes.
USERFLAGS_CANOCR	0x00400000	Can use OCR functions.
USERFLAGS_NODELETE	0x00800000	If set, then deleting from FaxUtil doesn't work.
USERFLAGS_SMARTRESUME	0x01000000	If set, then the default for faxes is to enable Smart-Resume.
USERFLAGS_NOBILLSEARCH	0x02000000	If set, then user cannot look up or view a list of billing codes.
USERFLAGS_AUTOPRINT_SUCCESSONLY	0x04000000	If set, only received faxes that do not have error codes auto-print for the user.
USERFLAGS_ADMIN_SUB_READONLY	0x08000000	Admin sub permission Read-Only Admin.
USERFLAGS_ADMIN_SUB_USERMGMT	0x10000000	Admin sub permission User Management Admin.
USERFLAGS_CANVIEWANALYTICS	0x20000000	If set, the user can view Analytics.

Constant	Value	Description
USERFLAGS_SYNCCREATED	0x40000000	If set, then this user was created by some synchronization system.
USERFLAGS_AUTOSENTOCR	0x80000000	Set to automatically OCR sent faxes.

Masks for USERFLAGS_

USERFLAGS_MASK_ANYADMIN (USERFLAGS_ADMIN|USERFLAGS_ADMIN_SUB_READONLY|USERFLAGS_ADMIN_SUB_USERMGMT)

USERFLAGS2_???

Constants for the ulUserFlags2 fields in USERINFO_10, USERINFO_11, and USERLISTELEMENT2.

Constant	Value	Description
USERFLAGS2_USETRASH	0x00000001	User is set to use the Trash folder by default (instead of deleting).
USERFLAGS2_EMPTYTRASH	0x00000002	User is set to empty Trash folder on exit of client program.
USERFLAGS2_WEBSENDTIFF	0x00000004	User will receive TIFFs instead of GIFs when using FaxUtil Web.
USERFLAGS2_WEBSENDRFX	0x00000008	User will receive RFXs instead of GIFs when using FaxUtil Web.
USERFLAGS2_RESERVED1	0x00000010	Reserved for internal use only.
USERFLAGS2_SFDENABLED	0x00000020	Used for SFD routing where users can log out of routing groups.
USERFLAGS2_NEEDSAPPROVAL	0x00000040	User's group monitor or alternate monitors must approve faxes.
USERFLAGS2_EXCLUDEFROMAGE	0x00000080	User's faxes should not age with group.
USERFLAGS2_CANRUNREPORTS	0x00000100	User is able to use the Fax Reporter.
USERFLAGS2_NOANNOTATION	0x00000200	User is not able to annotate faxes.
USERFLAGS2_NOBILLEDIT	0x00000400	User cannot change the billing codes for a fax.
USERFLAGS2_USEDEFBCONRCV	0x00000800	Assign default billing codes to received faxes.
USERFLAGS2_REQNTAUTHEN	0x00001000	User requires NT authentication.
USERFLAGS2_STAMPRECVPAGES	0x00002000	Stamp received pages with time-stamp.
USERFLAGS2_EDCRECEIVES	0x00004000	External-archive all received faxes.
USERFLAGS2_EDCSENDS	0x00008000	External-archive all sent faxes.
USERFLAGS2_SMSENABLED	0x00010000	User is allowed to send SMS messages.

Constant	Value	Description
USERFLAGS2_BYPASSPRIVACY	0x00020000	User can bypass various privacy restrictions.
USERFLAGS2_CANNOTCHANGENOTIFY	0x00040000	If NOT set, then user can change their own notification settings.
USERFLAGS2_NOMODIFYDELEGATES	0x00080000	User cannot modify delegate settings for their or other users' mailboxes.
USERFLAGS2_NOPHONEBOOK	0x00100000	User cannot edit or create phonebooks.
USERFLAGS2_EDCFWD_RTE_DEL	0x00200000	External-archive forwarded, routed, and deleted faxes.
USERFLAGS2_EDCFORWARDS	0x00400000	External-archive forwarded faxes.
USERFLAGS2_EDCROUTES	0x00800000	External-archive routed faxes.
USERFLAGS2_EDCDELETES	0x01000000	External-archive deleted faxes.
USERFLAGS2_NOSERVERPRINT	0x02000000	User cannot print to network printers configured on the fax server.
USERFLAGS2_NOGROUPSDELEGATE	0x04000000	User cannot delegate to groups.
USERFLAGS2_CANVIEWPRIORVERSIONS	0x08000000	User can view prior fax versions.
USERFLAGS2_CANMANAGESERVICESVIAWEB	0x10000000	Administrative user can configure services via Web Admin.
USERFLAGS2_DISALLOWSENDEMAIL	0x20000000	User cannot send outbound email.
USERFLAGS2_ONLYPHONEBOOKSEND	0x40000000	User may only send via phonebook contact.
USERFLAGS2_EXCLUDEFROMARCHIVE	0x80000000	Exclude this user from archiving.

USERFLAGS3_???

Constants for the ulUserFlags3 fields in USERINFO_10, USERINFO_11, and USERLISTELEMENT2.

Constant	Value	Description
USERFLAGS3_ALTNOTIFYSENT	0x00000001	
USERFLAGS3_REQUIREOAUTH	0x00000002L	Require OAuth.
USERFLAGS3_CANEDITWORKFLOWS	0x00000004L	Can create or update workflows.
USERFLAGS3_HASPFPFOLDERS	0x00000008L	A hint for the faxserver when processing PFP jobs.

USERMSG_OPTION__???

Constants for RFaxGetUserMessage().usOption to specify what type of message is being requested.

Constant	Value	Description
USERMSG_OPTION_NEWFAXNOTIFY	0x0001	Request a formatted message for the next new fax that's just arrived.

USERORDER_???

Filters for the usOrder parameters of function *RFaxGetPagedUserList()* on page 150, *RFaxGetUserList2()* on page 151, *RFaxGetUserList3()* on page 152, *RFaxGetUserList4()* on page 152, and *RFaxGetUserList5()* on page 153

Constant	Value	Description
USERORDER_DEFAULT	0	
USERORDER_USERID	1	
USERORDER_USERNAME	2	
USERORDER_ROUTECODE	3	
USERORDER_GROUP	4	
USERORDER_NUMFAXES	5	
USERORDER_SUBID	6	
USERORDER_ROUTETYPE	7	
USERORDER_ROUTEFMT	8	
USERORDER_NTAUTH	9	
USERORDER_DEFPRI	10	
USERORDER_UNPROT	11	
USERORDER_AUTOFWDRECV	12	
USERORDER_USEHIGH	13	

VERIFYUSER_???

Constants for loginType parameter of RFaxVerifyUser().

Constant	Value	Description
VERIFYUSER_NONE	0x0000	No admin permissions required.

Constant	Value	Description
VERIFYUSER_ADMIN	0x0001	User must have admin permission.
VERIFYUSER_ADMIN_READONLY	0x0002	User must have 'read-only' admin permission.
VERIFYUSER_ADMIN_USERMANAGEMENT	0x0004	User must have 'user management' admin permission.
VERIFYUSER_ADMIN_GROUP	0x0008	User must be an administrator of at least one group.

Masks

VERIFYUSER_ADMIN_ANY (VERIFYUSER_ADMIN | VERIFYUSER_ADMIN_READONLY | VERIFYUSER_ADMIN_USERMANAGEMENT)

VERSIONINFOFLAG_???

Constants for the ulFlags field of *VERSIONINFO_0* on page 387

Constant	Value	Description
VERSIONINFOFLAG_DATABASEDOWN	0x00000001	Persistent issues accessing the database. API operations are stalled.
VERSIONINFOFLAG_FILESDATABASEDOWN	0x00000002	Persistent issues accessing the files database. API file operations are stalled,
VERSIONINFOFLAG_HIDEBYPASSRESTRICT	0x00000004	When set, the "Can Bypass Privacy Restrictions" administrative option will be removed.
VERSIONINFOFLAG_LOCALSSO	0x00000008	Support local SSO.
VERSIONINFOFLAG_HIGHRESOLUTIONFAX	0x00000010	Support only high resolution faxing.
VERSIONINFOFLAG_JITCSERVER	0x00000020	Tells the API to function as a JITC server.

WORKFLOWACTION_???

Constants for the completionAction of FAXLISTELEMENT0 and WORKFLOWLISTELEMENT#.

Constant	Value	Description
None		Fax will not change workflow nor owning user.
Workflow		Fax will move to the configured single workflow.
WorkflowChoice		User will be prompted to choose a workflow from those related to the workflow with the corresponding relation type.
User		Fax will be routed to the configured single user.
PreviousWorkflow		Fax will move back to the prior workflow.

WORKFLOWDELEGATIONFLAGS_???

Constants for the usFlags field of *WORKFLOWDELEGATIONINFO_10* on page 389 and *WORKFLOWDELEGATIONLISTELEMENT0* on page 389.

Constant	Value	Description
WORKFLOWDELEGATIONFLAGS_USER	0x0001	Delegated party is a user.
WORKFLOWDELEGATIONFLAGS_GROUP	0x0002	Delegated party is a group.

WORKFLOWDELEGATIONINFO_???

Constants for the usFlags field of *WORKFLOWDELEGATIONINFO_10* on page 389 and *WORKFLOWDELEGATIONLISTELEMENT0* on page 389.

Constant	Value	Description
WORKFLOWDELEGATIONFLAGS_USER	0x0001	Delegated party is a user.
WORKFLOWDELEGATIONFLAGS_GROUP	0x0002	Delegated party is a group.

WORKFLOWFIELDFLAG_???

Constants for the ulFlags field of *WORKFLOWFIELDINFO_10* on page 390.

Constant	Value	Description
WORKFLOWFIELDFLAG_REQUIRED	0x00000001	
WORKFLOWFIELDFLAG_CHOICES	0x00000002	Admin can specify predefined choices for workflow fields.

WORKFLOWFLAGS_???

Constants for the flags fields of WORKFLOWINFO_10 and *WORKFLOWLISTELEMENT0* on page 391.

Constant	Value	Description
None	0x00000000	

WORKFLOWRELATIONINFO_10

The WORKFLOWRELATIONINFO_10 structure.

C Data Type	Member	Description
FAXHANDLE	hWorkflowOwner	DHandle of the workflow.
FAXHANDLE	hWorkflowTarget	Handle of the workflow being referred to.
USHORT	usType	Type of relationship. See *WORKFLOWRELATIONTYPE_???* below.

WORKFLOWRELATIONTYPE_???

Types used in WORKFLOWRELATIONINFO_##.usType and WORKFLOWRELATIONLISTEMENT#.usType.

Constant	Value	Description
WORKFLOWRELATIONTYPE_COMPLETION	1	The owning workflow can submit to the target workflow on completion.
WORKFLOWRELATIONTYPE_EXCEPTION	2	The owning workflow can submit to the target workflow on exception.
WORKFLOWRELATIONTYPE_REJECTION	3	The owning workflow can submit to the target workflow on rejection.

WORKFLOWVALUEFLAGS_???

Constants for the flags fields of WORKFLOWVALUEINFO_10

Constant	Value	Description
WORKFLOWVALUEFLAGS_NONE	0x00000000	None.
WORKFLOWVALUEFLAGS_CAPTURED	0x00000001	Value was captured by Intelligent Workflows.

WORKSERVICE_???

Constants for the SERVERSTAT_10.Workserver.ulServices field. See REQUEST_ in FD_CONST.h

Constant	Value	Description
WORKSERVICE_PCL	0x00000001	Workserver performs PCL conversion.
WORKSERVICE_DELETE	0x00000002	Workserver performs deletes.
WORKSERVICE_PRINT	0x00000004	Workserver performs prints.
WORKSERVICE_OVERLAY	0x00000008	Workserver performs overlays.
WORKSERVICE_POSTSCRIPT	0x00000010	Workserver performs postscript conversions.

Constant	Value	Description
WORKSERVICE_ARCHIVE	0x00000200	Workserver performs archives.
WORKSERVICE_FILEROUTE	0x00000400	Workserver performs file routes.
WORKSERVICE_COVERSHEET	0x00000800	Workserver performs coversheet conversions.
WORKSERVICE_OCR_SPDF	0x00002000	Workserver performs searchable PDF creation.
WORKSERVICE_OCR1	0x00004000	Workserver performs OCRs.
WORKSERVICE_OCR2	0x00008000	Workserver performs OCRs.
WORKSERVICE_CVL	0x00010000	Workserver performs CVL conversions.
WORKSERVICE_NETMSG	0x00200000	Workserver performs net messaging.
WORKSERVICE_XROUTE	0x00800000	Workserver performs xroutes.
WORKSERVICE_OCR_ENHANCE	0x40000000	Workserver performs image enhancement.

Appendix C: Status metric constants

The following constants are used with the STATUSVALUEENTRY structures and the RFaxStatusGetValues() function.

Fax Database Module Status Metrics

The Database Server Module is a multi-threaded database server that executes all database I/O on behalf of all client software, the RightFax API, and all server modules except the Fax Server Module. Only the Fax Server and Database Modules open and directly manipulate the FAXDATA.* files in the RightFax\Database folder. The Database Server Module uses a pool of threads to service client requests. The maximum number of threads in the pool is configured at the fax server and is fixed at startup. A change to the maximum requires a recycle of the Fax Server and Database Modules. Having too many threads in the pool is not considered a problem because each thread consumes few resources (20 to 30 handles and 64KB to 128KB of RAM).

The Database Module has two types of threads, database and file. Database threads are the most important and the most used. File threads are used by clients requesting file I/O services using the named pipe protocol (file I/O requests through other protocols are handled by the Fax RPC Server Module).

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_DB_INFOVERSION	None	Version of status data presented by module. Returns a value of 0x451.
SVE_DB_SECONDSUP	None	Number of seconds module has been executing. After 2^{31} seconds, the value resets to zero.
SVE_DB_NUMDBTHREADS	None	Number of database threads currently in the thread pool.
SVE_DB_NUMFSTHREADS	None	Number of file I/O threads currently in the thread pool. These threads are only used by clients requesting file I/O operations using the named pipes protocol.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_DB_DBTHREAD_OPERATION	ulIndex1 = Thread (0–*n*) from which to extract metric.	Current stage of command/operation being executed. Possible values are as follows: 0 = Initializing 1 = Creating 2 = Waiting for connect/Idle 3 = Disconnecting from client 4 = Reading data from client 5 = Writing data to client 6 = Shutting down 7 = Stopped 8 = Processing client command 9 = Waiting for database lock 10 = Reading from database 11 = Writing to database
SVE_DB_DBTHREAD_LASTCMD	ulIndex1 = Thread (0–*n*) from which to extract metric.	Last command executed by thread. Possible command values are proprietary and not documented.
SVE_DB_DBTHREAD_READFAILUES	ulIndex1 = Thread (0–*n*) from which to extract metric.	Number of failures reading data from client.
SVE_DB_DBTHREAD_WRITEFAILURES	ulIndex1 = Thread (0–*n*) from which to extract metric.	Number of failures writing data to client.
SVE_DB_DBTHREAD_BYTESREAD	ulIndex1 = Thread (0–*n*) from which to extract metric.	Number of bytes read from clients during life of thread.
SVE_DB_DBTHREAD_BYTESWRITTEN	ulIndex1 = Thread (0–*n*) from which to extract metric.	Number of bytes written to clients during life of thread.
SVE_DB_DBTHREAD_CMDSPROCESSED	ulIndex1 = Thread (0–*n*) from which to extract metric.	Number of commands processed by thread during life.
SVE_DB_DBTHREAD_LASTUSER	ulIndex1 = Thread (0–*n*) from which to extract metric.	Last NT account ID under which command was executed. Only commands originating from a client using named pipes protocol will have an ID other than SYSTEM.

Fax Server Module Status Metrics

The Fax Server Module is the main queuing and master system controller. All faxes are monitored, converted, and updated by this module.

The SVE_FSBRDS_??? metrics, only available on RightFax version 6+ servers, expose BoardServer metrics as seen from the Fax Server Module point of view. Some of the metrics are available directly from the BoardServer(s) and through the Fax Server Module. Each metric, except SVE_FSBRDS_BRDSERVERCOUNT, requires an index value in ulIndex1 indicating the index of the BoardServer being driven by the Fax Server Module from which to extract a query.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_FS_INFOVERSION	None	Version of status data presented by module. Returns a value of 0x452.
SVE_FS_SECONDSUP	None	Number of seconds module has been executing. After 2^{31} seconds, the value resets to zero.
SVE_FS_MAXIMUMEVENTS	None	Maximum number of concurrent events that can be concurrently processed by the Fax Server Module. Each fax in "process" consumes a single event. Additional in-process faxes are left idle on disk until events are freed by faxes completing their processing steps.
SVE_FS_NUMBEREVENTS	None	Current number of events being processed by the Fax Server Module. When this value reaches 90% of the maximum number of events, the server pauses certain operations that cause events to be consumed, such as picking up new faxes from the HPFAX queue. These paused operations are resumed when the percentage reaches 75%. The value "Fax Server Event Queue Usage" shown in Enterprise Fax Manager is calculated as NUMBEREVENTS / MAXIMUMEVENTS.
SVE_FS_EVENTSPROCESSED	None	Running counter of events processed by the Fax Server Module. The faster this value increased, the higher the throughput of the server.
SVE_FS_PERCENTAVAILABLEIMAGESPACE	None	Percentage of free disk space on the volume where fax images are being stored, normally RightFax\Image.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_FS_PERCENTAVAILABLEDBSPACE	None	Percentage of free disk space on the volume where the fax database is being stored, normally RightFax\Database. Note that in normal configurations, the Image and Database directories are on the same volume. In special cases, they can exist on different volumes, thus the two different metrics. If the available disk space falls below certain values, the Fax Server Module pauses certain operations that can, and usually do, consume additional disk space. Operations are resumed when disk space frees up.
SVE_FS_ACTIVITY	ulIndex1 = Activity log record to retrieve (0–*n*)	The Fax Server Module maintains a circular log of descriptions of the last 50 activities executed. The SVE_FS_ACTRECORDS metric indicates the number of elements in the log (up to 50). The SVE_FS_NEWESTACTINDEX metric indicates the newest entry in the log. Each entry in the log is a three-line textual description. Each line is 60 bytes (59 characters with a terminating zero byte). Line 1 starts are offset 0 (szData[0]), line 2 starts are offset 60 (szData[60]), and line 3 at offset 120 (szData[120]).
SVE_FS_NEWESTACTINDEX	None	Indicates the index (0–49) of the activity log entry that is the newest.
SVE_FS_ACTRECORDS	None	Indicates the number of elements in the circular activity log used. After a short while, all entries are used (50) and older entries are overwritten, i.e., the nature of a circular log.
SVE_FSBRDS_BRDSERVERCOUNT	None	Number of BoardServers being driven by the Fax Server Module. Only RightFax Enterprise licensed servers can use more than one BoardServer. All other licenses will always return a value of one (1).
SVE_FSBRDS_MACHINENAME	ulIndex1 = BoardServer index (0–15) to query	Machine name where BoardServer is running.
SVE_FSBRDS_IMAGEDIR	ulIndex1 = BoardServer index (0–15) to query	Path used by the BoardServer to access the Image folder on the fax server. BoardServers running on the same machine as the Fax Server Module should be using a drive letter-based path such as C:\Program Files\RightFax\Image. BoardServers on separate machines should be using UNC-based paths,such as \\FS1\Program Files\RightFax\Image.
SVE_FSBRDS_BFTDIR	ulIndex1 = BoardServer index (0–15) to query	Path used by the BoardServer to access the BFT folder on the fax server.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_FSBRDS_QUEUEDEPTHPAGES	ulIndex1 = BoardServer index (0–15) to query	Number of fax pages currently waiting to be sent on the BoardServer. Normally, “waiting” means waiting for an available outgoing fax channel/phone line. However, faxes delivered via Internet or IP fax service providers are “waiting” for delivery to the service provider.
SVE_FSBRDS_QUEUEDEPTHFAXES	ulIndex1 = BoardServer index (0–15) to query	Number of faxes waiting to be sent on the BoardServer.
SVE_FSBRDS_NUMCHANNELS	ulIndex1 = BoardServer index (0–15) to query	Number of configured channels on the BoardServer. The total number of channels configured on all BoardServers cannot exceed the licensed channel value of the Fax Server Module, or the Fax Server Module will shut itself down.
SVE_FSBRDS_DOWN	ulIndex1 = BoardServer index (0–15) to query	Returns a value of one (1) if the Fax Server Module considers the BoardServer down for some reason. The reason is described by the SVE_FSBRDS_ DOWNERROR value, but it is usually caused by a communications error, the BoardServer not up and running, an NT security violation, or a misconfiguration.
SVE_FSBRDS_DOWNTIME	ulIndex1 = BoardServer index (0–15) to query	A Julian time-stamp when the BoardServer was detected as down by the Fax Server Module.
SVE_FSBRDS_DOWNERROR	ulIndex1 = BoardServer index (0–15) to query	A error code indicating the cause of the “down” status. The error codes are always Win32 error codes.
SVE_FSBRDS_LOCAL	ulIndex1 = BoardServer index (0–15) to query	A value of one (1) indicates that the BoardServer is on the same machine as the Fax Server Module. A value of zero (0) indicates a remote BoardServer, e.g., one running on a machine/chassis different from that running the Fax Server Module.
SVE_FSBRDS_BRDSRVUNIQUEID	ulIndex1 = BoardServer index (0–15) to query	A character used by the BoardServer to name its inbound faxes so as to separate the file name spaces of multiple BoardServers. Each BoardServer sets its own unique ID.
SVE_FSBRDS_CURRENTLOAD	ulIndex1 = BoardServer index (0–15) to query	The availability index of the BoardServer as known by the Fax Server Module. The Fax Server Module caches the availability index, used in load balancing to reduce network traffic.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_FSBRDS_LOADTIMESTAMP	ulIndex1 = BoardServer index (0–15) to query	A Julian time-stamp indicating the age of the cached availability index.
SVE_FSBRDS_COMPACTSTATS1	ulIndex1 = BoardServer index (0–15) to query	Returns a COMPACTBRDSRVSTAT1 structure with most of the BoardServer metrics described above. This metric can be used to reduce the number of STATUSVALUEENTRY elements required to query all metrics concerning all BoardServers being driven by a fax server.

Gateway Module Status Metrics

The Gateway Modules are services responsible for routing faxes to email systems, converting email messages to faxes, and sending notifications to email destinations. More than one gateway can be used on a single fax server system (up to 16). Multiple gateways can be employed to support a variety of email systems, such as Exchange and SMTP, or for load-balancing and redundancy purposes. The Status APIs are not available for remote gateways, i.e., those not executing on the same machine as the Fax Server Module.

All SVE_GW_??? metrics require a Gateway index value, indicating from which Gateway to query status, in the ulIndex2 argument of the SERVERINFO structure. The index value must be in the range 0–15.

Metric	Additional argument required in ulIndex1 or ulIndex2?	Description
SVE_GW_INFOVERSION	ulIndex2 = Gateway from which to extract metric (0–15).	Version of status data presented by module. Returns a value of 0x465.
SVE_GW_SECONDSUP	ulIndex2 = Gateway from which to extract metric (0–15).	Number of seconds module has been executing. After 2^{31} seconds, the value resets to zero.
SVE_GW_MAILTYPE	ulIndex2 = Gateway from which to extract metric (0–15).	A string indicating the type of gateway, such as Exchange, Notes, SMTP, SAP, etc.
SVE_GW_MAILPO	ulIndex2 = Gateway from which to extract metric (0–15).	A string indicating the post office or mail server to which the gateway is connected.

Metric	Additional argument required in ulIndex1 or ulIndex2?	Description
SVE_GW_ACTIVITY	ulIndex1 = Activity log record to retrieve (0–*n*). ulIndex2 = Gateway from which to extract metric (0–15).	The Gateway Module maintains a circular log of descriptions of the last 50 activities executed. The SVE_GW_ACTRECORDS metric indicates the number of elements in the log (up to 50). The SVE_GW_ NEWESTACTINDEX metric indicates the newest entry in the log. Each entry in the log is a three-line textual description. Each line is 60 bytes (59 characters with a terminating zero byte). Line 1 starts are offset 0 (szData [0]), line 2 starts are offset 60 (szData[60]), and line 3 at offset 120 (szData [120]).
SVE_GW_NEWESTACTINDEX	None	Indicates the index (0–49) of the activity log entry that is the newest.
SVE_GW_ACTRECORDS	None	Indicates the number of elements in the circular activity log used. When all entries are used (50), older entries are overwritten, i.e., the nature of a circular log.

Global Server Metrics

Global Server Metrics are values that do not apply to a specific RightFax server module.

Metric	Description
SVE_GLOB_VERSION	Major version number of server, e.g., 5 or 6.
SVE_GLOB_MINORVERSION	Minor version of server,
SVE_GLOB_SERVICEPACK	Service pack installed or zero if none.
SVE_GLOB_ENTERPRISE	Returns a value of one (1) if the server is an Enterprise license.
SVE_GLOB_ CONFIGUREDWORKSERVERS	Number of configured WorkServer modules on the server. This value is actually one less than the number, so three workservers would return a value of 2, and zero WorkServers would return a value of -1.

Metric	Description
SVE_GLOB_ CONFIGUREDGATEWAYS	Number of configured Gateway modules on the server. This value is actually one less than the number, so one gateway would return a value of 0, and zero gateways would return a value of -1.
SVE_GLOB_CONFIGUREDCHANNELS	Number of fax channels configured on the server. Note that this value does not report the total number of channels being driven by the server, i.e., where a single Enterprise server is running multiple BoardServer modules on multiple NT machines.

RPC Server Module Status Metrics

The RPC Server Module implements the Status APIs on the fax server and is responsible for all RPC (TCP/IP, IPX/SPX, etc.) communications between the server and clients.

Metric	Description
SVE_RPC_INFOVERSION	Version of status data presented by module. Returns a value of 0x456.
SVE_RPC_SECONDSUP	Number of seconds module has been executing. After 231 seconds, the value resets to zero.
SVE_RPC_BYTESREAD	Number of bytes read from clients.
SVE_RPC_BYTESWRITTEN	Number of bytes written to clients.
SVE_RPC_CMDSPROCESSED	Number of client requests processed.

WorkServer Module Status Metrics

The WorkServer Modules are services responsible for executing operations taking a long time, where “long time” is defined as more than a few hundred milliseconds. By allowing multiple machines to host WorkServers, each with up to 16 WorkServers, RightFax servers can scale to handle a large volume of faxing even though each fax processed requires many CPU cycles of pre-processing, such as rendering, overlay, printing, OCR, etc. The Status APIs are not available for remote WorkServers, i.e., those not executing on the same machine as the Fax Server Module.

All SVE_WS_??? metrics require a WorkServer index value, indicating from which WorkServer to query status, in the ulIndex2 argument of the STATUSVALUEENTRY structure. The index value must be in the range 0–15.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_WS_INFOVERSION	ulIndex2 = WorkServer from which to extract metric (0–15).	Version of status data presented by module. Returns a value of 0x454.
SVE_WS_SECONDSUP	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of seconds module has been executing. After 231 seconds, the value resets to zero.
SVE_WS_ARCHIVES	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of archive operations executed.
SVE_WS_CONVERTS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of PCL-5 and cover sheet rendering operations executed.
SVE_WS_CVLS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of CVL (CVL = ConVert List) rendering operations executed. Each CVL can involve a number of other rendering operations, e.g., PCL-5, PostScript, SSA, etc.
SVE_WS_DELETES	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of file delete operations executed.
SVE_WS_FILEROUTES	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of received fax route-to-disk operations executed.
SVE_WS_OCRS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of OCR and OCR route operations performed.
SVE_WS_OVERLAYS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of form overlay operations performed.
SVE_WS_PRINTS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of fax print operations executed. This include automatic printing of sent or received faxes and on-demand printing requests from users.
SVE_WS_POSTSCRIPTS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of PostScript and PDF rendering operations executed.
SVE_WS_SENDMSGS	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of network popup message send operations executed.

Metric	Additional argument required in ulIndex1 or ulIndex2	Description
SVE_WS_XROUTES	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of InterConnect routing operations executed.
SVE_WS_TIMEFORLASTCMD	ulIndex2 = WorkServer from which to extract metric (0–15).	Number of seconds required to complete the last operation.
SVE_WS_LASTREQUEST	ulIndex2 = WorkServer from which to extract metric (0–15).	Type of operation last performed (see the REQUEST_??? constants for possible values).
SVE_WS_SERVICES	ulIndex2 = WorkServer from which to extract metric (0–15).	Bit-mask 32-bit value indicating the operations that the WorkServer is configured to perform.
SVE_WS_ACTIVITY	ulIndex1 = Activity log record to retrieve (0–*n*). ulIndex2 = WorkServer from which to extract metric (0–15).	The WorkServer module maintains a circular log of descriptions of the last 50 activities executed. The SVE_WS_ACTRECORDS metric indicates the number of elements in the log (up to 50). The SVE_WS_ NEWESTACTINDEX metric indicates the newest entry in the log. Each entry in the log is a three-line textual description. Each line is 60 bytes (59 characters with a terminating zero byte). Line 1 starts are offset 0 (szData [0]), line 2 starts are offset 60 (szData[60]), and line 3 at offset 120 (szData [120]).
SVE_WS_NEWESTACTINDEX	None	Indicates the index (0–49) of the activity log entry that is the newest.
SVE_WS_ACTRECORDS	None	Indicates the number of elements in the circular activity log used. After a short while, all entries are used (50), and older entries are overwritten, i.e., the nature of a circular log.