



COM API Reference EFT v8.0.7



Copyright Terms and Conditions

Copyright Help/Systems LLC and its group of companies.

The content in this document is protected by the Copyright Laws of the United States of America and other countries worldwide. The unauthorized use and/or duplication of this material without express and written permission from HelpSystems is strictly prohibited. Excerpts and links may be used, provided that full and clear credit is given to HelpSystems with appropriate and specific direction to the original content. HelpSystems and its trademarks are properties of the HelpSystems group of companies. All other marks are property of their respective owners.

202112200147

Using the Application Programming Interface (API)	11
What's New in the API?	12
Changes in v8.0.7	12
Using the API (with Examples)	13
Using the API	14
Scripting Standards	15
Difference between VARIANT_BOOL and SFTPAdvBool	16
Script Examples	17
Error Codes	21
Remotely Administering EFT™ Using the API	25
Copying Required DLL Files to the Remote Computer	25
Alphabetical List of Objects	29
Acceleration Module-Related Objects	29
Advanced Workflow Objects	30
ICIAWTaskActionParams Interface	30
ICIAdvancedWorkflowParams Interface	33
Advanced Workflow-Related Settings	37
Auditing and Reporting Module Objects	42
AS2-RelatedObjects	43
AS2 Objects	43
ICIAS2InboundSettings Interface	53
ICIAS2OutboundSettings Interface	65
ICIAS2SendActionParams Interface	84
ICIAS2TransactionInfo Interface	97
Authentication-Related Objects	101
ICIADAuthManagerSettings Interface	101
ICIGSAuthManagerSettings Interface	104

ICILDAPAuthManagerSettings Interface	105
ICIODBCAuthManagerSettings Interface	111
RSA Objects	112
RADIUS Objects	114
SAML Objects	117
Cloud-Related Interfaces	121
ICICloudAWSS3GeneralParams	121
ICICloudAWSS3UploadParams	125
ICICloudAWSS3DownloadParams	126
ICICloudAzureBlobGeneralParams	127
ICICloudAzureBlobUploadParams	129
ICICloudAzureBlobDownloadParams	130
ICICloudDownloadActionParams	131
ICICloudStorageMonitorEventRuleParams	133
ICICloudTransferActionParams	135
ICICloudUploadActionParams	136
ICIConnectionProfileParams	137
Content Integrity Control Objects	145
ICIContentIntegrityControlActionParams	146
ICIContentIntegrityControlProfile	147
DMZ Gateway-Related Objects	153
ICIDMZGateway Interface	154
Event Rule-Related Interfaces	165
ICIEventRule Interface	165
ICIEventRules Interface	168
ICIEventAction Interface	169
ICIEventActions Interface	170

ICIEventInfo Interface	171
ICIEventRuleParams Interface	172
ICIEventRuleFolder	174
ICIEventRuleFolders	175
ICIEventRuleStatement Interface	176
ICIActionStatement Interface	177
ICIActionStatements Interface	177
IAdhocMessage Interface	179
ICIAudit Interface	183
ICIBackupActionParams Interface	185
ICICallSubroutineActionParams Interface	186
ICICleanupActionParams Interface	186
ICICommandActionParams Interface	189
ICICommandSettings Interface	191
ICICompoundCondition Interface	197
ICICompressActionParams	198
ICIConnectionProfileParams	200
ICICsvExportActionParams	208
ICICsvImportActionParams	209
ICIDefineVariableActionParams Interface	211
ICIDownloadActionParams Interface	211
ICIFileActionFilter	211
ICIFileActionsChecksumParams	213
ICIFileActionConcatenateParams	214
ICIFileActionDeleteParams	215
ICIFileActionParams	216
ICIFileActionReadParams	218

ICIFileActionRenameParams	218
ICIFileActionWriteParams	220
ICIFolderActionParams	221
ICIFolderMonitorEventRuleParams Interface	222
ICIGetListingActionParams Interface	228
ICIfStatement Interface	236
ICILoopDatasetStatement	237
ICILoopDatasetStatementParams	239
ICIMailActionParams Interface	241
ICIOBJECTAccessRights	245
ICIOBJECTACE	246
ICIOBJECTACL	246
ICIPrivacyPolicy Interface	247
ICIReportActionParams Interface	249
ICIRESTInvocationAuthParams	254
ICIRESTInvocationEventRuleParams Interface	255
ICIRESTInvocationVerifyParams	257
ICIRoundRobinEventRuleSchedulingParams Interface	258
ICIRunPowerShellScriptActionParams Interface	258
ISATMessage	260
ICISimpleCondition Interface	262
ICIStopActionParams Interface	264
ICISynchronizeActionParams Interface	265
ICITimerEventRuleParams Interface	266
ICITimerEventRuleParams in Previous Versions	272
ICITermsOfService Interface	278
ICITransferActionParams Interface	280

ICIUploadActionParams	298
ICIUserAccountActionParams	298
ICIWebServiceActionParams	299
ICIWindowsEventLogActionParams	304
High Availability-Related Objects	306
ICHighAvailabilityParams Interface	306
ICIRoundRobinEventRuleSchedulingParams Interface	307
IP Access Rule-Related Objects	308
How Do the IP Access/Ban Methods and Properties Work?	308
IP Access Rule Settings	312
ICIAutobanIPAccessRule Interface	320
ICIBannedIP Interface	322
ICIManualIPAccessRule Interface	325
PGP-Related Objects	328
ICIPgpActionParams Interface	336
ICIPGPKeyInfo Interface	339
Privacy-Related Objects	341
GDPR and Privacy Settings	341
ICIClientDataField Interface	345
ICIGDPRSettings Interface	348
ICIPrivacyPolicy Interface	352
ICITermsOfService Interface	355
Server-Level Objects	357
ICIServerVersion Interface	357
ICIServer Interface	358
ICIServerInfo Interface	427
ICIServerNodeInfo	427

ICIServerNodeSiteInfo	431
ICIBackupArchiveInfo Interface	434
ICIAccountAdmin Interface	442
ICIAAdminID	445
ICIAAdminPermission Interface	446
ICISiteReportInfo Interface	447
ICISSLVersion	448
ICISSLVersions	448
Site-Level Interfaces	450
ICISites Interface	450
ICISite Interface	463
ICISiteBackupInfo Interface	597
ICISiteConnectedUser Interface	602
ICISiteEncrypted Folder	606
ICISiteCertInfo Interface	607
ICISiteClientDataField Interface	611
ICISiteComplexPasswordSettings Interface	614
ICISitePermission Interface	620
ICISiteResetPasswordSettings Interface	625
SFTP-Related Objects	628
ICISiteSSHKeyInfo Interface	629
SSL-Related Objects	630
User (Client)-Level Objects	632
ICISiteClientSettings Interface	632
ICISiteNewUserData Interface	680
ICISiteComplexPasswordSettings Interface	683
ICISiteConnectedUser Interface	689

ICIUserAccountActionParams	693
ICIAbortUserOperationActionParams	694
Web Objects	694
Workspaces Settings	694
ICIWorkspacePermissions	696
ICIWorkspace	697
ICIAddressBook	700
ICIPrivacyPolicy Interface	701
ICITermsOfService Interface	703
ICITwilioSmsProviderProfile	705
ICIUploadForm	707
ICIUploadFormElement	709
WTC-Related Settings	710
Enums	713

Using the Application Programming Interface (API)

The Enhanced File Transfer™ (EFT™) platform's API allows you to interact with EFT at a command line or in scripts. This reference does not teach you how to write scripts; it is simply a reference to describe the interfaces, properties, methods, and enumerations used in the API.

This reference assumes you know how to create and execute a script. Few examples are provided; if you need assistance with creating scripts, [Globalscape Professional Services](#) can create custom scripts for you.

The SFTPCOMInterface.CIServer class is the class that an application can use directly. To start using the SFTP interface, create the SFTPCOMInterface.CIServer class object, and then apply the methods and properties in the object. (If you do not understand what it means to "create the SFTPCOMInterface.CIServer class object," Globalscape Professional Services can create custom scripts for you.)

The interfaces are a way to instantiate an object of that type. For example:

```
Set SFTPServer = CreateObject("SFTPCOMInterface.CIServer")
```

("CIServer" implements the ICIServer interface.)

To find the interface, method, or property that you need to use in your script, use the Search feature in this reference to locate words or phrases (surrounded by quotation marks) within the content of the topics. (e.g., "certificate" yields ~50 topics; "certificate country" yields ~12 topics.)

For script examples, refer to [Script Examples](#).

Refer to the following Web sites for helpful information regarding scripting:

- [VBScript Run-time Errors](#) - Reference of run-time errors.
- [VBScript Syntax Errors](#) - Reference of syntax errors
- [Microsoft Windows Script Technologies](#) - Lists the Windows script technologies and describes the functionality included in each technology.

NOTE:

- Be sure to [heavily document or comment your script](#). Adding descriptive comments to your script now will remind you and others of the purpose of the script, and makes it easier to borrow sections of script to use later in other scripts. A year from now, when you don't even remember writing the script, you'll be glad you added comments!
- This guide is intended as a supplement to the [EFT Installation and Administration Guide](#). For an understanding of the concepts in this supplement, please refer to that guide for your version of EFT.

What's New in the API?

Each new version of the API offers improved programmatic control over EFT. If you are upgrading EFT, be sure to read the descriptions at the links below and update your scripts accordingly. If necessary, [Globalscape Professional Services](#) can assist you with writing scripts for your unique environment.

Included below are details of what is new in this version as well as what has changed in the API in previous versions.

Changes in v8.0.7

New Features

Added [ServerModule](#) enums to activate modules:

- Const EnterpriseActionsModule (v8.0.7 and later)
- Const FTPModule (v8.0.7 and later)
- Const RemoteAgentModule (v8.0.7 and later)
- Const SecureFormsModule (v8.0.7 and later)

Changes in v8.0.6

New Features

- Added ICIServer interface [CloseServerService](#) method
- Added ICISite interface [PGP properties](#):
 - PGPSendEmailExpirationCopyToAssociatedUser
 - PGPSendEmailPriorExpiration
 - PGPSendEmailUponExpiration

- PGPEmailExpirationRecipientsList
- PGPTIMEPriorExpirationInDays
- Added ICIPGPKeyInfo interface property: [UsageFlags](#)
- Added [ICIAbortUserOperationActionParams](#) interface
- Added [ICIEncryptedFolder](#) properties:
 - EncryptionKey
 - KeyAlias
- Added [Event Type](#) enums:
 - OnAS2Test,
 - OnAS2AsyncResend,
 - OnWorkspaceExpired,
 - OnSecureMessageComposed = 0xa001,
 - OnSecureMessageSent,
 - OnSecureMessageNotSent,
 - OnSecureMessageViewed,
 - OnSecureMessageAttachmentBeforeDownload,
 - OnSecureMessageAttachmentAfterDownload
- Added **EventActionType** enum:
 - [AbortUserOperationAction](#) = 536870912 (&H20000000)

Enhancements:

- Renamed **EventType** enum BeforeWorkspaceDeleted renamed to [OnWorkspaceBeforeDelete](#)

Using the API (with Examples)

This section provides general guidance and examples to demonstrate common uses of the API.

- [Using the API](#)
- [Scripting Standards](#)
- [Difference between VARIANT_BOOL and SFTPAAdvBool](#)
- [Script Examples](#)
- [Error Codes](#)
- [Remotely Administering EFT™ Using the API](#)

Using the API

To start using the API, create the **SFTPCOMInterface.CIServer** class object, and then apply the methods and properties in the object. For example:

```
Set SFTPServer = WScript.CreateObject
("SFTPCOMInterface.CIServer")
' Get rule's statement with index 0
```

To allow the FTP session via a command line to accept double slashes // when navigating paths, you must enable the following advanced property: **FixDoubleSlashInPathsForFTP**

false = disabled; 1 = true

Below is the general sequence:

1. Add a COM library reference to your project (e.g., 'Solution Explorer' => Right click on 'References' => 'Add Reference...' => 'COM tab' => 'SFTPCOMInterface 1.0 Type Library')
2. Add a **using** directive to your sources:
using SFTPCOMINTERFACELib;
3. Create the Server object:
CIServer server = new CIServer();
4. Connect to EFT:
server.Connect("Server_host", Server_port, "Admin_login", "Admin_Password");
5. Retrieve/modify sites, commands, users, event rules, and so on.

For example, to deal with Event Action statements ('If' statement and 'Action' statement), determine their exact types and then cast to it:

```
ICIEventRule rule = ...;
...
if (((ICIEventRuleStatement)rule.Statement(0)).type ==
    EventRuleStatementType.ActionStatement)
{
    ICIActionStatement actionStatement =
        (ICIActionStatement)rule.Statement(0);
    // manipulate with action
}
else
{
    ICIIIfStatement ifStatement = (ICIIIfStatement)rule.Statement(0);
```

```
// manipulate with condition
}
```

More Information

The API does not allow changing Event Rules if the change violates the Rule's integrity (invariant):

- You cannot use an Action/Condition if it is forbidden for this Event (e.g., the **AS2 Send Action** is forbidden for **On Server Stopped** events).
- You cannot change a Condition's value/operator if the value/operator is inconsistent with the Condition (e.g., **If Server Status Condition** does not allow the **Is One Of** operator).
- You cannot change an Action's parameters if new ones are inconsistent (e.g., 'action.Params = PGPParams' where PGPParams contains an empty file path or inconsistent pair, 'Operation = Encrypt' + 'Empty Key ID list').

Scripting Standards

As you identify tasks that you can perform with scripts, you will quickly have a large collection of scripts that you want to save, share with other administrators, and reuse. For ease of use and maintenance, you should develop a scripting standard that all administrators follow when creating and editing scripts in the collection. Following a standard style allows any administrator to open a script, know its purpose, and make any necessary customization. When you view/create scripts in Notepad, it is very difficult to separate comments from code. Following a standard scripting/coding style will make your scripts easier to read. There are also many tools (e.g., [EditPlus](#)) that display the various components in different colors, making reading easier.

Below are some suggestions for scripting standards.

- Comment section at the top of the script that contains the script's name, author, creation date, modified date, modified by, description (its purpose), and requirements/dependencies of the script (e.g., AD-specific, EFT Enterprise only), if any.
- Comments before each subsection stating the purpose of the section
- Comments regarding where the data or result is stored/comes from (e.g., writes the result to the Windows Event Viewer; appears in EFT Admin Actions report)
- Coding conventions might include naming conventions for objects, variables, and procedures; commenting conventions; and text formatting and indenting guidelines

The example below shows a basic header and commented subsections.

```

1  ' FILE: SetPermission
2  ' CREATED: 13 OCT 2004 RTO
3  ' PURPOSE: Creates a user, creates some virtual folders, and
sets various permissions.
4  ' DEPENDENCIES: Be sure to edit the script for your
connections parameters
5  ' (Admin ID, Password, Port, Server...).
6  ' =====
7  Private Sub Command1_Click()
8
9  Set oSFTPServer = CreateObject("SFTPCOMInterface.CIServer")
'instantiate EFT COM Object
10
11
12  ' =====
13  '   Initialize connection parameters
14  ' =====
15
16  Dim sAdminID, sAdminPassord, sAdminServer, oSites, oSite
17  sAdminID = "eft"'change this to your admin username
18  sAdminPassword = "eftserver"'change this to your admin
password
19  sAdminPort = "1100"'change this to your admin port
20  sAdminServer = "10.0.1.2"'change this to the IP address for
the admin connection to the Server

```

Difference between VARIANT_BOOL and SFTPAdvBool

In EFT, a check box can be selected (usually TRUE) or cleared (usually FALSE) in the user account, inherited from its Settings Template, or inherited from the Site. For this reason, the API needs the corresponding functions: VARIANT_BOOL and SFTPAdvBool.

When a script calls a function and wants to know the state (enabled/disabled) of that value, the result is either -1 (selected) or 0 (cleared). VARIANT_BOOL cannot return information about the inheritance of that value, only whether it is selected or cleared.

VARIANT_BOOL has only two possible values:

- VARIANT_FALSE(0)
- VARIANT_TRUE(-1)

However, a check box setting can be *inherited* (TRUE) and cleared (FALSE). For these functions, SFTPAdvBool is used instead of VARIANT_BOOL.

SFTPAdvBool has three possible values:

- abFalse = 0
- abTrue = 1
- abInherited = -2

For example, to make the state inherited, (-2) should be called.

Another parameter, plnherited, is similar to VARIANT_BOOL, but declared as a VARIANT due to VB limitations. plnherited is -1(VARIANT_TRUE) when pVal was inherited from the upper level and 0(VARIANT_FALSE) when it is not inherited.

Script Examples

The following scripts provide examples to guide you in your own script design. Each script is named per its function. For example, the ConfigureAdminUser script can be used to configure an administrative user account. If you do not have the software to open or edit CS or VBS files, you can open the script in a text editor to review the script for details of how it works and edit it. If you need help with custom scripts, contact [Globalscape Professional Services](#).

Each of the scripts described below is available in a ZIP file available at <https://help.globalscape.com/help/globalscape.com/api/scripts.zip>. The scripts are organized in the ZIP file by how it is to be used. (e.g., CreatUsers.cs is in the Users folder.) Do not double-click the script file to open it. Double-clicking will cause it to execute. Instead, right-click the file then click Open with and select your text editor of choice, or save it with a TXT extension, edit it in a text editor, and then save it with the CS or VBS extension.

IMPORTANT: Never use these scripts as is! These scripts are just examples. You must edit the scripts to reflect your environment (user accounts, passwords, paths, ports, and so on).

- Password-related scripts:
 - AllUsers-ForcePasswordChange.vbs - This script loops through all users of a specified Site and performs ForcePasswordChange to force all users to change their password at next login.
 - AllUsers-GetForcePasswordResetOnInitialLogon.vbs - This script gets a specific user's Password Reset on Initial Logon setting.

- ExportAllUsers-GetForcePasswordResetOnInitialLogon.vbs - This script exports ALL users Password Reset on Initial Logon setting.
- Test a connection to the ARM database - These scripts use the ICIServer [ARMTestConnection](#) and [ARMReconnect](#) methods.
 - ARM.cs
 - ARM.vbs
- Specify ARM parameters, restore configuration (from a backup), and then start the service - These scripts use ICIServer [BackupServerConfiguration](#), [GetBackupArchiveInfo](#), [RestoreServerConfigurationEx](#) methods (and others).
 - Backup.cs
 - Backup.vbs
 - BackupEx.cs
 - BackupEx.vbs
- Configure a specific Site - These scripts use ICISite methods such as [SetAllowFXP](#) to retrieve or set a specified Site's settings.
 - ConfigureSite.cs
 - ConfigureSite.vbs
- Configure a user account - These scripts use [ICISite](#), [ICIClientSettings](#), and [ICIResetPasswordSettings](#) methods to retrieve or set user account settings.
 - ConfigureUser.cs
 - ConfigureUser.vbs
- Create an administrator account - These scripts use [ICISite](#), [ICIClientSettings](#), and [ICIResetPasswordSettings](#) methods to retrieve or set user account settings.
 - CreateAdminUser.cs
 - CreateAdminUser.vbs
- Configure an administrator user account - These scripts use [ICIAccount](#) and [ICIAccountPermission](#) methods to configure an administrator account.
 - ConfigureAdminUser.cs
 - ConfigureAdminUser.vbs
- Create a Site - These scripts use the ICISites [AddLocalSite](#) method to create a new site on the server.
 - CreateSite.cs
 - CreateSite.vbs
- Create 1,000 users (for testing purposes) - These scripts use the ICISite [CreateUserEx](#) method to create 1,000 users.

- CreateUsers.cs
- CreateUsers.vbs
- Create an Event Rule - These scripts use [ICIFolderMonitorEventRuleParams](#), [ICIEventRule](#), [ICIDownloadActionParams](#), [ICIUploadActionParams](#), [ICITimerEventRuleParams](#), and [ICIAWTTaskActionParams](#) methods to create Event Rules on the Site.
 - EventRuleCreation.cs
 - OnTimerEventRuleCreation.vbs
 - OnTimerEventRuleForAwe.vbs
- Retrieve Trusted Certificate ID, description, and common name for a Site - These scripts use the ICISite [GetTrustedCertificates](#) method.
 - GetCertInfo.cs
 - GetCertInfo.vbs
- Configure an ODBC Site - These scripts use the [ICIODBCAuthManagerSettingsRefreshIntervalMinutes](#) and [UserDatabaseConnectionString](#) methods, and ICISite SetAMParams method (and others) to create an ODBC-authenticated Site.
 - ODBCAuthOptions.cs
 - ODBCAuthOptions.vbs
- Import PGP key - These scripts use the ICIServer [ImportSSHKey](#) and others to import a PGP key into the server.
 - PGP.cs
 - PGP.vbs
- Delete a Site from the Server - These scripts use the ICISites [Item](#) method to specify and then remove a Site.
 - RemoveSite.cs
 - RemoveSite.vbs
- Specify IP address access on a Site - These scripts use the [AddIPAccessRule](#) and [GetAllowedMasks](#) methods.
 - ServerIPBanEntry.cs
 - ServerIPBanEntry.vbs
- Retrieve a user's IP address access - These scripts use the ICIClientSettings [AddIPAccessRule](#) method.
 - UserIPBanEntry.cs
 - UserIPBanEntry.vbs

- Retrieve Server statistics (uptime, properties, last modified time, last modified by account) - These scripts use the ICIServer [LastModifiedBy](#) and [LastModifiedTime](#) properties.
 - ServerStats.cs
 - ServerStats.vbs
- Retrieve Site statistics (last modified time, last modified by account) - These scripts use the ICISite [LastModifiedBy](#) and [LastModifiedTime](#) properties.
 - SiteStats.cs
 - SiteStats.vbs
- Export each SSH key and re-import them with an altered name, create list, add all available keys to a user, assign all keys in the list to the user - These scripts use the ICIServer [AvailableSSHKeys](#) and ICIClientSettings SetSSHKeyIDs methods (and others) to import and assign an SSH key to a user.
 - SSH.cs
 - SSH.vbs
- Create a User Settings Template - These scripts use ICISite methods [GetUserSettings](#), [GetSettingsLevelSettings](#), and others, and [ICIResetPasswordSettings](#) methods to create and configure a User Settings Template.
 - UserSettingsTemplates.cs
- Retrieve user account statistics (last modified time, last modified by account, whether account is locked) - These scripts use the [ICIClientSettings](#) methods and properties.
 - UserStats.cs
 - UserStats.vbs
- Retrieve VFS settings for all folders for a particular Site - These scripts use the ICISite methods [GetFolderList](#), [GetFolderPermissions](#), [IsVirtualFolder](#), [IsEncrypted](#) (and others) to retrieve VFS settings for the Site.
 - VFS.cs
 - VFS.vbs
- Retrieve and update Windows authentication options for an Active Directory site - These scripts use the [ICIADAuthManagerSettings](#) properties.
 - WindowsAuthOptions.cs
 - WindowsAuthOptions.vbs

Error Codes

The table below list error codes you might see when using the COM API.

Return code	Return value
MXE_OK	MXError:0(00000000)
MXE_WINSOCK	MXError:1(00000001)
MXE_OBJECT_CREATE	MXError:2(00000002)
MXE_OUT_OF_MEMORY	MXError:3(00000003)
MXE_MESSAGE_IS_TOO_LONG	MXError:4(00000004)
MXE_STR_IS_TOO_LONG	MXError:5(00000005)
MXE_BIN_IS_TOO_LONG	MXError:6(00000006)
MXE_INVALID_MESSAGE	MXError:7(00000007)
MXE_INVALID_MESSAGE_SIZE	MXError:8(00000008)
MXE_INVALID_USER_NAME	MXError:9(00000009)
MXE_INVALID_PASSWORD	MXError:10(0000000A)
MXE_LOGIN_INCORRECT	MXError:11(0000000B)
MXE_CLIENT_TOO_OLD	MXError:12(0000000C)
MXE_CLIENT_TOO_NEW	MXError:13(0000000D)
MXE_NAME_TOO_LONG	MXError:14(0000000E)
MXE_NAME_TOO_SHORT	MXError:15(0000000F)
MXE_PASSWORD_TOO_LONG	MXError:16(00000010)
MXE_PASSWORD_TOO_SHORT	MXError:17(00000011)
MXE_VAL_IS_OUT_OF_RANGE	MXError:18(00000012)
MXE_REQUESTED_PARAM_IS_ABSENT	MXError:19(00000013)
MXE_ALREADY_LOGGED_IN	MXError:20(00000014)
MXE_USER_EXIST	MXError:21(00000015)
MXE_NOT_LOGGED_IN	MXError:22(00000016)
MXE_USER_NOT_FOUND	MXError:23(00000017)
MXE_NOTHING_WAS_FOUND	MXError:24(00000018)
MXE_OBJECT_IN_DESTRUCTION_STATE	MXError:25(00000019)
MXE_CANNOT_ADD_SELF	MXError:26(0000001A)
MXE_ALREADY_EXIST	MXError:27(0000001B)
MXE_VF_NOT_CREATED	MXError:28(0000001C)
MXE_USER_REGISTERED	MXError:29(0000001D)
MXE_SSL_NOT_FOUND	MXError:30(0000001E)
MXE_SSL_ERROR	MXError:31(0000001F)
MXE_INVALID_SITE_ID	MXError:32(00000020)

Return code	Return value
MXE_NOT_REGISTERED	MXError:33(00000021)
MXE_COMMAND_SOCKET_ERROR	MXError:34(00000022)
MXE_SSL_SOCKET_ERROR	MXError:35(00000023)
MXE_NO_AUTHMANAGER	MXError:36(00000024)
MXE_ROOT_FOLDER_INCORRECT	MXError:37(00000025)
MXE_SFTP_ERROR	MXError:38(00000026)
MXE_SFTP_SOCKET_ERROR	MXError:39(00000027)
MXE_HTTP_ERROR	MXError:40(00000028)
MXE_HTTP_SOCKET_ERROR	MXError:41(00000029)
MXE_HTTPS_SOCKET_ERROR	MXError:42(0000002A)
MXE_EVENT_INVALID	MXError:43(0000002B)
MXE_SITE_PGP_PASS_INVALID	MXError:44(0000002C)
MXE_ACCESS_DENIED	MXError:45(0000002D)
MXE_INVALID_GATEWAY_ID	MXError:46(0000002E)
MXE_PASSWORD_INVALID_POLICY	MXError:47(0000002F)
MXE_ARM_TEST_CONNECTION_FAILED	MXError:48(00000030)
MXE_ARM_CONNECTION_FAILED	MXError:49(00000031)
MXE_ACCESS_DELEGATION_NOT_FOUND	MXError:50(00000032)
MXE_HISTORY_PASSWORD_FOUND	MXError:51(00000033)
MXE_NEED_TO_REFRESH	MXError:52(00000034)
MXE_UNRESOLVED_CONFLICTS	MXError:53(00000035)
MXE_EVENT_PGP_PASS_INVALID	MXError:54(00000036)
MXE_REPORT_RENDERING_FAILED	MXError:55(00000037)
MXE_INVALID_AMDB_PATH	MXError:56(00000038)
MXE_PASSWORD_EXPIRED	MXError:57(00000039)
MXE_PASSWORD_INITIAL_RESET	MXError:58(0000003A)
MXE_SFTP_INVALID_KEYPAIR	MXError:59(0000003B)
MXE_AS2_EMPTY_ID	MXError:60(0000003C)
MXE_AS2_BAD_SERVER_CERTIFICATE	MXError:61(0000003D)
MXE_AS2_BAD_OUTBOUND_PARTNER_CERTIFICATE	MXError:62(0000003E)
MXE_PLAIN_AS2	MXError:63(0000003F)
MXE_AS2_INVALID_HOT_FOLDER	MXError:64(00000040)
MXE_INVALID_ST_NAME	MXError:65(00000041)
MXE_BACKUP_NOT_FOUND	MXError:66(00000042)
MXE_BACKUP_INVALID_CREDENTIALS	MXError:67(00000043)
MXE_SSL_WEAK_CERT	MXError:68(00000044)
MXE_FIPS_INIT_ERROR	MXError:69(00000045)

Return code	Return value
MXE_AS2_BAD_INBOUND_PARTNER_CERTIFICATE	MXError:70(00000046)
MXE_AW_INVALID_TASK_NAME	MXError:71(00000047)
MXE_AW_RENAME_TASK_FAILED	MXError:72(00000048)
MXE_SFTP_WEAK_KEY	MXError:73(00000049)
MXE_SERVER_RESTORATION_FAILED	MXError:74(0000004A)
MXE_AM_TEST_FAILED	MXError:75(0000004B)
MXE_COMMAND_NOT_EXIST	MXError:76(0000004C)
MXE_SSH_UNIQUE_KEY_NAME_REQUIRED	MXError:77(0000004D)
MXE_SSH_NO_PUBLIC_KEYS	MXError:78(0000004E)
MXE_SSH_PARSING_ERROR	MXError:79(0000004F)
MXE_GATEWAY_LICENSE_ERROR	MXError:80(00000050)
MXE_AUTH_METHOD_UNALLOWED	MXError:81(00000051)
MXE_FOLDER_NOT_FOUND	MXError:82(00000052)
MXE_SERVER_REGISTERED	MXError:83(00000053)
MXE_INVALID_TER_RESPONSE	MXError:84(00000054)
MXE_TER_RESPONSE_EXPIRED	MXError:85(00000055)
MXE_LOG_NOT_FOUND	MXError:86(00000056)
MXE_INVALID_AUTH_SETTINGS	MXError:87(00000057)
MXE_PASSWORD_NEED_TO_REMIND	MXError:88(00000058)
MXE_EVENTRULE_NOT_FOUND	MXError:89(00000059)
MXE_EVENTRULE_CHANGEDESC_MISSING	MXError:90(0000005A)
MXE_LISTENING_IP_INVALID	MXError:96(00000060)
MXE_INVALID_FOLDER_NAME	MXError:97(00000061)
MXE_CANT_CREATE_PGP_LOG_PATH	MXError:98(00000062)
MXE_SSL_FIPS_INVALID_DIGEST_ALG	MXError:99(00000063)
MXE_EVENT_FM_FOLDER_NOT_ACCESSIBLE	MXError:100(00000064)
MXE_SHAREDDATA_TIMEOUT	MXError:101(00000065)
MXE_CLUSTER_START_FAILURE	MXError:102(00000066)
MXE_CLUSTER_MSMQ_NOT_INSTALLED	MXError:103(00000067)
MXE_ADMIN_CHANNEL_IN_USE	MXError:104(00000068)
MXE_CONFIG_VERSION_NOT_CURRENT	MXError:105(00000069)
MXE_EVENT_AW_TASK_DOES_NOT_EXIST	MXError:106(0000006A)
MXE_EVENT_COMMAND_DOES_NOT_EXIST	MXError:107(0000006B)
MXE_CONNECTION_PROFILE_DOES_NOT_EXIST	MXError:108(0000006C)
MXE_CLIENT_VERSION_INCOMPATIBLE	MXError:110(0000006E)
MXE_CLIENT_VERSION_MORE_RECENT	MXError:116(00000074)
MXE_FOLDER_NOT_EMPTY	MXError:117(00000075)

Return code	Return value
MXE_ETL_RENAME_TASK_FAILED	MXError:118(00000076)
MXE_CANNOT_IMPORT_EVENT_RULE	MXError:119(00000077)
MXE_CANNOT_EXPORT_EVENT_RULE	MXError:120(00000078)
MXE_NO_GUEST_ST	MXError:121(00000079)
MXE_BACKUP_NOT_COMPATIBLE	MXError:122(0000007A)
MXE_WEBSSO_REQUIRES_SSL_KEY	MXError:123(0000007B)
MXE_SEND_TEST_EMAIL_FAILED	MXError:124(0000007C)
MXE_INVALID_SMTP_CONFIG	MXError:125(0000007D)
MXE_SSL_CIPHER_STRING_INVALID	MXError:126(0000007E)
MXE_SSL_PROTOCOL_VERSION_MASK_INVALID	MXError:127(0000007F)
MXE_SSL_CREATE_CERT_PKCS_FAILED	MXError:128(00000080)
MXE_SSL_CERT_SIGN_FAILED	MXError:129(00000081)
MXE_PATH_ALREADY_RESERVED	MXError:130(00000082)
MXE_BUFFER_OVERFLOW	MXError:256(00000100)
MXE_SERVER_INTERNAL0	MXError:65280(0000FF00)
MXE_SERVER_INTERNAL1	MXError:65281(0000FF01)
MXE_SERVER_INTERNAL2	MXError:65282(0000FF02)
MXE_SERVER_INTERNAL3	MXError:65283(0000FF03)
MXE_SERVER_INTERNAL4	MXError:65284(0000FF04)
MXE_SERVER_INTERNAL5	MXError:65285(0000FF05)
MXE_PF_NOT_CREATED	MXError:65286(0000FF06)
MXE_GATEWAY_INVALID_HOSTNAME	MXError:65287(0000FF07)
MXE_INVALID_ACCELERATION_PORT	MXError:65288(0000FF08)
MXE_ACCELERATION_ALREADY_IN_USE	MXError:65289(0000FF09)
MXE_INVALID_ACCELERATION_PARAM	MXError:65290(0000FF0A)

Remotely Administering EFT™ Using the API

You can remotely administer EFT using the API, but you must first copy the applicable DLL files to a folder on the remote computer and register SFTPCOMInterface.DLL on the remote computer using regsvr32 ([described below](#)).

NOTE: Remote administration is not allowed after the trial expires if you do not activate the software.

Before you can connect from the remote Administrator, you must configure the Server. You must do this locally, on the EFT computer.

- Before attempting to connect to a remote EFT, first be sure that the [remote EFT service is running](#), and that it [allows remote administration](#).

If you have configured remote administration, but are unable to connect to EFT, one or more of the following could be preventing the connection:

- The IP address of the computer on which you are attempting to connect to EFT is listed in the [Remote Administration Ban IP](#) list.
- Your SSL certificate is expired or invalid.
- [Remote administration is not enabled](#).
- The [remote administration port value](#) has changed.
- [EFT's IP address](#) has changed since the last login.
- The firewall settings of the computer on which EFT is installed are blocking the connection.
- There is a version mismatch between your administration interface and the EFT service you are trying to administer.
- The [administrator account](#) with which you are attempting the remote connection does not have access permission to EFT.
- Network errors

Copying Required DLL Files to the Remote Computer

Administering EFT remotely requires that you copy the following DLL files from the EFT computer to the computer that will be making the remote COM calls (depending on your

needs, as indicated below).

- Required by the COM API. These files are installed on the EFT computer in C:\Program Files(x86)\Common Files\GlobalSCAPE\SFTPCOMInterface:
 - atl100.dll
 - mfc100.dll
 - mfc100u.dll
 - msvc100.dll
 - msucr100.dll
 - SFTPCOMInterface.dll (You must [register this file](#) on the remote computer.*)
- Required by the COM API and the SSL implementation. These DLLs are required anytime COM connects to EFT, even when the Require SSL for remote administration check box is not selected in the administration interface. These files are installed on the EFT Server computer in C:\Program Files(x86)\Common Files\GlobalSCAPE\SFTPCOMInterface:
 - SSL.dll
 - SSLFips.dll
 - GSCrypto.dll
 - GSCrypto.dll.sha1
- Required to create or administer SSH Keys via COM. These files can be found installed on EFT in the install directory of EFT, by default C:\Program Files(x86)\GlobalSCAPE\EFT Server Enterprise:
 - Sftp.dll sftp
 - FIPS.dll
 - cryptopp.dll

*All DLL files should be copied to the same folder as the SFTPCOMInterface.dll and should not be moved after the SFTPCOMInterface.dll is registered using regsvr32.exe, as described below:

To register SFTPCOMInterface.dll

- In the directory on the remote computer to which you copied the above files, run **regsvr32**. For example, type:

```
Regsvr32.exe SFTPCOMInterface.dll
```

For more information about the Registration Tool, refer to [Explanation of Regsvr32 usage and error messages](#) on the Microsoft Support site.

Below are several facts to consider regarding remote administration:

- SSL certificates cannot be created or managed remotely. You are prohibited from creating certificates for EFT while remotely administering EFT because this action can create a security breach. Any certificates you create remain on the computer on which you created them, unless you take steps to deliver and associate these files with another computer.
- Organizations complying with the PCI DSS are required to use SSL for remote administration. If you attempt to allow remote administration on a High Security Site, a message warns you that this setting violates PCI DSS 2.3, and allows you to continue with reason or disable the feature.
- File browse operations are disabled for remote administration. You can, however, type a path that is relevant to the EFT computer (not the remote interface). You are able to browse for a Settings Template folder, because you are browsing the VFS, not the physical folders.
- When the trial period has expired, all remote connections are disallowed.
- You cannot configure remote administration remotely.
- You must configure the local connection before you can configure a remote connection.
- For remote Active Directory connections, the connecting account must have access to the computer on which EFT is installed.
- You should restrict remote administration to one or more known static IP addresses.
- By default, all IP addresses are granted remote access to EFT. EFT allows you to grant access to only one specific IP address or a range of IP addresses, or deny access to one specific address or a range of addresses.
- For command-line login, the EFT listening IP address must be set to a specific IP address, not All Incoming. Remote administration must be configured and EFT must be in the same domain as the computer from which you are attempting to log in.
- If you are logged in to EFT remotely, your username and password are passed to the Windows System Services on the computer running EFT. The account that you log on with must have administrative rights to make any changes to the Globalscape EFT service running on that computer.
- If you are using SQL Express as your database, you may not be able to generate a report remotely, unless the connecting account is a trusted SQL Server connection (e.g., if SQL Server and the remote computer are in the same domain, or if SQL Server is configured to allow "mixed authentication.")

- When objects are created, added, removed, modified, enabled, disabled, started, or stopped remotely, the action is logged to the database and reported in the Administrator Actions Log.
- The EFT variable for remote EFT connections is %CONNECTION.REMOTE_IP%.

Alphabetical List of Objects

This chapter contains all of the COM API objects, grouped by feature or function.

Acceleration Module-Related Objects

The Acceleration module objects appear in the [ICIDMZGateway](#) interface and [Enum reference](#).

In the ICIDMZGateway Interface:

- Property **AccelerationPort** As Long

Retrieves or specifies the acceleration port.

```
HRESULT AccelerationPort([out,retval] LONG* pVal);
HRESULT AccelerationPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Acceleration port: " & CStr(oGW.AccelerationPort)
```

Specifying:

```
oGW.AccelerationPort = 1234
```

EFT v7.3 and later

- Property **EnableHTTPSAcceleration** As Boolean

Retrieves or specifies whether to enable acceleration.

```
HRESULT EnableHTTPSAcceleration([out,retval] VARIANT_BOOL* pVal);
HRESULT EnableHTTPSAcceleration(VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Acceleration enabled: " & CStr
(oGW.EnableHTTPSAcceleration)
```

Specifying:

```
oGW.EnableHTTPSAcceleration = True
```

EFT v7.3 and later

In the [Enum Reference](#):

The ServerModule enumerator is used in the [ICIServer](#) and [ICIServerNodeInfo](#) interface [ModuleRegistrationState](#) method to retrieve an EFT module state. Used with [RegistrationState](#).

- Const AccelerateModule = 14 (v7.3 and later)
 - Function **ModuleRegistrationState**(m As ServerModule) As RegistrationState

```
HRESULT ModuleRegistrationState([in] ServerModule m,  
[out, retval] RegistrationState* pRS);
```

Advanced Workflow Objects

This chapter contains AWE-related COM objects.

ICIAWTaskActionParams Interface

Use ICIAWTaskActionParams to create, modify, retrieve, and execute Automated Workflow Event Rule Actions.

interface ICIAWTaskActionParams : IDispatch

Examples (VBScript):

```
If ActionStatement.Action.Type = AWTaskAction Then  
Set AWTaskParams = ActionStatement.Action.Params  
` Modify AW Task parameters  
  and assign them back to the action  
` . . .  
ActionStatement.Action.Params  
  = AWTaskParams  
End If
```

The CIAWTaskActionParams co-class supports creation of the AW Task Event Action. The co-class implements ICIAWTaskActionParams interface.

```
Set AWTaskParams = CreateObject  
("SFTPCOMInterface.CIAWTaskActionParams")  
` Setup AW task parameters
```

```
\ . . .
Rule.AddActionStatement(0, AWTTaskParams)
```

Retrieving:

```
MsgBox AWTTaskParams.TaskName & " task will be
executed"
```

Specifying:

```
AWTaskParams.TaskName = "Sample SNMP Get"
```

NOTE: "Function" and "Sub" indicate that the member is a method; "Property" indicates that it is a property.

Members

- Sub **AddVariable**(bstrVarName As String, bstrVarValue As String)

Adds new variable

```
HRESULT AddVariable([in] BSTR bstrVarName,[in] BSTR
bstrVarValue);
```

Example (VBScript):

```
oAWT.AddVariable "Variable_0", "Value_0"
```

EFT v6.3 and later

- Function **GetVariableName**(nIndex As Long) As String

Retrieves a variable name by index.

```
HRESULT GetVariableName([in] LONG nIndex,[out,retval] BSTR*
pVal);
```

Example (VBScript):

```
MsgBox "Variable name at 0 index is: " & CStr
(oAWT.GetVariableName(0))
```

EFT v6.3 and later

- Function **GetVariableValue**(nIndex As Long) As String

Retrieves a variable value by index.

```
HRESULT GetVariableValue([in] LONG nIndex, [out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Variable value at 0 index is: " & CStr  
(oAWT.GetVariableValue(0))
```

EFT v6.3 and later

- Sub **RemoveVariable**(nIndex As Long)

Remove variable by index.

```
HRESULT RemoveVariable(  
[in] LONG nIndex);
```

Example (VBScript):

```
oAWT.RemoveVariable 0
```

EFT v6.3 and later

- Sub **SetVariable**(nIndex As Long, bstrVarName As String, bstrVarValue As String)

Set new variable name and value by index

```
HRESULT SetVariable(  
[in] LONG nIndex,  
[in] BSTR bstrVarName,  
[in] BSTR bstrVarValue);
```

Example (VBScript):

```
oAWT.AddVariable 1, "NewName", "NewValue"
```

EFT v6.3 and later

- Property **TaskName** As String

Retrieve or specify the name of an AW task (workflow) to execute.

```
HRESULT TaskName([out, retval] BSTR* pVal);  
HRESULT TaskName([in] BSTR newVal);
```

EFT v6.1 and later

- Property **VariablesCount** As Long read-only

Retrieves the number of variables.

```
HRESULT VariablesCount([out, retval] LONG* pVal);
```

Example (VBScript), retrieving:

```
MsgBox "Variable count is: " & oAWT.VariablesCount
```

EFT v6.3 and later

- Property **WaitUntilCompletes** As Boolean

Specify whether the Event Rule should wait for this action to complete.

```
HRESULT WaitUntilCompletes([out, retval] VARIANT_BOOL  
*pVal);  
HRESULT WaitUntilCompletes([in] VARIANT_BOOL newVal);
```

EFT v8.0.5

ICIAdvancedWorkflowParams Interface

The ICIAdvancedWorkflowParams interface allows you to create/modify/retrieve Advanced Workflow configuration.

Example (VBScript):

```
`Edit workflow  
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")  
if idxAW <> -1 Then  
Set AWparams = site.GetAdvancedWorkflowParams(idxAW)  
`Change AWparams properties  
`...  
`and apply changes  
site.SetAdvancedWorkflowParams idxAW, AWparams  
End If
```

The CIAdvancedWorkflowParams co-class supports the creation of Advanced Workflows, and implements the ICIAdvancedWorkflowParams interface.

Example (VBScript):

```
set AWParams = CreateObject  
("SFTPCOMInterface.CIAdvancedWorkflowParams")  
`...setup workflow parameters  
site.AddAdvancedWorkflow(AWParams)
```

Properties

- Property **Code**

Retrieves or specifies the source code of Advanced Workflow.

```
HRESULT Code([out, retval] BSTR *pVal); HRESULT Code([in]
BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox AWparams.Code
```

Specifying:

```
` Let some_code_from_file be a content of a file
` produced by AW Task Builder
AWparams.Code = some_code_from_file
```

EFT v6.5 and later

- Property **DateCreated** (Read-only)

Retrieves the time when the Advanced Workflow was created.

```
HRESULT DateCreated([out, retval] DATE *pVal);
```

Example (VBScript):

```
MsgBox "Task was created: " & CStr(AWparams.DateCreated)
```

EFT v6.5 and later

- Property **Description**

Retrieves or specifies the description of Advanced Workflow.

```
HRESULT Description([out, retval] BSTR *pVal); HRESULT
Description([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox AWparams.Description
```

Specifying:

```
`Mark as updated
AWparams.Description = AWparams.Name & " (updated)"
```

EFT v6.5 and later

- Function **GetACL()** As Object

```
HRESULT GetACL([out, retval] IDispatch** ppVal);
```

See also [SetACL](#).

EFT v7.2

- Property **LogMode**

(See [AWLogMode](#) enum.)

Retrieves or specifies the debug logging mode of the Advanced Workflow

```
HRESULT LogMode([out, retval] AWLogMode *pVal); HRESULT
LogMode([in] AWLogMode newVal);
```

Example (VBScript):

Retrieving:

```
If AWparams.LogMode = 0 Then
  ` no logging...
End If
```

Specifying:

```
` Turn on most detailed logging
AWparams.LogMode = 3
```

EFT v6.5 and later

- Property **Name**

Retrieves or specifies the name of Advanced Workflow.

```
HRESULT Name([out, retval] BSTR *pVal); HRESULT Name([in]
BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox AWparams.Name
```

Specifying:

```
`Mark as updated
AWparams.Name = AWparams.Name & " (updated)"
```

EFT v6.5 and later

- Sub SetACL(pVal As Object)

```
HRESULT SetACL([in] IDispatch* pVal);
```

See also [GetACL](#).

EFT v7.2

- Property **TimeoutSeconds**

Retrieves or specifies the maxim duration (in seconds) of the Advanced Workflow execution after which it is forcibly terminated by EFT. (Value of 0 means "no timeout".)

```
HRESULT TimeoutSeconds([out, retval] long *pVal); HRESULT
TimeoutSeconds([in] long newVal);
```

Example (VBScript):**Retrieving:**

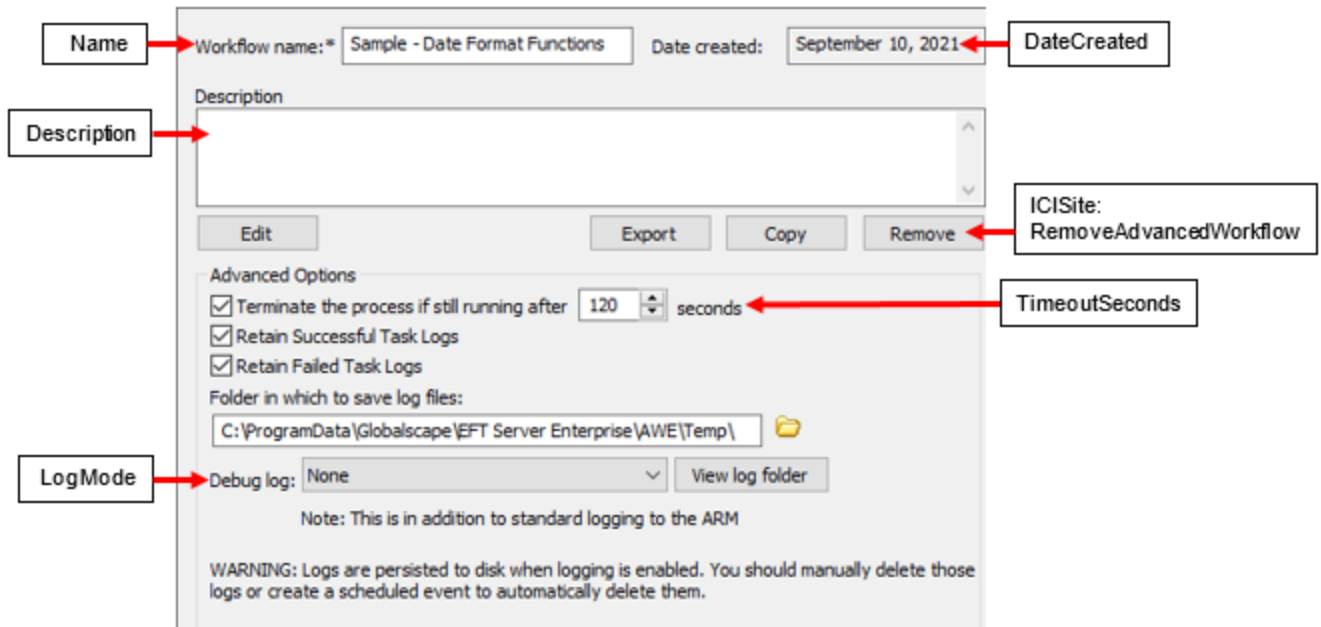
```
If AWparams.TimeoutSeconds = 0 Then
  ` no timeout...
End If
```

Specifying:

```
` Set max duration of workflow execution to 2 minutes
AWparams.TimeoutSeconds = 120
```

EFT v6.5 and later

Dialog Box Equivalents



Advanced Workflow-Related Settings

In addition to the [ICIAdvancedWorkflowParams Interface](#) and [ICIAWTaskActionParams Interface](#), Advanced Workflow-related members and properties can be found in the [ICISite Interface](#) and [Enum Reference](#).

In the ICISite interface:

- Function **AddAdvancedWorkflow**(pdisplParams As Object) As Long
Adds new Advanced Workflow to this Site.

```
HRESULT AddAdvancedWorkflow( [in] IDispatch* pdisplParams,
// AW params // as ICIAdvancedWorkflowParams* l[out, retval]
long* plIndex); // zero-based index of created AW
```

Example (VBScript):

```
set AWParams = CreateObject
("SFTPCOMInterface.CIAdvancedWorkflowParams")
'...setup workflow
site.AddAdvancedWorkflow(AWParams)
```

EFT v6.5 and later

- Property **AdvancedWorkflowsCount** As Long

Retrieves the total count of site's Advanced Workflows.

```
HRESULT AdvancedWorkflowsCount([out, retval] long* plValue);
```

Example (VBScript):

```
If Site.AdvancedWorkflowsCount = 0 Then
'Site has no Advanced Workflows...
End If
```

EFT v6.5 and later

- Function **GetACLForAdvancedWorkflowsRoot()** As Object

See also **SetACLForAdvancedWorkflowsRoot**.

```
HRESULT GetACLForAdvancedWorkflowsRoot([out, retval]
IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetACLForConnectionProfilesRoot()** As Object

See also **SetACLForConnectionProfilesRoot**.

```
HRESULT GetACLForConnectionProfilesRoot([out, retval]
IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetACLForCustomCommandsRoot()** As Object

```
HRESULT GetACLForCustomCommandsRoot([out, retval]
IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetACLForEventRulesRoot()** As Object

```
HRESULT GetACLForEventRulesRoot([out, retval] IDispatch**
ppVal);
```

EFT v7.2 and later

- Function **GetAdvancedWorkflowIndex** (bzName As String) As Long

Retrieves the zero-based index of the Advanced Workflow with the given name.

```
HRESULT GetAdvancedWorkflowIndex(
[in] BSTR bzName, // AW name
[out, retval] long* plIndex); // zero-based index of the AW
// or -1 if not found
```

Example (VBScript)

```
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")
if idxAW < 0 Then
  'Work with workflow via its index
End If
```

EFT v6.5 and later

- Function **GetAdvancedWorkflowParams**(lIndex As Long) As Object

(See also **SetAdvancedWorkflowParams**)

Given Advanced Workflow zero-based index, retrieves the Workflow's parameters.

```
HRESULT GetAdvancedWorkflowParams([in] long lIndex, [out, retval] IDispatch**
ppdisplParams);
```

Example (VBScript):

```
'Display the name in message for each workflow
For idxAW = 0 to site.AdvancedWorkflowsCount - 1
  Set AWparams = site.GetAdvancedWorkflowParams(idxAW)
  MsgBox AWparams.Name
Next
```

EFT v6.5 and later

- Sub **RemoveAdvancedWorkflow**(lIndex As Long)

Given the Advanced Workflow zero-based index, removes the Workflow from this site

```
HRESULT RemoveAdvancedWorkflow(
[in] long lIndex); // zero-based index if the AW to remove
```

Example (VBScript):

```
'Remove workflow by name
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")
if idxAW < 0 Then
  site.RemoveAdvancedWorkflow idxAW
End If
```

EFT v6.5 and later

- Sub **SetACLForAdvancedWorkflowsRoot**(pVal As Object)

See also **GetACLForAdvancedWorkflowsRoot**.

```
HRESULT SetACLForAdvancedWorkflowsRoot([in] IDispatch*
pVal);
```

EFT v7.2 and later

- Sub **SetACLForConnectionProfilesRoot**(pVal As Object)

See also **GetACLForConnectProfilesRoot**.

```
HRESULT SetACLForConnectionProfilesRoot([in] IDispatch*
pVal);
```

EFT v7.2 and later

- Sub **SetACLForCustomCommandsRoot**(pVal As Object)

See also **GetACLForCustomCommandsRoot**.

```
HRESULT SetACLForCustomCommandsRoot([in] IDispatch* pVal);
```

EFT v7.2 and later

- Function **SetACLForEventRulesRoot**(pVal As Object)

See also **GetACLForEventRulesRoot**.

```
HRESULT SetACLForEventRulesRoot([in] IDispatch* pVal);
```

EFT v7.2 and later

- Sub **SetAdvancedWorkflowParams**(lIndex As Long, pdisplParams As Object)

(See also **GetAdvancedWorkflowParams**)

Given Advanced Workflow zero-based index, changes the Workflow's parameters

```
HRESULT SetAdvancedWorkflowParams(
[in] long lIndex, // zero-based index of the AW
[in] IDispatch* pdisplParams); // AW parameters
// as ICIAdvancedWorkflowParams*
```

Example (VBScript):

```
'Rename workflow
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")
if idxAW >= 0 Then
Set AWparams = site.GetAdvancedWorkflowParams(idxAW)
AWparams.Name = "My Workflow - new name"
site.SetAdvancedWorkflowParams idxAW, AWparams
End If
```

EFT v6.5 and later

In the Enum Reference:**AWLogMode (v6.5 and later)**

AWLogMode enumeration represents different Advanced Workflow debug logging modes (see also `ICIAdvancedWorkflowParams::LogMode`).

- AWLogNone = 0; no logging
- AWLogMinimal = 1; minimal logging
- AWLogNormal = 2; normal logging
- AWLogVerbose = 3; verbose logging

EventActionType (v6.0 and later)

Used in the `ICIEventAction` Interface type property.

- AWTTaskAction = 1024 (added in v6.1)

ServerModule (v6.1 and later)

Used in the `ICIServer` Interface `ModuleRegistrationState` method to retrieve an EFT module state. Used with the `RegistrationState` enumerator.

- AWModule = 8

Auditing and Reporting Module Objects

ARM objects are available in the [ICIBackupArchiveInfo Interface](#), the [ICIServer Interface](#), and the [Enum Reference](#). These objects deal with the management of the ARM database; the ARM-related objects used in [Event Rules](#) are not discussed below.

In the ICIServer Interface:

- Property [ARMAuthenticationType](#) As ARMAuthenticationType
- Property [ARMDatabaseName](#) As String
- Property [ARMDatabaseType](#) As ARMDatabaseType
- Property [ARMFailureAuditFolder](#) As String
- Property [ARMFailureEmailList](#) As String
- Property [ARMHealConnection](#) As Boolean
- Property [ARMNotifyOnDisconnect](#) As Boolean
- Property [ARMNotifyOnReconnect](#) As Boolean
- Property [ARMPassword](#) As String
- Function [ARMReconnect\(\)](#) As Boolean
- Property [ARMReconnectPeriod](#) As DWord
- Property [ARMServerName](#) As String
- Property [ARMStopAuditingOnError](#) As Boolean
- Function [ARMTTestConnection\(\)](#) As Boolean
- Property [ARMUserName](#) As String
- Property [EnableARM](#) As Boolean

In the Enum Reference:

- [ARMAuthenticationType](#) (v6.3 and later)
- [ARMDatabaseType](#) (v6.3 and later)
- [ServerModule](#) (v6.1 and later)

AS2-RelatedObjects

This chapter contains AS2-related COM objects.

AS2 Objects

In addition to the [ICIAS2InboundSettings Interface](#), [ICIAS2OutboundSettings Interface](#), [ICIAS2SendActionParams Interface](#), and [ICIAS2TransactionInfo Interface](#), the following AS2 settings are in the [ICISite Interface](#), [ICIClientSettings Interface](#), and [Enum Reference](#).

For SFTPAdvBool, used below:

- abFalse = 0
- abInherited = -2
- abTrue = 1

In the [ICISite](#) interface:

- Property **AS2AccountManagementPath** As String (read only)

Retrieves URL for AS2 account management.

```
HRESULT AS2AccountManagementPath([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 management path is: " &  
oSite.AS2AccountManagementPath
```

EFT v6.3 and later

- Property **AS2Certificate** As String

Retrieves or specifies AS2 certificate path.

```
HRESULT AS2Certificate([out, retval] BSTR* pVal);  
HRESULT AS2Certificate([in] BSTR newVal);
```

Retrieving:

```
MsgBox "AS2 certificate is: " & oSite.AS2Certificate
```

Specifying:

```
oSite.AS2Certificate = "C:\MyCerts\AS2Cert.crt"
```

EFT v6.3 and later

- Property **AS2Identifier** As String

Retrieves or specifies the AS2 identifier.

```
HRESULT AS2Identifier([out, retval] BSTR* pVal);
HRESULT AS2Identifier([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 identifier is: " & oSite.AS2Identifier
```

Specifying:

```
oSite.AS2Identifier = "my_as2_id"
```

EFT v6.3 and later

- Property **AS2Passphrase** As String

Retrieves or specifies the AS2 passphrase.

```
HRESULT AS2Passphrase([out, retval] BSTR* pVal);
HRESULT AS2Passphrase([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 passphrase is: " & oSite.AS2Passphrase
```

Specifying:

```
oSite.AS2Passphrase = "new_passphrase"
```

EFT v6.3 and later

- Property **AS2PrivateKey** As String

Retrieves or specifies the AS2 private key.

```
HRESULT AS2PrivateKey([out, retval] BSTR* pVal);
HRESULT AS2PrivateKey([in] BSTR newVal);
```


Example (VBScript):

Retrieving:

```
MsgBox "AS2 certificate is: " & oSite.AS2PrivateKey
```

Specifying:

```
oSite.AS2PrivateKey = "C:\MyCerts\AS2Cert.crt"
```

EFT v6.3 and later

- Property **EnableAS2** As Boolean

Retrieves or specifies whether AS2 is enabled on the Site.

```
HRESULT EnableAS2([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableAS2([in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

Example (VBScript):

Retrieving:

```
if False = oSite.EnableAS2 then
MsgBox "Disable"
else
MsgBox "Remove"
end if
```

Specifying:

```
oSite.EnableAS2 = True
```

EFT v6.3 and later

- Function **GetAS2Transactions**(vbShowSuccesses As Boolean, vbShowFailures As Boolean, vbShowInProgress As Boolean, bstrMessageIDFilter As String, bstrFileNameFilter As String, ILastDays As Long) As Variant

Retrieves AS2 transactions.

```
HRESULT GetAS2Transactions([in] VARIANT_BOOL
vbShowSuccesses,
[in] VARIANT_BOOL vbShowFailures, [in] VARIANT_BOOL
```

```
vbShowInProgress,
[in] BSTR bstrMessageIDFilter,[in] BSTR bstrFileNameFilter,
[in] LONG lLastDays,[out,retval] VARIANT* pTransInfo);
```

Example (VBScript):

```
For Each trans In oSite.GetAS2Transactions(True, True, True,
"\"", "\"", 7)
MsgBox "Transaction id: " & trans.TransactionID
Next
```

EFT v6.3 and later

In the ICIClientSettings interface:

- Property **AS2InboundSettings** As CIAS2InboundSettings

Retrieves user account's AS2 inbound settings.

```
HRESULT AS2InboundSettings(
[out,retval] IDispatch** pSttg);
[out,retval] ICIAS2InboundSettings** pSttg);
```

Example (VBScript):

```
Set oAS2InboundSettings = oClientSettings.AS2InboundSettings
```

EFT v6.3 and later

- Property **AS2OutboundSettings** As CIAS2OutboundSettings

Retrieves user account's AS2 outbound settings.

```
HRESULT AS2OutboundSettings(
[out,retval] IDispatch** pSttg);
```

Example (VBScript):

```
Set oAS2OutboundSettings =
oClientSettings.AS2OutboundSettings
```

EFT v6.3 and later

- Function **AS2TestOutboundProfile()** As AS2OutboundProfileStatus

Tests user account's AS2 outbound profile and returns status.

```
HRESULT AS2TestOutboundProfile(
    [out, retval] AS2OutboundProfileStatus* pVal);
```

Example (VBScript):

```
if AS2StatusOk = oClientSettings.AS2TestOutboundProfile then
    MsgBox "Success."
else
    MsgBox "Some going wrong."
end if
```

EFT v6.3 and later

- Function **GetAS2Inbound**([pInherited]) As Boolean

Retrieve whether AS2 inbound transactions are enabled for the user or Settings Template.

```
HRESULT GetAS2Inbound(
    [out, optional] VARIANT* pInherited,
    [out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
bEnabled = oClientSettings.GetAS2Inbound(bInherited)
MsgBox "Enabled: " & CStr(bEnabled) & " Inherited: " & CStr(
    bInherited)
```

EFT v6.3 and later

- Function **GetAS2Outbound**([pInherited]) As Boolean

Retrieve whether AS2 outbound transactions are enabled for the user or Settings Template.

```
HRESULT GetAS2Outbound(
    [out, optional] VARIANT* pInherited,
    [out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
bEnabled = oClientSettings.GetAS2Outbound(bInherited)
MsgBox "Enabled: " & CStr(bEnabled) & " Inherited: " & CStr
```

```
(bInherited)
```

EFT v6.3 and later

- Sub **SetAS2Inbound**(val As SFTPAdvBool)

Specify whether AS2 Inbound is enabled for the user or Settings Template.

```
HRESULT SetAS2Inbound(  
[in] SFTPAdvBool val);
```

Example (VBScript):

```
oClientSettings.SetAS2Inbound abInherited
```

EFT v6.3 and later

- Sub **SetAS2Outbound**(val As SFTPAdvBool)

Specify whether AS2 Outbound is enabled for the user or Settings Template.

```
HRESULT SetAS2Outbound(  
[in] SFTPAdvBool val);
```

Example (VBScript):

```
oClientSettings.SetAS2Outbound abTrue
```

EFT v6.3 and later

In the [Enum Reference](#):

- **AS2AuthenticationType** (v6.3 and later)

Used in the ICIAAS2InboundSettings [AuthenticationType](#) property used to specify the AS2 authentication mechanism.

- AS2AuthTypeUsernameAndPassword = 0
- AS2AuthTypeMsgLevelSecurity = 1

- **AS2ContentType** (v6.1 and later)

Used to specify the type of the document to send in the [ICIAS2OutboundSettings](#) and [ICIAS2SendActionParams](#) **ContentType** properties.

- X12 = 0
- EDIConsent = 3
- EDIFACT = 1

- Binary = 4
- XML = 2
- Plaintext = 5
- **AS2DoubleContentPolicy** (v6.3 and later)

Used in the ICIAAS2InboundSettings [DoubleContentPolicy](#) property to specify the AS2 duplicate contents policy.

 - AS2DoubleContentPolicyOverwrite = 0
 - AS2DoubleContentPolicyNumerate = 1
 - AS2DoubleContentPolicyReject = 2
- **AS2DoubleMsgIDPolicy** (v6.3 and later)

Used in the ICIAAS2InboundSettings [DoubleMsgIDPolicy](#) policy used to specify the AS2 duplicate message ID policy.

 - AS2DoubleMsgIDPolicyAccept = 0
 - AS2DoubleMsgIDPolicyReject = 1
- **AS2EncryptionAlgorithm** (v7.4.9 and later)

Used in the ICIAAS2InboundSettings [EncryptionAlgorithm](#) property to specify allowed encryption algorithms.

 - Const Encryption_3DES = 0
 - Const Encryption_DES = 1
 - Const Encryption_AESCBC128 = 2
 - Const Encryption_AESCBC192 = 3
 - Const Encryption_AESCBC256 = 4
- **AS2EncryptMissingPolicy** (v6.3 and later)

Used in the ICIAAS2InboundSettings [EncryptMissingPolicy](#) property.

 - AS2EncryptMissingPolicyAccept = 0
 - AS2EncryptMissingPolicyReject = 1
- **AS2IDMismatchPolicy** (v6.3 and later)

Used in the ICIAAS2InboundSettings [EFTIDMismatchPolicy](#) and [PartnerIDMismatchPolicy](#) properties to specify the mismatch policy.

 - AS2IDMismatchPolicyAccept = 0
 - AS2IDMismatchPolicyReject = 1
- **AS2OutboundProfileStatus** (v6.3 and later)

Used in the ICIClientSettings [AS2TestOutboundProfile](#) method to retrieve the result for AS2 Test Outbound Profile operation.

- AS2StatusOk = 0
- AS2StatusVerifyMDNFailed = 4
- AS2StatusNotRegistered = 1
- AS2StatusDatabaseFailure = 5
- AS2StatusCreateRequestFailed = 2
- AS2StatusUnexpectedError = 6
- AS2StatusSendDataFailed = 3

- **AS2ProxyType** (v6.3 and later)

Used in the ICIAS2OutboundSettings [ProxyType](#) property to specify the AS2 proxy type.

- AS2ProxyDMZ = 0
- AS2ProxyHTTP = 1

- **AS2ReceiptDelivery** (v6.1 and later)

Used to specify the receipt delivery mechanism in the [ICIAS2OutboundSettings](#) and [ICIAS2SendActionParams](#) ReceiptDelivery property.

- Sync = 0
- Async = 1

- **AS2ReceiptPolicy** (v6.1 and later)

Used to specify the type of the receipt required in the [ICIAS2OutboundSettings](#) and [ICIAS2SendActionParams](#) ReceiptPolicy property.

- NoReceipt = 0
- UnsignedReceipt = 1
- SignedReceipt = 2

- **AS2SignatureAlgorithm** (v7.4.9 and later)

The AS2SignatureAlgorithm enum is used in the ICIAS2OutboundSettings [SignatureAlgorithm](#) property.

- Signature_MD5 = 1
- Signature_SHA1 = 0
- Signature_SHA_224 = 5
- Signature_SHA_256 = 2

- Signature_SHA_384 = 3
- Signature_SHA_512 = 4

- **AS2SignatureMissingPolicy** (v6.3 and later)

Used in the [ICIAS2InboundSettings](#) **SignatureMissingPolicy** property to specify the AS2 signature-missing policy.

- AS2SignatureMissingPolicyAccept = 0
- AS2SignatureMissingPolicyReject = 1

- **AS2TransactionDirection** (v6.3 and later)

Used in the [ICIAS2TransactionInfo Interface](#) **Direction** property to retrieve the direction of an AS2 transaction.

- AS2TransDirectionInbound = 0
- AS2TransDirectionOutbound = 1

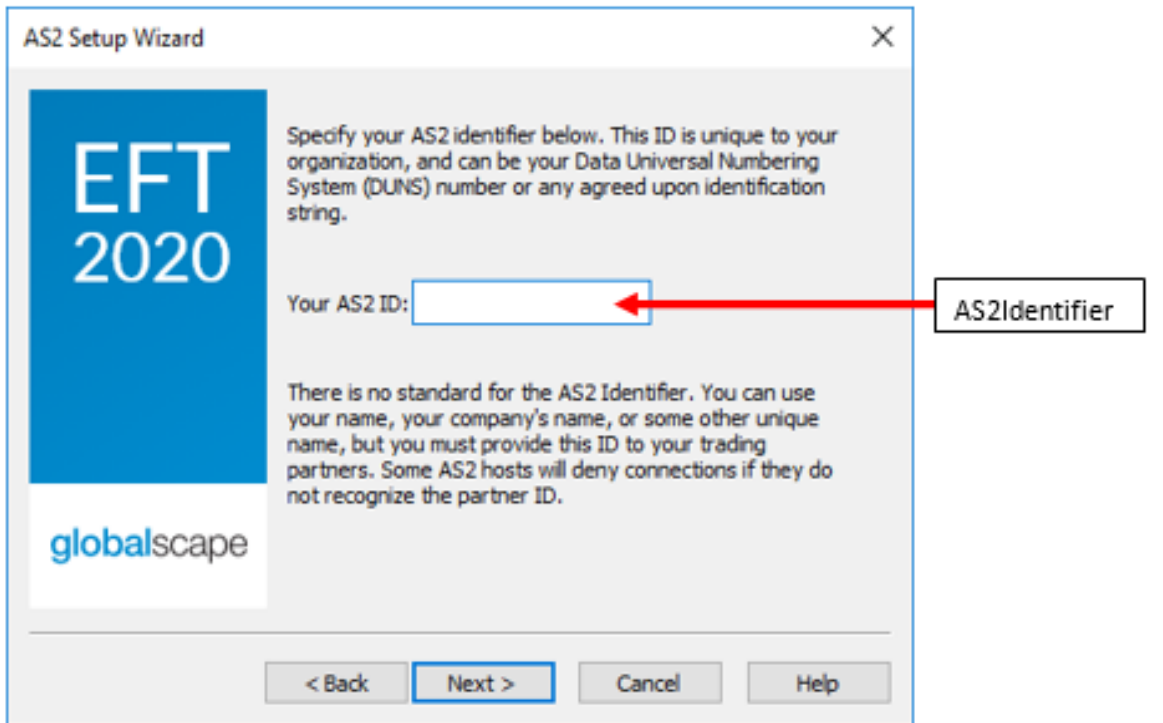
- **AS2TransactionStatus** (v6.3 and later)

Used in the [ICIAS2TransactionInfo Interface](#) **Status** property to retrieve the status of an AS2 transaction.

- AS2TransactionInProgress = 0
- AS2TransactionSuccess = 1
- AS2TransactionFailedCanResend = 2
- AS2TransactionFailedNoResend = 3
- AS2TransactionOkSendMDNFailed = 4

Dialog Box Equivalents

Site AS2 Settings



The screenshot shows the 'AS2 Setup Wizard' dialog box. On the left is a blue panel with 'EFT 2020' and the 'globalscape' logo. The main area has a title bar with a close button. The text reads: 'Specify your AS2 identifier below. This ID is unique to your organization, and can be your Data Universal Numbering System (DUNS) number or any agreed upon identification string.' Below this is a text input field labeled 'Your AS2 ID:'. A red arrow points from a box labeled 'AS2Identifier' to this field. Further down, a paragraph explains: 'There is no standard for the AS2 Identifier. You can use your name, your company's name, or some other unique name, but you must provide this ID to your trading partners. Some AS2 hosts will deny connections if they do not recognize the partner ID.' At the bottom are four buttons: '< Back', 'Next >', 'Cancel', and 'Help'.

AS2 Setup Wizard

EFT 2020

globalscape

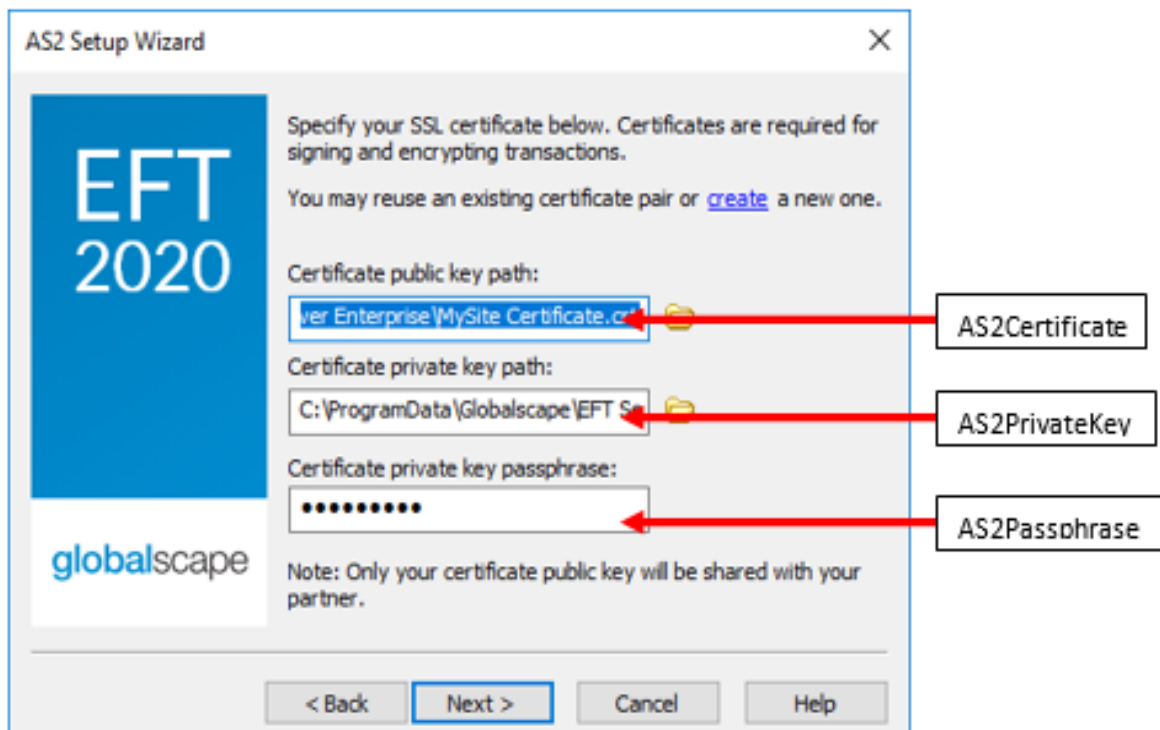
Specify your AS2 identifier below. This ID is unique to your organization, and can be your Data Universal Numbering System (DUNS) number or any agreed upon identification string.

Your AS2 ID:

AS2Identifier

There is no standard for the AS2 Identifier. You can use your name, your company's name, or some other unique name, but you must provide this ID to your trading partners. Some AS2 hosts will deny connections if they do not recognize the partner ID.

< Back Next > Cancel Help



The screenshot shows the 'AS2 Setup Wizard' dialog box for SSL certificate configuration. The left panel is identical to the previous dialog. The main area text reads: 'Specify your SSL certificate below. Certificates are required for signing and encrypting transactions. You may reuse an existing certificate pair or [create](#) a new one.' Below this are three fields: 'Certificate public key path:' with a file explorer icon, 'Certificate private key path:' with a file explorer icon, and 'Certificate private key passphrase:' with a masked input field. Red arrows point from boxes labeled 'AS2Certificate', 'AS2PrivateKey', and 'AS2Passphrase' to these respective fields. A note at the bottom states: 'Note: Only your certificate public key will be shared with your partner.' The bottom buttons are '< Back', 'Next >', 'Cancel', and 'Help'.

AS2 Setup Wizard

EFT 2020

globalscape

Specify your SSL certificate below. Certificates are required for signing and encrypting transactions. You may reuse an existing certificate pair or [create](#) a new one.

Certificate public key path:

AS2Certificate

Certificate private key path:

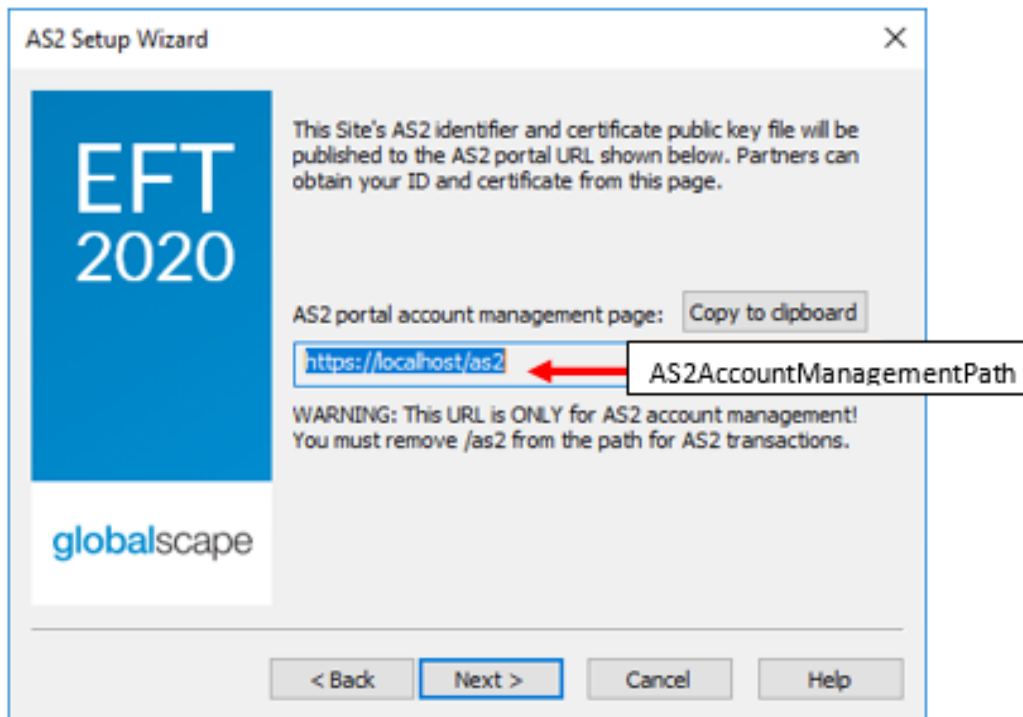
AS2PrivateKey

Certificate private key passphrase:

AS2Passphrase

Note: Only your certificate public key will be shared with your partner.

< Back Next > Cancel Help



ICIAS2InboundSettings Interface

The ICIAS2InboundSettings interface is used to define the AS2 inbound (retrieve) transaction settings for a partner account. See also [ICIAS2OutboundSettings Interface](#), [ICIAS2SendActionParams Interface](#), and [ICIAS2TransactionInfo Interface](#).

```
interface ICIAS2InboundSettings : IDispatch
```

See also:

- [Difference Between VARIANT_BOOL and SFTPAAdvBool](#)
- [ICIAS2OutboundSettings](#)
- [ICIAS2SendActionParams](#)
- [ICIAS2TransactionInfo](#)
- [ICIClientSettings::AS2InboundSettings](#) property
- [Enumerations Reference](#)

NOTE: "Sub" indicates that the member is a method; "Property" indicates that it is a property.

Members

- Property **AS2ID** As String

Retrieves or specifies a unique AS2-To ID that EFT will accept from this partner's inbound messages.

```
HRESULT AS2ID([out, retval] BSTR* pVal);
HRESULT AS2ID([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "My AS2 ID is: " & oAS2I.AS2ID
```

Specifying:

```
oAS2I.AS2ID = "my_as2_id"
```

EFT v6.3 and later

- Property **AuthenticationType** As [AS2AuthenticationType](#)

Retrieves or specifies whether the client is authenticating with username and password or using Message Level Security (MLS).

```
HRESULT AuthenticationType([out, retval]
AS2AuthenticationType* pVal);
HRESULT AuthenticationType([in] AS2AuthenticationType
newVal);
```

Example (VBScript):

Retrieving:

```
if AS2AuthTypeUsernameAndPassword = oAS2I.AuthenticationType
then
MsgBox "Username and password"
end if
```

Specifying:

```
oAS2I.AuthenticationType = AS2AuthTypeMsgLevelSecurity
```

EFT v6.3 and later

- Property **DoubleContentPolicy** As [AS2DoubleContentPolicy](#)

Retrieves or specifies overwrite, numerate, or reject AS2 transactions where the file is a duplicate of an existing file already residing in the upload directory.

```
HRESULT DoubleContentPolicy([out, retval]
AS2DoubleContentPolicy* pVal);
HRESULT DoubleContentPolicy([in] AS2DoubleContentPolicy
newVal);
```

Example (VBScript):

Retrieving:

```
if AS2DoubleContentPolicyOverwrite =
oAS2I.DoubleContentPolicy then
MsgBox "Overwrite"
end if
```

Specifying:

```
oAS2I.DoubleContentPolicy = AS2DoubleContentPolicyReject
```

EFT v6.3 and later

- Property **DoubleMsgIDPolicy** As [AS2DoubleMsgIDPolicy](#)

Retrieves or specifies whether to accept or reject AS2 transactions with duplicate Message IDs.

If an incoming request has the same Message ID and the same payload as a prior incoming message, EFT sends the exact same response from that prior transaction back to the client, and does not extract or process the payload (that is, discards the payload, does not save to file system, does not trigger Event Rules, etc.).

If an incoming request has the same Message ID, but the payload is different, EFT processes or discards the message based on the configuration provided by the user.

```
HRESULT DoubleMsgIDPolicy([out, retval]
AS2DoubleMsgIDPolicy* pVal);
HRESULT DoubleMsgIDPolicy([in] AS2DoubleMsgIDPolicy newVal);
```

Example (VBScript):

Retrieving:

```
if AS2DoubleMsgIDPolicyAccept = oAS2I.DoubleMsgIDPolicy then
MsgBox "Accept"
end if
```

Specifying:

```
oAS2I.DoubleMsgIDPolicy = AS2DoubleMsgIDPolicyReject
```

EFT v6.3 and later

- Property **EncryptionAlgorithm** As String

Retrieves or specifies encryption algorithm to use for inbound transfers. Refer to the [Enumerations Reference](#) for allowed algorithms.

EFT v7.4.9 and later

- Property **EFTIDMismatchPolicy** As [AS2IDMismatchPolicy](#)

Retrieves or specifies accept or reject AS2 transactions where the connecting partner's AS2-To ID does not match EFT AS2 identifier defined for that partner in that partner's AS2 Inbound tab.

```
HRESULT EFTIDMismatchPolicy([out, retval]
AS2IDMismatchPolicy* pVal);
HRESULT EFTIDMismatchPolicy([in] AS2IDMismatchPolicy
newVal);
```

Example (VBScript):**Retrieving:**

```
if AS2IDMismatchPolicyAccept = oAS2I.EFTIDMismatchPolicy
then
MsgBox "Accept."
end if
```

Specifying:

```
oAS2I.EFTIDMismatchPolicy = AS2IDMismatchPolicyReject
```

EFT v6.3 and later

- Property **EncryptMissingPolicy** As [AS2EncryptMissingPolicy](#)

Retrieves or specifies whether EFT will Accept or Reject the transaction if it is not encrypted.

```
HRESULT EncryptMissingPolicy([out, retval]
AS2EncryptMissingPolicy* pVal);
HRESULT EncryptMissingPolicy([in] AS2EncryptMissingPolicy
newVal);
```

Example (VBScript):**Retrieving:**

```
if AS2EncryptMissingPolicyAccept =
oAS2I.EncryptMissingPolicy then
MsgBox "accept"
end if
```

Specifying:

```
oAS2I.EncryptMissingPolicy = AS2EncryptMissingPolicyReject
```

EFT v6.3 and later

- Function **GetFailureCommand()** As Object

Retrieve [ICICCommandActionParams interface](#), which specified a custom command to occur upon failure.

```
HRESULT GetFailureCommand([out, retval] IDispatch**
ppdispCommandParams);
```

Example (VBScript):

```
Set oCmd = oAS2O.GetFailureCommand()
```

EFT v6.3 and later

- Function **GetFailureEmail()** As Object

Retrieves the [ICIMailActionParams interface](#), which specified e-mail notification parameters that will be send after failure transaction happened.

```
HRESULT GetFailureEmail([out, retval] IDispatch**
ppdispEmailParams);
```

Example (VBScript):

```
Set oMailParams = oAS2O.GetFailureEmail()
```

EFT v6.3 and later

- Function **GetSuccessCommand()** As Object

Retrieve [ICICommandActionParams](#) interface, which specified a custom command to occur upon successful transaction.

```
HRESULT GetSuccessCommand([out, retval] IDispatch**
ppdispCommandParams);
```

Example (VBScript):

```
Set oCmd = oAS2O.GetSuccessCommand()
```

EFT v6.3 and later

- Function **GetSuccessEmail()** As Object

Retrieve the [ICIMailActionParams](#) interface, which specified e-mail notification parameters that will be sent after successful transaction happened.

```
HRESULT GetSuccessEmail([out, retval] IDispatch**
ppdispEmailParams);
```

Example (VBScript):

```
Set oMailParams = oAS2O.GetSuccessEmail()
```

EFT v6.3 and later

- Property **MDNAsyncSendTimeoutSec** As Long

Retrieves or specifies the timeout period after which an asynchronous MDN send attempt shall be considered a failure if no response is received from the remote server. Range: 0-600, 60 seconds is the default. 0 = no timeout.

```
HRESULT MDNAsyncSendTimeoutSec([out, retval] LONG* pVal);
HRESULT MDNAsyncSendTimeoutSec([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "MDN send timeout is: " & CStr
(oAS2I.MDNSendTimeoutSec)
```

Specifying:

```
oAS2I.MDNSendTimeoutSec = 50
```

EFT v6.3 and later

- Property **MDNSendRetriesCount** As Long

(In v6.3.0, it was "MDNSendAttemptRetries")

Retrieves or specifies number of time to attempt to send MDN. Range: 0 (no retry) to 999; 10 is the default.

```
HRESULT MDNSendRetriesCount([out, retval] LONG* pVal);
HRESULT MDNSendRetriesCount([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "MDN send retries is: " & CStr
(oAS2I.MDNSendRetriesCount)
```

Specifying:

```
oAS2I.MDNSendRetriesCount = 40
```

EFT v6.3 and later

- Property **MDNSendRetryDelaySec** As Long

Retrieves or specifies retry interval. Range: 0 (no wait) to - 600 seconds; 30 seconds is the default.

```
HRESULT MDNSendRetryDelaySec([out, retval] LONG* pVal);
HRESULT MDNSendRetryDelaySec([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "MDN retry delays is: " & CStr
(oAS2I.MDNSendRetryDelaySec)
```

Specifying:

```
oAS2I.MDNSendRetryDelaySec = 120
```

EFT v6.3.1 and later

- Property **MICAlgorithm** As [AS2MICHash](#)

```
HRESULT MICAlgorithm([out, retval] AS2MICHash* pVal);
HRESULT MICAlgorithm([in] AS2MICHash newVal);
```

EFT v7.4.11 and later

- Property **MoveReceivedDataToFolder** As String

Retrieves or specifies a folder in which to save received data.

```
HRESULT MoveReceivedDataToFolder([out, retval] BSTR* pVal);
HRESULT MoveReceivedDataToFolder([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Destination folder is: " &
oAS2I.MoveReceivedDataToFolder
```

Specifying:

```
oAS2I.MoveReceivedDataToFolder = "C:\DestFolder"
```

EFT v6.3 and later

- Property **PartnerCertificatePath** As String

Retrieves or specifies the AS2 certificate to be used for verifying signed messages.

```
HRESULT PartnerCertificatePath([out, retval] BSTR* pVal);
HRESULT PartnerCertificatePath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 partner certificate path: " & CStr
(oAS2I.PartnerCertificatePath)
```

Specifying:

```
oAS2I.PartnerCertificatePath = "C:\Certs\Ivan.crt"
```

EFT v6.3 and later

- Property **PartnerAS2ID** As String

Retrieves or specifies your AS2 partner id. Used to identify the AS2-From ID for inbound messages emanating from partners.

```
HRESULT PartnerAS2ID([out, retval] BSTR* pVal);
```

```
HRESULT PartnerAS2ID([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "My AS2 partner ID is: " & oAS2I.PartnerAS2ID
```

Specifying:

```
oAS2I.PartnerAS2ID = "my_as2_partner_id"
```

EFT v6.3 and later

- Property **PartnerIDMismatchPolicy** As [AS2IDMismatchPolicy](#)

Retrieves or specifies accept or reject AS2 transactions where the connecting partner's AS2-From ID does not match the AS2 identifier defined for that partner.

```
HRESULT PartnerIDMismatchPolicy([out, retval]
AS2IDMismatchPolicy* pVal);
HRESULT PartnerIDMismatchPolicy([in] AS2IDMismatchPolicy
newVal);
```

Example (VBScript):

Retrieving:

```
if AS2IDMismatchPolicyAccept = oAS2I.PartnerIDMismatchPolicy
then
MsgBox "Accept."
end if
```

Specifying:

```
oAS2I.PartnerIDMismatchPolicy = AS2IDMismatchPolicyReject
```

EFT v6.3 and later

- Property **RenameFilesTo** As String

Retrieves or specifies string that appends filename to path using specified variables. Preserve filename if undefined.

```
HRESULT RenameFilesTo([out, retval] BSTR* pVal);
HRESULT RenameFilesTo([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Rename files to: " & oAS2I.RenameFilesTo
```

Specifying:

```
oAS2I.RenameFilesTo = ""
```

EFT v6.3 and later

- Sub **SetFailureCommand**(pdispCommandParams As Object)

Specify [ICICommandActionParams interface](#), which specified a custom command to occur upon failed send.

```
HRESULT SetFailureCommand(IDispatch* pdispCommandParams);
```

Example (VBScript):

```
oAS2O.SetFailureCommand oCmd
```

EFT v6.3 and later

- Sub **SetFailureEmail**(psidpEmailParams As Object)

Use the [ICIMailActionParams interface](#) to specified e-mail notification parameters that will be sent after failure transaction happened.

```
HRESULT SetFailureEmail(IDispatch* pdispEmailParams);
```

Example (VBScript):

```
oAS2O.SetFailureEmail oMailParams
```

EFT v6.3 and later

- Sub **SetSuccessCommand**(psidpCommandParams As Object)

Specify the [IC ICommandActionParams](#) interface, which specified a custom command to occur upon successful transaction.

```
HRESULT SetSuccessCommand(IDispatch* pdispCommandParams);
```

Example (VBScript):

```
oAS2O.SetSuccessCommand oCmd
```

EFT v6.3 and later

- Sub **SetSuccessEmail**(psidpEmailParams As Object)

Use the [ICIMailActionParams interface](#) to specify e-mail notification parameters that will be send after successful transaction happened.

```
HRESULT SetSuccessEmail(IDispatch* pdispEmailParams);
```

Example (VBScript):

```
oAS2O.SetSuccessEmail oMailParams
```

EFT v6.3 and later

- Property **SignatureAlgorithm** As String

Used to retrieve or specify the allowed signature algorithm for inbound AS2 transfers.

EFT v7.4.9 and later

- Property **SignatureMissingPolicy** As [AS2SignatureMissingPolicy](#)

Retrieves or specifies whether EFT will accept or reject the transaction if it is not signed.

```
HRESULT SignatureMissingPolicy([out, retval]
AS2SignatureMissingPolicy* pVal);
HRESULT SignatureMissingPolicy([in]
AS2SignatureMissingPolicy newVal);
```

Example (VBScript):

Retrieving:

```
if AS2SignatureMissingPolicyAccept =
oAS2I.SignatureMissingPolicy then
MsgBox "accept"
end if
```

Specifying:

```
oAS2I.SignatureMissingPolicy =
AS2SignatureMissingPolicyReject
```

EFT v6.3 and later

- Property **SkipHTTPContinue** As Boolean

Retrieves or specifies "continue" option. "100 Continue" is a part of the HTTP protocol and means "Everything is OK for now, continue this transaction." Many AS2 servers use it to avoid timeouts. It is the client software's responsibility to process this reply properly (just skip until 200 OK or real HTTP error is received). If the client does not process the reply correctly, you can turn off the "100 Continue" response. If your AS2 partner receives multiple MDN failures when sending to EFT in HTTPS, turn off the "100 Continue" reply.

```
HRESULT SkipHTTPContinue([out, retval] VARIANT_BOOL* pVal);
HRESULT SkipHTTPContinue([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

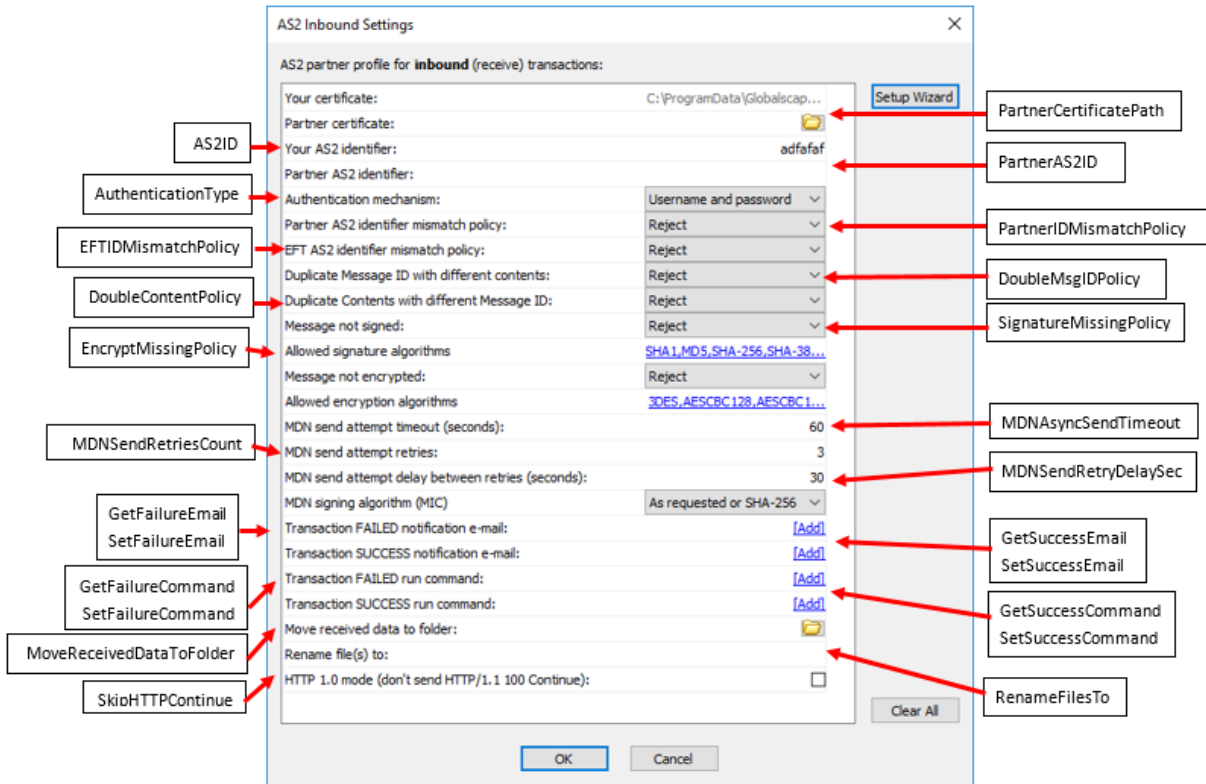
```
MsgBox "Skip continue: " & CStr(oAS2I.SkipHTTPContinue)
```

Specifying:

```
oAS2I.SkipHTTPContinue = False
```

EFT v6.3 and later

Dialog Box Equivalents



ICIAS2OutboundSettings Interface

The ICIAS2OutboundSettings interface is used to define the AS2 outbound (send) transaction settings for a partner account.

```
interface ICIAS2OutboundSettings : IDispatch
```

See also:

- [Difference Between VARIANT_BOOL and SFTPAdvBool](#)
- [ICIAS2InboundSettings - AS2 Inbound Settings Interface](#),
- [ICIAS2SendActionParams - AS2 Send Action Parameters Interface](#)
- [ICIAS2TransactionInfo - AS2 Transaction Information Interface](#)
- [ICIClientSettings::AS2OutboundSettings](#) property

NOTE: "Sub" indicates that the member is a method; "Property" indicates that it is a property.

Members

- Property **AS2ID** As String

Specifies or retrieves the AS2 ID of the file sender.

```
HRESULT AS2ID([out, retval] BSTR* pVal);
HRESULT AS2ID([in] BSTR newVal);
```

Example (VBScript)

Retrieving:

```
MsgBox "My AS2 ID is: " & oAS2O.AS2ID
```

Specifying:

```
oAS2O.AS2ID = "my_as2_id"
```

EFT v6.3 and later

- Property **AsyncReceiptTimeoutMin** As Long

Specifies or retrieves the maximum wait time (in minutes) for asynchronous AS2 receipt. This property is applicable only if asynchronous receipt is requested (i.e., [ReceiptPolicy](#) property is not set to **NoReceipt** and [ReceiptDelivery](#) property is set to **Async**).

```
HRESULT AsyncReceiptTimeoutMin([out, retval] LONG* pVal);
HRESULT AsyncReceiptTimeoutMin([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 asynchronous receipt timeout: " & CStr
(oAS2O.AsyncReceiptTimeoutMin)
```

Specifying:

```
oAS2O.AsyncReceiptTimeoutMin = 6000
```

EFT v6.3 and later

- Property **Compress** As Boolean

Specifies or retrieves whether an AS2 file will be compressed before sending.

```
HRESULT Compress([out, retval] VARIANT_BOOL* pVal);
HRESULT Compress([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 compress message: " & CStr(oAS2O.Compress)
```

Specifying:

```
oAS2O.Compress = False
```

EFT v6.3 and later

- Property **ContentType** As [AS2ContentType](#)

Specifies or retrieves content type of AS2 message to send.

```
HRESULT ContentType([out,retval] AS2ContentType* pVal);
HRESULT ContentType([in] AS2ContentType newVal);
```

Example (VBScript):**Retrieving:**

```
if XML = oAS2O.ContentType then
MsgBox "AS2 content type: XML"
end if
```

Specifying:

```
oAS2O.ContentType = Plaintext
```

EFT v6.3 and later

- Property **CustomHeaders** As String

```
HRESULT CustomHeaders([out, retval] BSTR* pVal);
HRESULT CustomHeaders([in] BSTR newVal);
```

EFT v7.4.11 and later

- Property **DeleteSourceFile** As Boolean

Specifies or retrieves whether source file will be removed after successful delivery to AS2 partner.

```
HRESULT DeleteSourceFile([out, retval] VARIANT_BOOL* pVal);
HRESULT DeleteSourceFile([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 delete source file: " & oAS2O.DeleteSourceFile
```

Specifying:

```
oAS2O.DeleteSourceFile = False
```

EFT v6.3 and later

- Property **Encrypt As Boolean**

Specifies or retrieves whether an AS2 message will be encrypted with the AS2 partner certificate.

```
HRESULT Encrypt([out, retval] VARIANT_BOOL* pVal);
HRESULT Encrypt([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 encrypt message: " & CStr(oAS2O.Encrypt)
```

Specifying:

```
oAS2O.Encrypt = True
```

EFT v6.3 and later

- Property **EncryptionAlgorithm As [AS2EncryptionAlgorithm](#)**

Used to specify the encryption algorithm for outbound AS2 transfers.

EFT v7.4.9 and later.

- Property **FileExcludeMask As String**

Specifies or retrieves which of the files must NOT be sent using the file mask.


```
HRESULT FileExcludeMask([out, retval] BSTR* pVal);
HRESULT FileExcludeMask([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 exclude mask is: " & oAS2O.FileExcludeMask
```

Specifying:

```
oAS2O.FileExcludeMask = "*.*)"
```

EFT v6.3 and later

- Property **FileIncludeMask** As String

Specifies or retrieves which of the files must be sent using the file mask.

```
HRESULT FileIncludeMask([out, retval] BSTR* pVal);
HRESULT FileIncludeMask([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 include mask is: " & oAS2O.FileIncludeMask
```

Specifying:

```
oAS2O.FileIncludeMask = "*.*)"
```

EFT v6.3 and later

- Function **GetFailureCommand()** As Object

Retrieve [ICICCommandActionParams interface](#), which specified the custom command to occur upon failure.

```
HRESULT GetFailureCommand([out, retval] IDispatch**
ppdispCommandParams);
```

Example (VBScript):

```
Set oCmd = oAS2O.GetFailureCommand()
```

EFT v6.3 and later

- Function **GetFailureEmail()** As Object

Retrieves the [ICIMailActionParams interface](#), which specified e-mail notification parameters that will be sent after the failed transaction happens.

```
HRESULT GetFailureEmail([out, retval] IDispatch**
  ppdispEmailParams);
```

Example (VBScript):

```
Set oMailParams = oAS2O.GetFailureEmail()
```

EFT v6.3 and later

- Function **GetSuccessCommand()** As Object

Retrieves [ICICommandActionParams interface](#), which specified a custom command to occur upon successful transaction.

```
HRESULT GetSuccessCommand([out, retval] IDispatch**
  ppdispCommandParams);
```

Example (VBScript):

```
Set oCmd = oAS2O.GetSuccessCommand()
```

EFT v6.3 and later

- Function **GetSuccessEmail()** As Object

Retrieves the [ICIMailActionParams interface](#), which specified e-mail notification parameters that will be sent after successful transaction happened.

```
HRESULT GetSuccessEmail([out, retval] IDispatch**
  ppdispEmailParams);
```

Example (VBScript):

```
Set oMailParams = oAS2O.GetSuccessEmail()
```

EFT v6.3 and later

- Property **Host** As String

Specifies or retrieves the remote host to which you are sending an AS2 file, and the protocol prefix, http:// or https://. If no protocol prefix is specified, plain HTTP protocol will be used.

```
HRESULT Host([out, retval] BSTR* pVal);
HRESULT Host([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 destination host: " & oAS2O.Host
```

Specifying:

```
oAS2O.Host = "192.168.0.88"
```

EFT v6.3 and later

- Property **MailboxPath** As String

(EFT v6.3.0.57 only; see [RemotePath](#).)

Retrieves or specifies a destination directory path.

```
HRESULT MailboxPath([out, retval] BSTR* pVal);
HRESULT MailboxPath([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 destination path: " & oAS2O.MailboxPath
```

Specifying:

```
oAS2O.MailboxPath = "/"
```

EFT v6.3.0.57 only (see [RemotePath](#))

- Property **MessageSendAttemptRetries** As Long

(EFT v6.3.0.57 only; Removed in EFT v6.3.1; See [Retries Count](#).)

Retrieves or specifies the number of message send attempts retries.

```
HRESULT MessageSendAttemptRetries([out, retval] LONG* pVal);
HRESULT MessageSendAttemptRetries([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 retries count is: " & CStr  
(oAS2O.MessageSendAttemptRetries)
```

Specifying:

```
oAS2O.MessageSendAttemptRetries = 80
```

EFT v6.3.0.57 only; Removed in EFT v6.3.1

- Property **MessageSendAttemptTimeout** As Long

(EFT v6.3.0.57 only; Removed in EFT v6.3.1; See [SendAttemptDelayBetweenRetries](#))

```
HRESULT MessageSendAttemptTimeout([out, retval] LONG* pVal);  
HRESULT MessageSendAttemptTimeout([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 Timeout is: " & CStr  
(oAS2O.MessageSendAttemptTimeout)
```

Specifying:

```
oAS2O.MessageSendAttemptTimeout = 10
```

EFT v6.3.0.57 only; Removed in EFT v6.3.1

- Property **MonitoredHotFolder** As String

Specifies or retrieves the monitored hot folder path. Newly created/added files will be sent in the AS2 message.

```
HRESULT MonitoredHotFolder([out, retval] BSTR* pVal);  
HRESULT MonitoredHotFolder([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 hot folder: " & oAS2O.MonitoredHotFolder
```

Specifying:

```
oAS2O.MonitoredHotFolder = "C:\HotFolder "
```

EFT v6.3 and later

- Property **PartnerAS2ID** As String

Specifies or retrieves the AS2 ID of the AS2 partner to which you are sending an AS2 file.

```
HRESULT PartnerAS2ID([out, retval] BSTR* pVal);
HRESULT PartnerAS2ID([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Her AS2 ID is: " & oAS2O.PartnerAS2ID
```

Specifying:

```
oAS2O.PartnerAS2ID = "her_as2_id"
```

EFT v6.3 and later

- Property **PartnerCertificatePath** As String

Specifies or retrieves the path to the AS2 partner certificate for AS2 message encrypting.

```
HRESULT PartnerCertificate([out, retval] BSTR* pVal);
HRESULT PartnerCertificatePath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 partner certificate path: " & CStr
(oAS2O.PartnerCertificatePath)
```

Specifying:

```
oAS2O.PartnerCertificatePath = "C:\Certs\Ivan.crt"
```

EFT v6.3 and later

- Property **Password** As String

Specifies or retrieves the password used for authentication on the AS2 Server to which you are sending an AS2 file.

```
HRESULT Password([out, retval] BSTR* pVal);
HRESULT Password([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 password: " & oAS2O.Password
```

Specifying:

```
oAS2O.Password = "swordfish"
```

EFT v6.3 and later

- Property **Port** As Long

Specifies or retrieves the remote port.

```
HRESULT Port([out, retval] LONG* pVal);
HRESULT Port([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 destination port: " & oAS2O.Port
```

Specifying:

```
oAS2O.ort = 80
```

EFT v6.3 and later

- Property **ProxyHostName** As String

Specifies or retrieves the proxy host.

```
HRESULT ProxyHostName([out, retval] BSTR* pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy hostname is: " & oAS2O.ProxyHostName
```

Specifying:

```
oAS2O.ProxyHostName = "192.168.0.89"
```

EFT v6.3 and later

- Property **ProxyPassword** As String

Specifies or retrieves the proxy password.

```
HRESULT ProxyPassword([out, retval] BSTR* pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy password is: " & oAS2O.ProxyPassword
```

Specifying:

```
oAS2O.ProxyPassword = "swordfish"
```

EFT v6.3 and later

- Property **ProxyPort** As Long

Specifies or retrieves the proxy port.

```
HRESULT ProxyPort([out, retval] LONG* pVal);
HRESULT ProxyPort([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy port is: " & CStr(oAS2O.ProxyPort)
```

Specifying:

```
oAS2O.ProxyPort = 8080
```

EFT v6.3 and later

- Property **ProxyType** As [AS2ProxyType](#)

Specifies or retrieves the proxy type.

```
HRESULT ProxyType([out, retval] AS2ProxyType* pVal);
HRESULT ProxyType([in] AS2ProxyType newVal);
```

Example (VBScript):**Retrieving:**

```
if AS2ProxyHTTP = oAS2O.ProxyType then
MsgBox "AS2 using HTTP proxy"
end if
```

Specifying:

```
oAS2O.ProxyType = AS2ProxyDMZ
```

EFT v6.3 and later

- Property **ProxyUserName** As String

Specifies or retrieves the proxy username.

```
HRESULT ProxyUserName([out, retval] BSTR* pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy user name is: " & oAS2O.ProxyUserName
```

Specifying:

```
oAS2O.ProxyUserName = "User_1"
```

EFT v6.3 and later

- Property **ReceiptDelivery** As [AS2ReceiptDelivery](#)

Specifies or retrieves the mechanism of receipt delivery required for an AS2 message.

This property is applicable only if [ReceiptPolicy](#) property is not set to **NoReceipt**.

```
HRESULT ReceiptDelivery([out, retval] AS2ReceiptDelivery*
pVal);
HRESULT ReceiptDelivery([in] AS2ReceiptDelivery newVal);
```


Example (VBScript):**Retrieving:**

```
if Sync = oAS2O.ReceiptDelivery then
MsgBox "Synchronous"
end if
```

Specifying:

```
oAS2O.ReceiptDelivery = Async
```

EFT v6.3 and later

- Property **ReceiptPolicy** As [AS2ReceiptPolicy](#)

Specifies or retrieves the type of the receipt required for an AS2 message.

```
HRESULT ReceiptPolicy([out, retval] AS2ReceiptPolicy* pVal);
HRESULT ReceiptPolicy([in] AS2ReceiptPolicy newVal);
```

Example (VBScript):**Retrieving:**

```
if NoReceipt = oAS2O.ReceiptPolicy then
MsgBox "Do not request a receipt"
end if
```

Specifying:

```
oAS2O.ReceiptPolicy = UnsignedReceipt
```

EFT v6.3 and later

- Property **RemotePath** As String

Retrieve/specify a destination directory path.

```
HRESULT RemotePath([out, retval] BSTR* pVal);
HRESULT RemotePath([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 destination path: " & oAS2O.RemotePath
```

Specifying:

```
oAS2O.RemotePath = "/"
```

EFT v6.3.1 and later

- Property **RetriesCount** As Long

Specifies or retrieves the remote path to which you are sending an AS2 file.

```
HRESULT RetriesCount([out, retval] LONG* pVal);
HRESULT RetriesCount([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 retries count is: " & CStr(oAS2O.RetriesCount)
```

Specifying:

```
oAS2O.RetriesCount = 80
```

EFT v6.3.1 and later

- Property **RetryDelaySec** As Long

Specifies or retrieves the interval between successive sending retries (in seconds) when a send fails. This property is applicable only if [RetriesCount](#) property is not set to 0.

```
HRESULT RetryDelaySec([out, retval] LONG* pVal);
HRESULT RetryDelaySec([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 delay between retries: " & CStr
(oAS2O.RetryDelaySec)
```

Specifying:

```
oAS2O.RetryDelaySec = 60
```

EFT v6.3.1 and later

- Property **SendAttemptDelayBetweenRetries** As Long

Retrieves or specifies the delay between retries, in seconds.

```
HRESULT SendAttemptDelayBetweenRetries([out, retval] LONG*
pVal);
HRESULT SendAttemptDelayBetweenRetries([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 delay between retries: " & CStr
(oAS20.SendAttemptDelayBetweenRetries)
```

Specifying:

```
oAS20.SendAttemptDelayBetweenRetries = 60
```

EFT v6.3.0

- Sub **SetFailureCommand**(pdispCommandParams As Object)

Specifies [ICICCommandActionParams](#) interface, which specified a custom command to occur upon failed.

```
HRESULT SetFailureCommand(IDispatch* pdispCommandParams);
```

Example (VBScript):

```
oAS20.SetFailureCommand oCmd
```

EFT v6.3 and later

- Sub **SetFailureEmail**(pdispEmailParamsAs Object)

Specifies the [ICIMailActionParams](#) interface for e-mail notification parameters that will be sent after a failed transaction.

```
HRESULT SetFailureEmail(IDispatch* pdispEmailParams);
```

Example (VBScript):

```
oAS20.SetFailureEmail oMailParams
```

EFT v6.3 and later

- Sub **SetSuccessCommand**(pdispCommandParams As Object)

Specifies the [ICICommandActionParams](#) interface, which specified a custom command to occur upon successful transaction.

```
HRESULT SetSuccessCommand(IDispatch* pdispCommandParams);
```

Example (VBScript):

```
oAS20.SetSuccessCommand oCmd
```

EFT v6.3 and later

- Sub **SetSuccessEmail**(pdispEmailParams As Object)

Specifies the [ICIMailActionParams](#) interface for e-mail notification parameters that will be sent after a successful transaction.

```
HRESULT SetSuccessEmail(IDispatch* pdispEmailParams);
```

Example (VBScript):

```
oAS20.SetSuccessEmail oMailParams
```

EFT v6.3 and later

- Property **Sign** As Boolean

Specifies or retrieves whether an AS2 message will be signed with the certificate specified in the Site's AS2 Settings.

```
HRESULT Sign([out, retval] VARIANT_BOOL* pVal);
HRESULT Sign([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 sign message: " & CStr(oAS20.Sign)
```

Specifying:

```
oAS20.Sign = True
```

EFT v6.3 and later

- Property **SignatureAlgorithm** As [AS2SignatureAlgorithm](#)

Used to specify the signature algorithm for outbound AS2 transfers.

EFT v7.4.9 and later.

- Property **Subject** As String

Specifies or retrieves the AS2 message subject.

```
HRESULT Subject([out, retval] BSTR* pVal);
HRESULT Subject([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 message subject: " & oAS20.Subject
```

Specifying:

```
oAS20.Subject = "Some subject "
```

EFT v6.3 and later

- Property **TimeoutSec** As Long

Specifies or retrieves the timeout (in seconds) while sending a file.

```
HRESULT TimeoutSec([out, retval] LONG* pVal);
HRESULT TimeoutSec([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 Timeout is: " & CStr(oAS20.TimeoutSec)
```

Specifying:

```
oAS20.TimeoutSec = 10
```

EFT v6.3.1 and later

- Property **UseProxy** As Boolean

Specifies or retrieves whether the proxy will be used when sending a file.

```
HRESULT UseProxy([out, retval] VARIANT_BOOL* pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "AS2 using proxy: " & CStr(oAS2O.UseProxy)
```

Specifying:

```
oAS2O.UseProxy = False
```

EFT v6.3 and later

- **Property User As String**

Specifies or retrieves the username for authentication on the AS2 Server to which you are sending an AS2 file.

```
HRESULT User([out, retval] BSTR* pVal);  
HRESULT Username([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

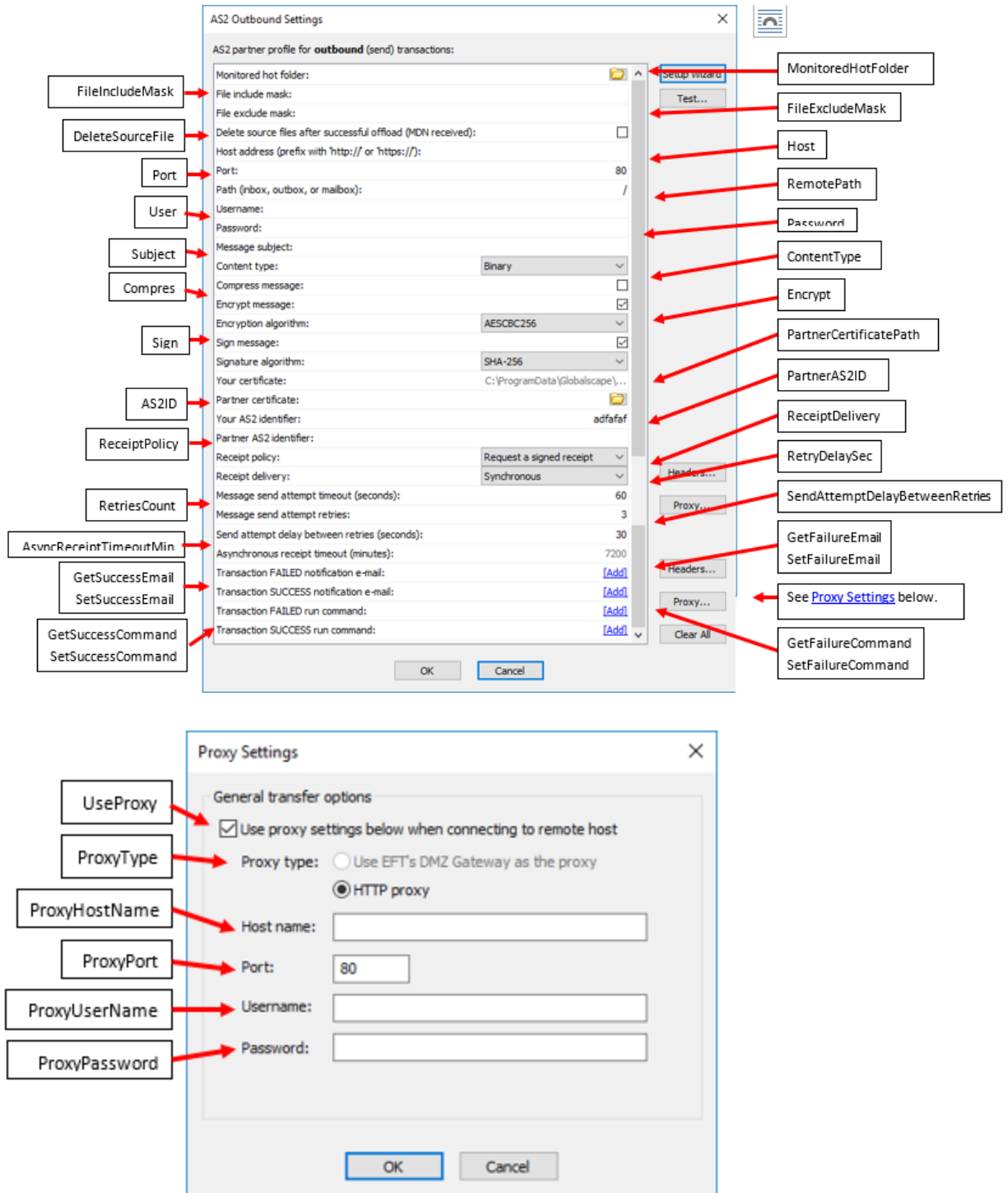
```
MsgBox "AS2 user: " & oAS2O.User
```

Specifying:

```
oAS2O.User = "User_1"
```

EFT v6.3.1 and later

Dialog Box Equivalents



ICIAS2SendActionParams Interface

The ICias2SendActionParams interface allows you to create, modify, and retrieve AS2 Send File Event Rule Actions. ([Dialog box equivalents](#) appear below the table.) See also [ICias2InboundSettings - AS2 Inbound Settings Interface](#), [ICias2OutboundSettings - AS2 Outbound Settings Interface](#), and [ICias2TransactionInfo - AS2 Transaction Information Interface](#).

```
interface ICias2SendActionParams : IDispatch
```

Example (VBScript):

```
If ActionStatement.Action.Type = AS2SendAction Then
Set AS2SendParams = ActionStatement.Action.Params
` Modify AS2 Send File parameters and assign them back to the
action
ActionStatement.Action.Params = AS2SendParams
End If
```

The **CIAS2SendActionParams** co-class supports creation of the **Send AS2 File Event Action**. The co-class implements the ICias2SendActionParams interface.

Example (VBScript):

```
Set AS2Params = CreateObject
("SFTPCOMInterface.CIAS2SendActionParams")
` Setup Send AS2 File parameters
` . . .
Rule.AddActionStatement(0, AS2Params)
```

Members

- Property **AS2ID** As String

Specifies or retrieves the AS2 ID of the file sender.

```
HRESULT AS2ID([out, retval] BSTR* pVal);
HRESULT AS2ID([in] BSTR newVal);
```

Examples:

Retrieving:

```
MsgBox "Send message as " & AS2SendParams.AS2ID & " AS2
identifier"
```

Specifying:


```
AS2SendParams.AS2ID = "EFT : User1"
```

EFT v6.1 and later

- Property **AsyncReceiptTimeoutMin** As Long

This property is applicable only if asynchronous receipt is requested (i.e., [ReceiptPolicy](#) property is not set to NoReceipt and [ReceiptDelivery](#) property is set to Async).

Specifies or retrieves the maximum wait time (in minutes) for asynchronous AS2 receipt.

```
HRESULT AsyncReceiptTimeoutMin([out, retval] long* pVal);
HRESULT AsyncReceiptTimeoutMin([in] long newVal);
```

Example (VBScript):

Retrieving:

```
If AS2SendParams.AsyncReceiptTimeoutMin < 60 Then
MsgBox 'Asynchronous receipt waiting period is too short'
End If
```

Specifying:

```
AS2SendParams.AsyncReceiptTimeoutMin = 7200 '5 days
```

EFT v6.1 and later

- Property **ClientProfile** As String

Specifies or retrieves the username of AS2 partner whose AS2 Outbound Profile will be used for this AS2 Send operation.

```
HRESULT ClientProfile([out, retval] BSTR* pVal);
HRESULT ClientProfile([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
If AS2SendParams.UseClientProfile Then
MsgBox AS2SendParams.ClientProfile & "'s AS2 profile will be used"
End If
```

Specifying:

```
AS2SendParams.UseClientProfile = True
AS2SendParams.ClientProfile = "Some_User"
```

EFT v6.1 and later

- Property **Compress** As Boolean

Specifies or retrieves whether an AS2 file will be compressed before sending.

```
HRESULT Compress([out, retval] VARIANT_BOOL* pVal);
HRESULT Compress([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
If Not AS2SendParams.Compress Then
MsgBox "Sending large files without compression is time-
consuming!"
End If
```

Specifying:

```
AS2SendParams.Compress = True
```

EFT v6.1 and later

- Property **ContentType** As [AS2ContentType](#)

Specifies or retrieves content type of AS2 message to send.

```
HRESULT ContentType([out, retval] AS2ContentType* pVal);
HRESULT ContentType([in] AS2ContentType newVal);
```

Example (VBScript):**Retrieving:**

```
If AS2SendParams.ContentType = Binary Then
MsgBox "File type: Binary"
End If
```

Specifying:

```
AS2SendParams.ContentType = Plaintext
```

EFT v6.1 and later

- Property **CustomHeaders** As String

Used to specify a custom header for outbound AS2 transfers.

EFT v7.4.9

- Property **DeleteSourceFile** As Boolean

Specifies or retrieves whether source file will be removed after successful delivery to AS2 partner.

```
HRESULT DeleteSourceFile([out, retval] VARIANT_BOOL* pVal);
HRESULT DeleteSourceFile([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If AS2SendParams.DeleteSourceFile Then
MsgBox "Warning! Source file will be removed after
delivery!"
End If
```

Specifying:

```
AS2SendParams.DeleteSourceFile = True
```

EFT v6.1 and later

- Property **Encrypt** As Boolean

Specifies or retrieves whether an AS2 message will be encrypted with the AS2 partner certificate.

```
HRESULT Encrypt([out, retval] VARIANT_BOOL* pVal);
HRESULT Encrypt([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If Not AS2SendParams.Encrypt Then
MsgBox "Sending data without encryption is not secure!"
End If
```

Specifying:

```
AS2SendParams.Encrypt = True
```

EFT v6.1 and later

- Property **EncryptionAlgorithm** As String

Retrieves or specifies encryption algorithm to use for AS2 transfers. Refer to the [Enumerations Reference](#) for allowed algorithms.

EFT v7.4.9 and later

- Property **Host** As String

Specifies or retrieves the remote host to which you are sending an AS2 file, and the protocol prefix, http:// or https://. If no protocol prefix is specified, plain HTTP protocol will be used.

```
HRESULT Host([out, retval] BSTR* pVal);
HRESULT Host([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "The file will be sent to " & AS2SendParams.Host
```

Specifying:

```
AS2SendParams.Host = "https://some_as2_server.com"
```

EFT v6.1 and later

- Property **LocalPath** As String

Specifies or retrieves local path of the file to send.

```
HRESULT LocalPath([out, retval] BSTR* pVal);
HRESULT LocalPath([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox AS2SendParams.LocalPath & " file will be sent"
```

Specifying:

```
AS2SendParams.LocalPath = "%FS.PATH%"
```

EFT v6.1 and later

- Property **PartnerAS2ID** As String

Specifies or retrieves the AS2 ID of the AS2 partner to which you are sending an AS2 file.

```
HRESULT PartnerAS2ID([out, retval] BSTR* pVal);
HRESULT PartnerAS2ID([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Send message to " & AS2SendParams.PartnerAS2ID & " partner"
```

Specifying:

```
AS2SendParams.PartnerAS2ID = "John Smith"
```

EFT v6.1 and later

- Property **PartnerCertificatePath** As String

Specifies or retrieves the path to the AS2 partner certificate for AS2 message encrypting.

```
HRESULT PartnerCertificatePath([out, retval] BSTR* pVal);
HRESULT PartnerCertificatePath([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Encrypt message with " & AS2SendParams.PartnerCertificatePath
```

Specifying:

```
AS2SendParams.PartnerCertificatePath = "C:\Partner.pem"
```

EFT v6.1 and later

- Property **Password** As String

Specifies or retrieves the password used for authentication on the AS2 Server to which you are sending an AS2 file.

```
HRESULT Password([out, retval] BSTR* pVal);
HRESULT Password([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
sHTTPCredentials = AS2SendParams.User & ":" &
AS2SendParams.Password
```

Specifying:

```
AS2SendParams.Password = "Password1"
```

EFT v6.1 and later

- Property **Port** As Long

Specifies or retrieves the remote port.

```
HRESULT Port([out, retval] long* pVal);
HRESULT Port([in] long newVal);
```

Example (VBScript):**Retrieving:**

```
If (AS2SendParams.Port <> 80) And (AS2SendParams.Port <>
443) Then
MsgBox "Are you sure you want to use non-standard HTTP
port?"
End If
```

Specifying:

```
AS2SendParams.Port = 443
```

EFT v6.1 and later

- Property **PortAsString** As String

```
HRESULT PortAsString([out, retval] BSTR *pVal);
HRESULT PortAsString([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ProxyHostName** As String

Used to specify the AS2 proxy host name.

EFT v7.4.9 and later

- Property **ProxyPassword** As String

Used to specify the AS2 proxy password.

EFT v7.4.9 and later

- Property **ProxyPort** As Long

Used to specify the AS2 proxy port.

EFT v7.4.9 and later

- Property **ProxyType** As [AS2ProxyType](#)

Used to specify the AS2 proxy type.

EFT v7.4.9 and later

- Property **ProxyUserName** As String

Used to specify the AS2 proxy user name.

EFT v7.4.9 and later

- Property **ReceiptDelivery** As [AS2ReceiptDelivery](#)

This property is applicable only if **ReceiptPolicy** property, below, is *not* set to NoReceipt.

Specifies or retrieves the mechanism of receipt delivery required for an AS2 message. (Refer to [AS2ReceiptDelivery](#) enumeration for available mechanisms.)

```
HRESULT ReceiptDelivery([out, retval] AS2ReceiptDelivery*
pVal);
HRESULT ReceiptDelivery([in] AS2ReceiptDelivery newVal);
```

Example (VBScript):**Retrieving:**

```
If (AS2SendParams.ReceiptPolicy <> NoReceipt)
And (AS2SendParams.ReceiptDelivery = Async)
Then
MsgBox "Are you ready to wait for receipt for several days?"
End If
```

Specifying:

```
AS2SendParams.ReceiptDelivery = Sync
```

EFT v6.1 and later

- Property **ReceiptPolicy** As [AS2ReceiptPolicy](#)

Specifies or retrieves the type of the receipt required for AS2 message. (Refer to [AS2ReceiptPolicy](#) enumeration for available receipt types).

```
HRESULT ReceiptPolicy([out, retval] AS2ReceiptPolicy* pVal);
HRESULT ReceiptPolicy([in] AS2ReceiptPolicy newVal);
```

Example (VBScript):**Retrieving:**

```
If AS2SendParams.ReceiptPolicy <> SignedReceipt Then
MsgBox "Request signed receipt to authenticate AS2 partner!"
End If
```

Specifying:

```
AS2SendParams.ReceiptPolicy = SignedReceipt
```

EFT v6.1 and later

- Property **RemotePath** As String

Specifies or retrieves the remote path to which you are sending an AS2 file.


```
HRESULT RemotePath([out, retval] BSTR* pVal);
HRESULT RemotePath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
sURL = AS2Params.Host & ":" & CStr(AS2Params.Port) &
AS2Params.RemotePath
```

Specifying:

```
AS2SendParams.RemotePath = "/AS2.aspx"
```

EFT v6.1 and later

- Property **RetriesCount** As Long

Specifies or retrieves the maximum amount of sending retries in the case of failure (timeout, network errors, etc.); 0 means 'no retry if failed'.

```
HRESULT RetriesCount([out, retval] long* pVal);
HRESULT RetriesCount([in] long newVal);
```

Example (VBScript):

Retrieving:

```
If AS2SendParams.RetriesCount = 0 Then
MsgBox "Are you sure no retry is needed for sending
failures?"
End If
```

Specifying:

```
AS2SendParams.RetriesCount = 5 'Try another 5 times if fails
to send
```

EFT v6.1 and later

- Property **RetryDelaySec** As Long

Specifies or retrieves the interval between successive sending retries (in seconds) in the case of sending failure. This property is applicable only if **RetriesCount** property is not set to 0.

```
HRESULT RetryDelaySec([out, retval] long* pVal);
HRESULT RetryDelaySec([in] long newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Will wait for " & AS2Params.RetryDelaySec " seconds
before resend"
```

Specifying:

```
AS2SendParams.RetryDelaySec = 60 `1 minute
```

EFT v6.1 and later

- Property **Sign As Boolean**

Specifies or retrieves whether an AS2 message will be signed with the certificate specified in the Site's AS2 settings.

```
HRESULT Sign([out, retval] VARIANT_BOOL* pVal);
HRESULT Sign([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
If Not AS2SendParams.Sign Then
MsgBox "Unsigned message will not allow partner to
authenticate you!"
End If
```

Specifying:

```
AS2SendParams.Sign = True
```

EFT v6.1 and later

- Property **SignatureAlgorithm As [AS2SignatureAlgorithm](#)**

```
HRESULT SignatureAlgorithm([out, retval]
AS2SignatureAlgorithm* pVal);
HRESULT SignatureAlgorithm([in] AS2SignatureAlgorithm
newVal);
```

EFT v7.4.11 and later

- Property **Subject** As String

Specifies or retrieves the AS2 message subject.

```
HRESULT Subject([out, retval] BSTR* pVal);
HRESULT Subject([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Message Subject: " & AS2SendParams.Subject
```

Specifying:

```
AS2SendParams.Subject = "Requested client list attached"
```

EFT v6.1 and later

- Property **TimeoutSec** As Long

Specifies or retrieves the timeout (in seconds) while sending the file.

```
HRESULT TimeoutSec([out, retval] long* pVal);
HRESULT TimeoutSec([in] long newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Send Timeout: " & CStr(AS2SendParams.TimeoutSec) & " seconds"
```

Specifying:

```
AS2SendParams.TimeoutSec = 900 ` 15 minutes
```

EFT v6.1 and later

- Property **UseClientProfile** As Boolean

Specifies or retrieves whether a whole AS2 Outbound Profile of certain EFT client will be used for this AS2 Send operation. Otherwise, you must explicitly specify each AS2 Send parameter. (Refer also to the [ClientProfile](#) property.)

```
HRESULT UseClientProfile([out, retval] VARIANT_BOOL* pVal);
HRESULT UseClientProfile([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If Not AS2SendParams.UseClientProfile Then
` Specify each AS2 Send parameter
End If
```

Specifying:

```
AS2SendParams.UseClientProfile = True
```

EFT v6.1 and later

- Property **UseProxy** As Boolean

Specify whether to use proxy settings when connecting to a remote host.

```
HRESULT UseProxy([out, retval] VARIANT_BOOL* pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

EFT v7.4.11 and later

- Property **User** As String

Specifies or retrieves the username for authentication on the AS2 Server to which you are sending an AS2 file.

```
HRESULT User([out, retval] BSTR* pVal);
HRESULT User([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

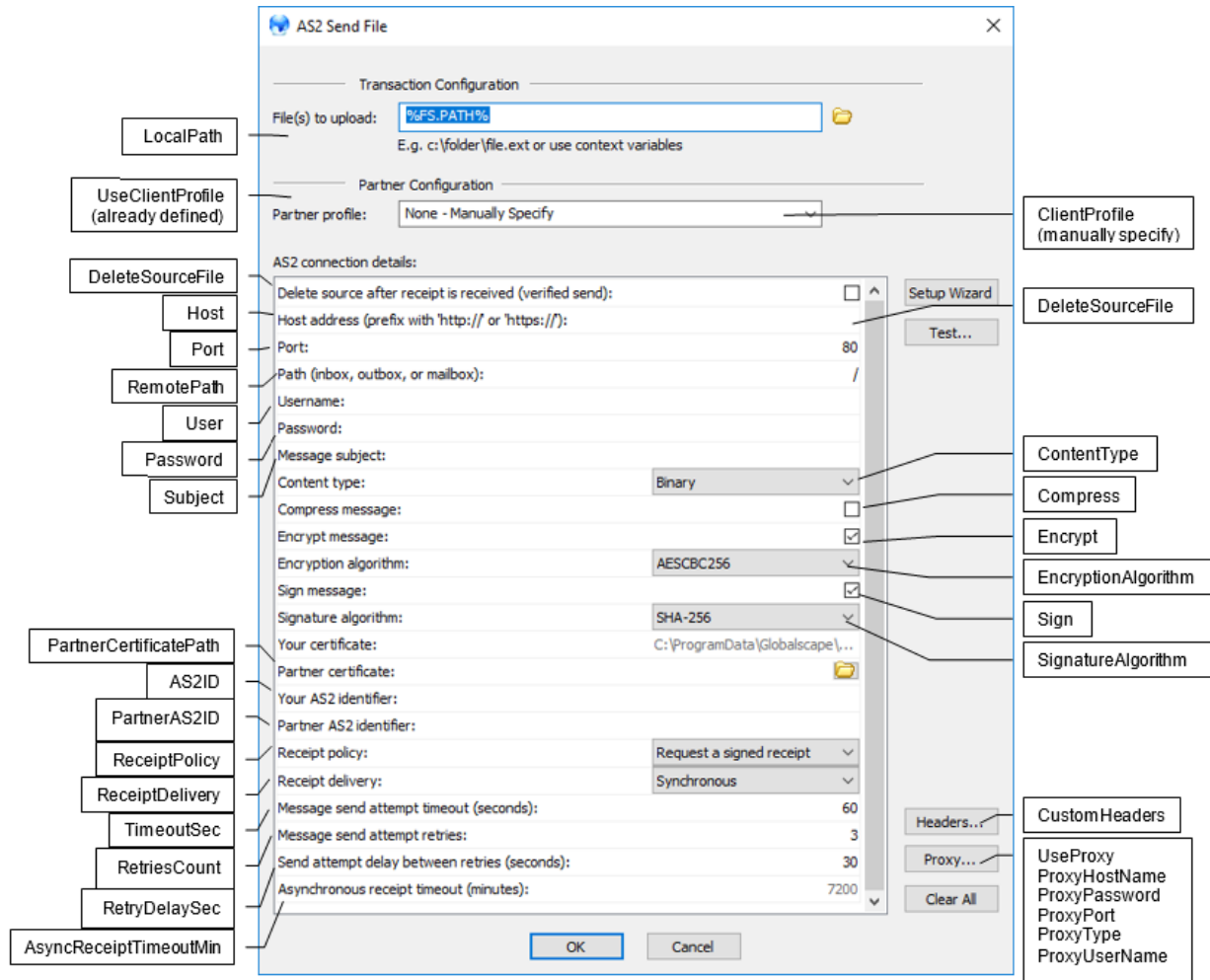
```
MsgBox "Will send the file using " & AS2SendParams.User & " account"
```

Specifying:

```
AS2SendParams.User = "User1"
```

EFT v6.1 and later

Dialog Box Equivalents



ICIAS2TransactionInfo Interface

The ICIAS2TransactionInfo interface is used to retrieve information about an AS2 transaction. See also [ICIAS2OutboundSettings - AS2 Outbound Settings Interface](#), [ICIAS2InboundSettings - AS2 Inbound Settings Interface](#), and [ICIAS2SendActionParams - AS2 Send Action Parameters Interface](#).

```
interface ICIAS2TransactionInfo : IDispatch {
```

Properties

- Property **Direction** As [AS2TransactionDirection](#) read-only
Retrieves the direction of a transaction.

```
HRESULT Direction([out, retval] AS2TransactionDirection* pVal);
```

AS2TransactionDirection:

AS2TransDirectionInbound = 0

AS2TransDirectionOutbound = 1

Example (VBScript):

```
if AS2TransDirectionOutBound = oAS2T.Direction then
MsgBox "Outbound."
end if
```

EFT v6.3 and later

- Property **File** As String read-only

Retrieves the name of file transferred.

```
HRESULT File([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 file name: " & oAS2T.File
```

EFT v6.3 and later

- Property **From** As String read-only

Retrieves the alias of the sender of the AS2 message.

```
HRESULT From([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 sender: " & oAS2T.From
```

EFT v6.3 and later

- Property **MessageID** As String read-only

Retrieves the message ID from an AS2 header.

```
HRESULT MessageID([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 message ID: " & oAS2T.MessageID
```

EFT v6.3 and later

- Property **Path** As String read-only

Retrieves the file path – local inbox or outbox.

```
HRESULT Path([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 file path: " & oAS2T.Path
```

EFT v6.3 and later

- Property **RemoteHost** As String read-only

Retrieves the host address of the sender (Inbound)/receiver (Outbound).

```
HRESULT RemoteHost([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 remote host address: " & oAS2T.RemoteHost
```

EFT v6.3 and later

- Property **Status** As [AS2TransactionStatus](#) read-only

Retrieves the status of an AS2 transaction.

```
HRESULT Status([out, retval] AS2TransactionStatus* pVal);
```

Example (VBScript):

```
Const AS2TransactionInProgress = 0
Const AS2TransactionSuccess = 1
Const AS2TransactionFailedCanResend = 2
Const AS2TransactionFailedNoResend = 3
Const AS2TransactionOkSendMDNFailed = 4

if AS2TransactionSuccess = oAS2T.Status then
MsgBox "AS2 transaction success."
end if
```

EFT v6.3 and later

- Property **Time** As Date read-only

Retrieves the date/time last recorded status for the transaction.

```
HRESULT Time([out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "AS2 Date/Time: " & CStr(oAS2T.Time)
```

EFT v6.3 and later

- Property **To** As String read-only

Retrieves the alias of the receiver of the AS2 message.

```
HRESULT To([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 receiver: " & oAS2T.To
```

EFT v6.3 and later

- Property **TransactionID** As String read-only

Retrieves the ID of an AS2 transaction.

```
HRESULT TransactionID([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 transaction ID: " & oAS2T.TransactionID
```

EFT v6.3 and later

Authentication-Related Objects

This chapter contains Authentication-related COM objects.

ICIADAuthManagerSettings Interface

Use the ICIADAuthManagerSettings interface to specify Active Directory authentication settings on a Site.

```
interface ICIADAuthManagerSettings : IDispatch {
```

VARIANT_BOOL equals VARIANT_TRUE(-1) when it is true and VARIANT_FALSE(0) when it is false.

See also:

- [Difference Between VARIANT_BOOL and SFTPAAdvBool](#)
- [ICISites - Multi-Site Management](#)
- To see a sample script, refer to [Script Examples](#).
- Scroll down to see the dialog box equivalents.

For an example using these properties, refer to the Windows Authentication script in [Script Examples](#).

Properties

- Property **AssignHomeFolderFromUserProfile** As Boolean

Retrieves or specifies whether to create a virtual folder using the user's home folder in the user's AD profile or to create a physical folder under the Site root folder using the user's username.

```
HRESULT AssignHomeFolderFromUserProfile([out, retval]
VARIANT_BOOL* pVal);
HRESULT AssignHomeFolderFromUserProfile([in] VARIANT_BOOL
newVal);
```

TRUE(-1) =Virtual folder; FALSE(0) =Physical folder

EFT v6.3 and later

- Property **CreateUsersOnlyByLogin** As Boolean

```
HRESULT CreateUsersOnlyByLogin([out, retval] VARIANT_BOOL*
pVal);
[propput, id(9), helpstring("property
```

```
CreateUsersOnlyByLogin"))]
HRESULT CreateUsersOnlyByLogin([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property **Domain** As String

Retrieves or specifies the domain name of the AD server.

```
HRESULT Domain([out, retval] BSTR* pVal);
HRESULT Domain([in] BSTR newVal);
```

EFT v6.3 and later

- Property **Group** As String

Retrieves or specifies the group to use for Windows authentication (Everyone or a specific group). Multiple groups can be specified, separated by commas.

```
HRESULT Group([out, retval] BSTR* pVal);
HRESULT Group([in] BSTR newVal);
```

EFT v6.3 and later

- Property **KeepHomeFoldersInSync** As Boolean

```
HRESULT KeepHomeFoldersInSync([out, retval] VARIANT_BOOL*
pVal);
HRESULT KeepHomeFoldersInSync([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property **LogonAttribute** As [ADAuthManagerLogonAttribute](#)

Retrieves or specifies which user attribute to use for the logon name.

```
HRESULT LogonAttribute([out, retval]
ADAuthManagerLogonAttribute* pVal);
HRESULT LogonAttribute([in] ADAuthManagerLogonAttribute
newVal);
```

EFT v6.3 and later

- Property **RefreshIntervalMinutes** As Long

Retrieves or specifies the number of minutes between refreshing the user database.

```
HRESULT RefreshIntervalMinutes([out, retval] LONG* pVal);
HRESULT RefreshIntervalMinutes([in] LONG newVal);
```

EFT v6.3 and later

- Property **SkipDomainPrefix** As Boolean

Retrieves or specifies whether to add the domain prefix to the folder name.

```
HRESULT SkipDomainPrefix([out,retval] VARIANT_BOOL* pVal);
HRESULT SkipDomainPrefix([in] VARIANT_BOOL newVal);
```

TRUE(-1) = Skip domain prefix; FALSE(0) =Do not skip domain prefix

Example (VBScript):

Retrieving:

```
MsgBox "Skip domain prefix: " & CStr(oADAM.SkipDomainPrefix)
```

Specifying:

```
oADAM.SkipDomainPrefix = False
```

EFT v6.3 and later

- Property **Type** As [ADAuthManagerType](#)

Retrieves or specifies whether to use an AD server or local system account for Windows authentication.

```
HRESULT Type([out, retval] ADAuthManagerType* pVal);
HRESULT Type([in] ADAuthManagerType newVal);
```

EFT v6.3 and later

- Property **UseWindowsAccessPermissions**

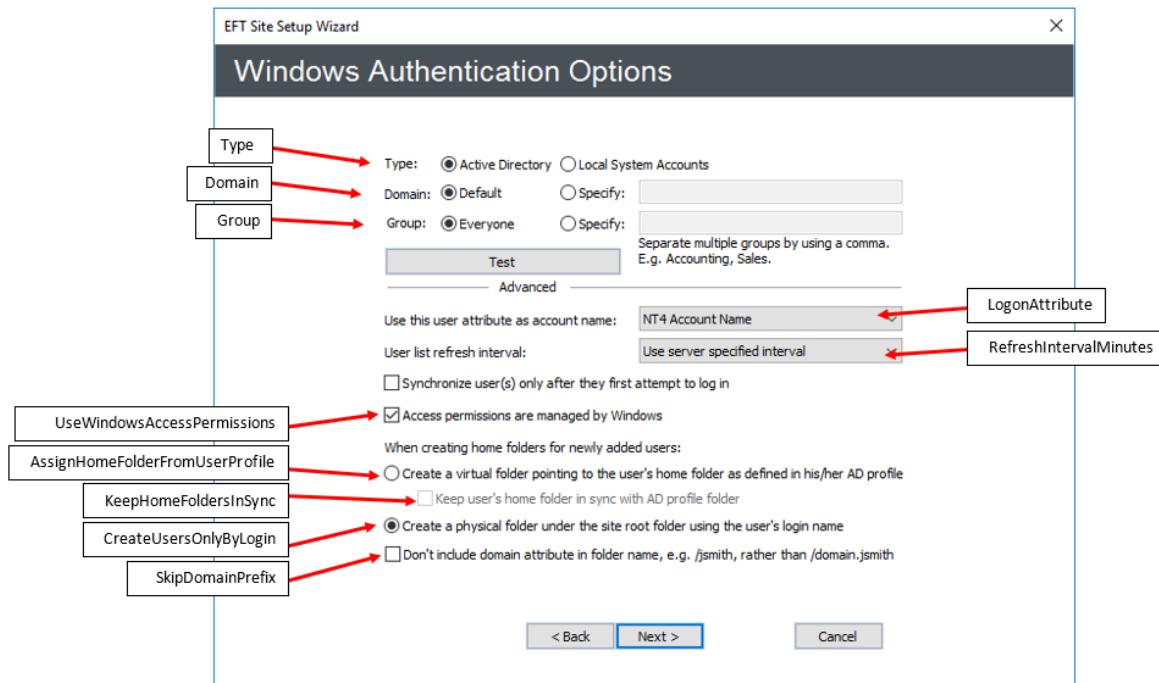
Retrieves or specifies whether to use Windows access permissions.

```
HRESULT UseWindowsAccessPermissions([out, retval] VARIANT_
BOOL* pVal);
HRESULT UseWindowsAccessPermissions([in] VARIANT_BOOL
newVal);
```

EFT v7.4.9 and later

Dialog Box Equivalents

The ICIADAuthManagerSettings interface properties correlate to the following fields and controls in the EFT administration interface:



ICIGSAuthManagerSettings Interface

Use the ICIGSAuthManagerSettings interface to edit Globalscape authentication manager settings. See also [ICISites - Multi-Site Management](#).

```
interface ICIGSAuthManagerSettings : IDispatch
```

Properties

- Property **RefreshIntervalMinutes** As Long

Retrieves or specifies the user database refresh frequency.

```
HRESULT RefreshIntervalMinutes([out, retval] LONG* pVal);
HRESULT RefreshIntervalMinutes([in] LONG newVal);
```

EFT v6.3 and later

- Property **UserDatabasePath** As String

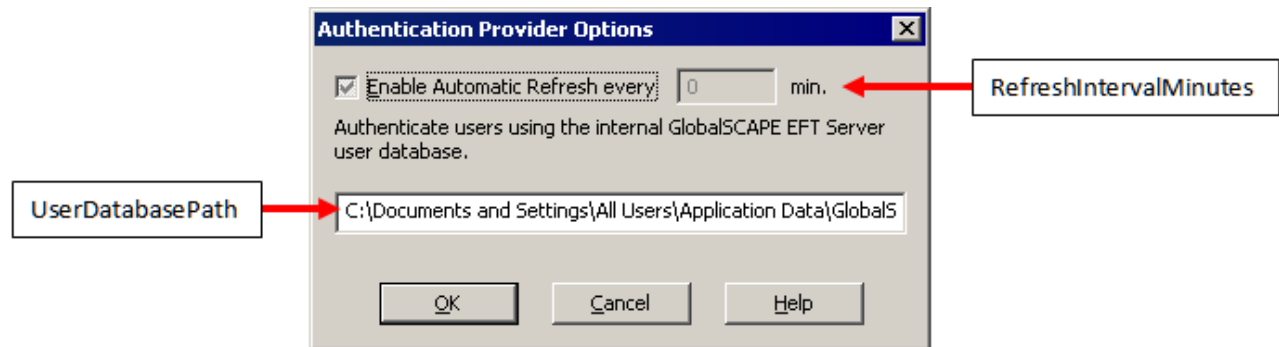
Retrieves or sets the path to the user database.

```
HRESULT UserDatabasePath([out, retval] BSTR* pVal);
HRESULT UserDatabasePath([in] BSTR newVal);
```

EFT v6.3 and later

Dialog Box Equivalents

The ICIGSAuthManagerSettings interface properties correlate to the following fields and controls in the EFT administration interface:



ICILDAPAuthManagerSettings Interface

The LDAP Authentication Manager Settings interface members are listed below. See also [ICISites - Multi-Site Management](#).

```
interface ICILDAPAuthManagerSettings : IDispatch
```

For an example using these properties, refer to the Windows Authentication script in [Script Examples](#).

Members

- Property **AdditionalUserSettingsPath** As String

Retrieves or specifies the path to the user database.

```
HRESULT AdditionalUserSettingsPath([out, retval] BSTR* pVal);
HRESULT AdditionalUserSettingsPath([in] BSTR newVal);
```

EFT v6.3 and later

- Property **AutoCreateSettingsTemplate** As String

Retrieves or specifies the path to the user database.

```
HRESULT AutoCreateSettingsTemplate([out, retval] VARIANT_
BOOL* pVal);
HRESULT AutoCreateSettingsTemplate([in] VARIANT_BOOL
newVal);
```

EFT v7.4.13 and later

- Property **BaseDN** As String

Retrieves or specifies the BaseDN.

```
HRESULT BaseDN([out, retval] BSTR* pVal);
HRESULT BaseDN([in] BSTR newVal);
```

EFT v6.3 and later

- Property **ChaseReferrals** As Boolean

Retrieves or specifies whether to enable chase referrals.

```
HRESULT ChaseReferrals([out, retval] VARIANT_BOOL* pVal);
HRESULT ChaseReferrals([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property **LDAPVersion** As [LDAPVersion](#)

Retrieves or specifies the LDAP version.

```
HRESULT LDAPVersion([out, retval] LDAPVersion* pVal);
HRESULT LDAPVersion([in] LDAPVersion newVal);
```

EFT v6.3 and later

- Property **LoginAttribute** As String

Retrieves or specifies the login attribute.

```
HRESULT LoginAttribute([out, retval] BSTR* pVal);
HRESULT LoginAttribute([in] BSTR newVal);
```

EFT v6.3 and later

- Property **PageSizeEntries** As Long

Retrieves or specifies the search page size.

```
HRESULT PageSizeEntries([out, retval] LONG* pVal);
HRESULT PageSizeEntries([in] LONG newVal);
```

EFT v6.3 and later

- Property **Port** As Long

Retrieves or specifies the LDAP server port.

```
HRESULT Port([out, retval] LONG* pVal);
HRESULT Port([in] LONG newVal);
```

EFT v6.3 and later

- Property **RefreshIntervalMinutes** As Long

Retrieves or specifies how often to refresh the user database.

```
HRESULT RefreshIntervalMinutes([out, retval] LONG* pVal);
HRESULT RefreshIntervalMinutes([in] LONG newVal);
```

EFT v6.3 and later

- Property **SearchAttributes** As String

Retrieves or specifies the search attributes.

```
HRESULT SearchAttributes([out, retval] BSTR* pVal);
HRESULT SearchAttributes([in] BSTR newVal);
```

EFT v6.3 and later

- Property **SearchScope** As [LDAPSearchScope](#)

Retrieves or specifies the search scope.

```
HRESULT SearchScope([out, retval] LDAPSearchScope* pVal);
HRESULT SearchScope([in] LDAPSearchScope newVal);
```

EFT v6.3 and later

- Property **Server** As String

Retrieves or specifies the LDAP server host name.

```
HRESULT Server([out, retval] BSTR* pVal);
HRESULT Server([in] BSTR newVal);
```

EFT v6.3 and later

- Property **ServerPassword** As String

Retrieves or specifies the password for Simple authentication.

```
HRESULT ServerPassword([out, retval] BSTR* pVal);  
HRESULT ServerPassword([in] BSTR newVal);
```

EFT v6.3 and later

- Property **ServerUserName** As String

Retrieves or specifies the username for Simple authentication.

```
HRESULT ServerUserName([out, retval] BSTR* pVal);  
HRESULT ServerUserName([in] BSTR newVal);
```

EFT v6.3 and later

- Property **SettingsTemplateAttributes** As String

Retrieves or specifies Settings Template attributes.

```
HRESULT SettingsTemplateAttributes([out, retval] BSTR*  
pVal);  
HRESULT SettingsTemplateAttributes([in] BSTR newVal);
```

EFT v7.4.13 and later

- Property **TimeoutSeconds** As Long

Retrieves or specifies the timeout in seconds.

```
HRESULT TimeoutSeconds([out, retval] LONG* pVal);  
HRESULT TimeoutSeconds([in] LONG newVal);
```

EFT v6.3 and later

- Property **UsePageControl** As Boolean

Retrieves or specifies whether to control the search page size.

```
HRESULT UsePageControl([out, retval] VARIANT_BOOL* pVal);  
HRESULT UsePageControl([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property **UserFilter** As String

Retrieves or specifies the user filter.

```
HRESULT UserFilter([out, retval] BSTR* pVal);  
HRESULT UserFilter([in] BSTR newVal);
```

EFT v6.3 and later

- Property **UseSearchAttributes** As Boolean

```
HRESULT UseSearchAttributes([out, retval] VARIANT_BOOL*  
pVal);  
HRESULT UseSearchAttributes([in] VARIANT_BOOL newVal);
```

EFT v7.0.4 and later

- Property **UseSSL** As Boolean

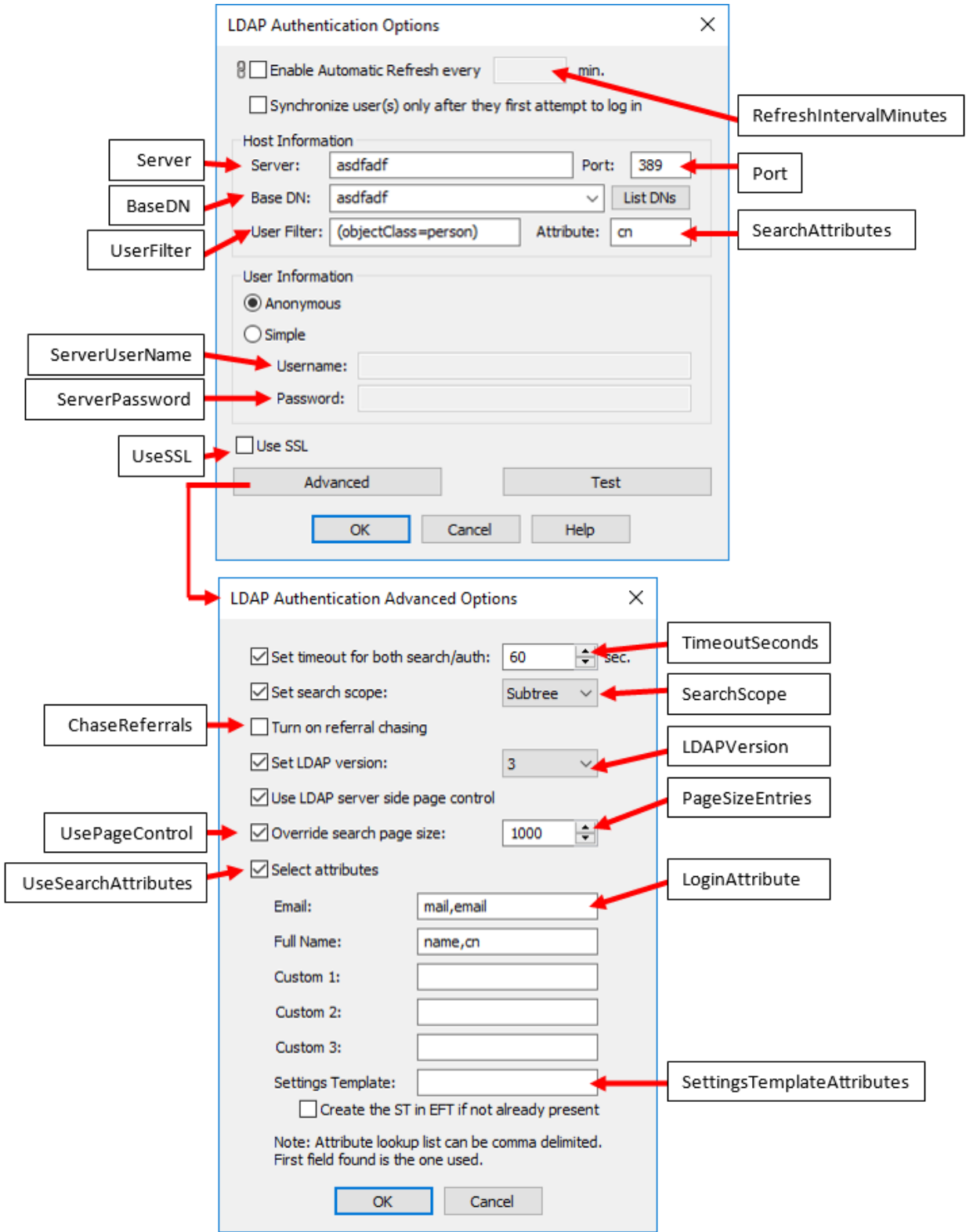
Retrieves or specifies whether to use SSL for LDAP authentication.

```
HRESULT UseSSL([out, retval] VARIANT_BOOL* pVal);  
HRESULT UseSSL([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

Dialog Box Equivalents

The ICILDAPAuthManagerSettings interface properties correlate to the following fields and controls in the EFT administration interface:



ICIODBCAuthManagerSettings Interface

The ODBC Authentication Manager Settings interface members are listed below. See also [ICISites - Multi-Site Management](#).

For an example using these properties, refer to the ODBC Authentication script in [Script Examples](#).

Properties

- Property **RefreshIntervalMinutes** As Long

Retrieves or specifies how often to refresh the user database.

```
HRESULT RefreshIntervalMinutes([out, retval] LONG* pVal);
HRESULT RefreshIntervalMinutes([in] LONG newVal);
```

EFT Server v6.3 and later

- Property **UserDatabaseConnectionString** As String

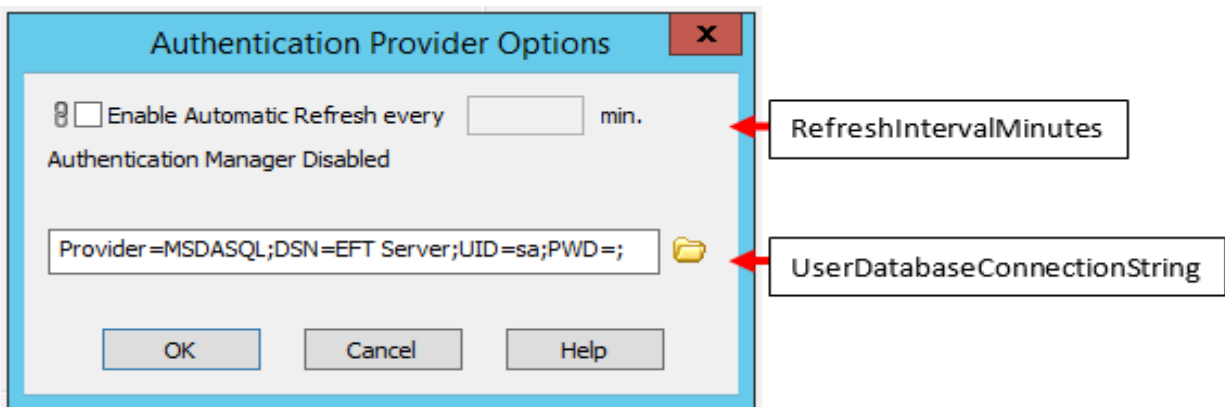
Retrieves or specifies the string used to connect to the user database.

```
HRESULT UserDatabaseConnectionString([out, retval] BSTR* pVal);
HRESULT UserDatabaseConnectionString([in] BSTR newVal);
```

EFT Server v6.3 and later

Dialog Box Equivalents

The ICIODBCAuthManagerSettings interface properties correlate to the following fields and controls in the EFT administration interface:



RSA Objects

RSA-related objects are in the [ICISite interface](#), [ICIServer Interface](#), [ICINewUserData interface](#), and [Enum Reference](#).

In the [ICIServer Interface](#):

- Function [CreatePGPKey](#)(bstrFullName As String, bstrEmailAddress As String, dtExpirationDate As Date, nKeyType As PGPKeyType, nKeyCipher As PGPKeyCipher, ulKeySize As DWord, bstrPassphrase As String) As String

Create a PGPKey on EFT.

```
HRESULT CreatePGPKey(
[in] BSTR bstrFullName,
[in] BSTR bstrEmailAddress,
[in] DATE dtExpirationDate,
[in] PGPKeyType nKeyType,
[in] PGPKeyCipher nKeyCipher,
[in] ULONG ulKeySize,
[in] BSTR bstrPassphrase,
[out,retval] BSTR* pVal);
```

PGPKeyType

- PGPKeyType_DSS = 0
- PGPKeyType_RSA = 1
- PGPKeyType_RSALegacy = 2

PGPKeyCipher

- PGPKeyCipher_IDEA = 0
- PGPKeyCipher_3DES = 1
- PGPKeyCipher_CAST5 = 2
- PGPKeyCipher_AES128 = 3
- PGPKeyCipher_AES192 = 4
- PGPKeyCipher_AES256 = 5
- PGPKeyCipher_TWOFISH = 6

Example (VBScript):

```
strKeyId = oServer.CreatePGPKey("Ivan Petrov",
    "ipetrov@mail.com", expDate,
    PGPKeyType_RSA, PGPKeyCipher_IDEA, 1024, "password")
MsgBox "New key id is " & strKeyId
```

EFT v6.3 and later

In the ICISite interface:

- Property **RSAConfigurationPath** As String

Retrieves the path to the RSA configuration file.

```
HRESULT RSAConfigurationPath([out, retval] BSTR* pVal);
HRESULT RSAConfigurationPath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "RSA config path: " & oSite.RSAConfigurationPath
```

Specifying:

```
oSite.RSAConfigurationPath = "C:\RSAConfig\SDConf.rec"
```

EFT v6.3 and later

In the [ICINewUserData](#) interface:

- Property **TwoFactorAuthentication** As SFTPAAdvBool

Specifies whether RADIUS/RSA is enabled, disabled, or inherited.

```
HRESULT TwoFactorAuthentication([out, retval] SFTPAAdvBool
    *pVal);
HRESULT TwoFactorAuthentication([in] SFTPAAdvBool newVal);
```

EFT v6.3.8 and later

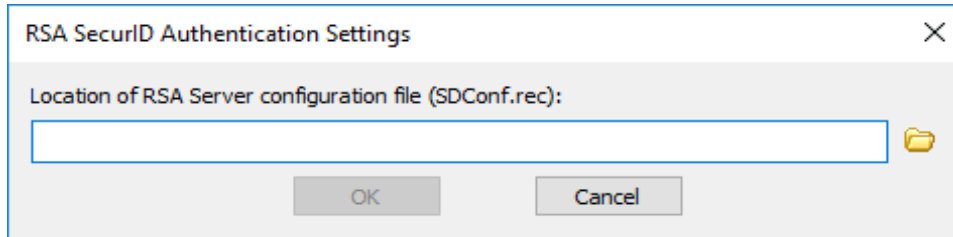
In the [Enum Reference](#):

- **PGPKeyType** (v6.3 and later)

Used in the ICIServer Interface [CreatePGPKey](#) method to specify the PGP key type.

- PGPKeyType_DSS = 0
- PGPKeyType_RSA = 1
- PGPKeyType_RSALegacy = 2

Dialog Box Equivalents



RADIUS Objects

RADIUS objects are in the [ICISite Interface](#), the [ICINewUserData Interface](#), and the [Enum Reference](#).

In the ICISite Interface:

- Property **RadiusConnectionRetriesCount** As Long

Retrieves or specifies the number of connections retries allowed to connect to the RADIUS server.

```
HRESULT RadiusConnectionRetriesCount([out, retval] long* pVal);
HRESULT RadiusConnectionRetriesCount([in] long newVal);
```

Example (VBScript):

```
Site.RadiusConnectionRetriesCount = 2
```

EFT v6.3 and later

- Property **RadiusNasIdentifier** As String

Retrieves or specifies the NAS identifier of the RADIUS server.

```
HRESULT RadiusNasIdentifier([out, retval] BSTR* pVal);
HRESULT RadiusNasIdentifier([in] BSTR newVal);
```

Example (VBScript):

```
MsgBox "NAS identifier: " & CStr(oSite.RadiusNasIdentifier)
```

EFT v6.3 and later

- Property **RadiusServerName** As String

Retrieves or specifies the name of the RADIUS server.

```
HRESULT RadiusServerName([out, retval] BSTR* pVal);
HRESULT RadiusServerName([in] BSTR newVal);
```

Example (VBScript):

```
MsgBox "Radius server name: " & CStr(oSite.
RadiusServerName)
```

EFT v6.3 and later

- Property **RadiusServerPort** As Long

Retrieves or specifies the port number of the RADIUS server.

```
HRESULT RadiusServerPort([out, retval] long* pVal);
HRESULT RadiusServerPort([in] long newVal);
```

Example (VBScript):

```
oSite. RadiusServerPort = 1645
```

EFT v6.3 and later

- Property **RadiusSharedSecret** As String

Retrieves or specifies the shared secret of the RADIUS server.

```
HRESULT RadiusSharedSecret([out, retval] BSTR* pVal);
HRESULT RadiusSharedSecret([in] BSTR newVal);
```

Example (VBScript):

```
MsgBox "Radius shared secret: " & CStr(oSite.
RadiusSharedSecret)
```

EFT v6.3 and later

- Property **RadiusTimeout** As Long

Retrieves or specifies the timeout for connection attempts to the RADIUS server in milliseconds.

```
HRESULT RadiusTimeout([out, retval] long* pVal);
HRESULT RadiusTimeout([in] long newVal);
```

Example (VBScript):

```
oSite.RadiusTimeout = 10000
```

EFT v6.3 and later

In the [ICINewUserData](#) Interface:

- Property **TwoFactorAuthentication**

Specifies whether RADIUS/RSA is enabled, disabled, or inherited.

```
HRESULT TwoFactorAuthentication([out, retval] SFTPAAdvBool *pVal);
HRESULT TwoFactorAuthentication([in] SFTPAAdvBool newVal);
```

EFT v6.3.8 and later

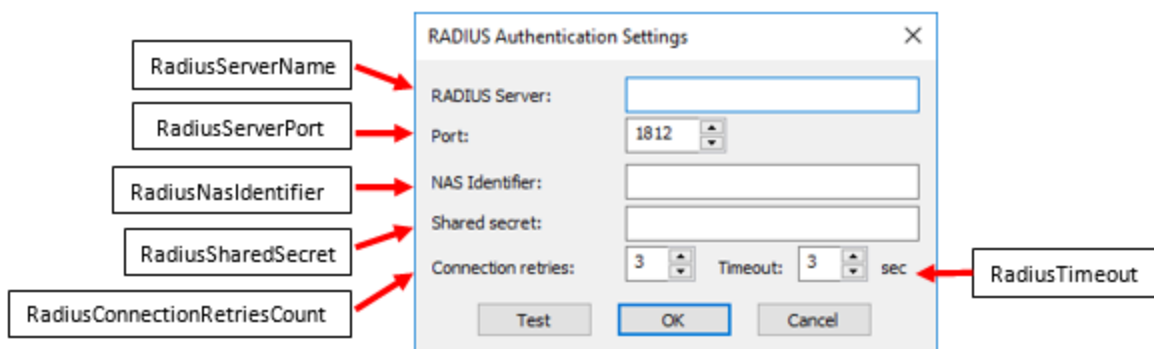
In the [Enum Reference](#):

In the [ICISite Interface](#), the **TwoFactorAuthentication** property is used to specify special authentication mode for Site users.

- TwoFactorAuth_None = 0
- TwoFactorAuth_RADIUS = 1
- TwoFactorAuth_RSA = 2

Dialog Box Equivalent

The ICISite RADIUS authentication properties correlate to the following fields and controls in the EFT administration interface:



SAML Objects

The SAML (Web SSO) objects are in the [ICISite](#) interface and [Enum Reference](#).

Refer to [illustration below](#) for dialog box equivalents.

In the ICISite interface:

- Property **EnableWebSSO** As Boolean

```
HRESULT EnableWebSSO([in] VARIANT_BOOL newVal);
HRESULT EnableWebSSO([out, retval] VARIANT_BOOL* pVal);
```

EFT v7.3.3

- Property **WebSSOAttributeName** As String

```
HRESULT WebSSOAttributeName([in] BSTR newVal);
HRESULT WebSSOAttributeName([out, retval] BSTR* pVal);
```

EFT v7.3.3

- Property **WebSSOExtendUsernameLookupToAuthProvider** As Boolean

```
HRESULT WebSSOExtendUsernameLookupToAuthProvider([out,
retval] VARIANT_BOOL* pVal);
HRESULT WebSSOExtendUsernameLookupToAuthProvider([in]
VARIANT_BOOL newVal);
```

EFT v7.3.3

- Property **WebSSOIdpEntityId** As String

```
HRESULT WebSSOIdpEntityId([out, retval] BSTR* pVal);
HRESULT WebSSOIdpEntityId([in] BSTR newVal);
```

EFT v7.3.3

- Property **WebSSOIdpPostUrl** As String

```
HRESULT WebSSOIdpPostUrl([in] BSTR newVal);
HRESULT WebSSOIdpPostUrl([out, retval] BSTR* pVal);
```

EFT v7.3.3

- Property **WebSSOIdpPublicKeyPath** As String

```
HRESULT WebSSOIdpPublicKeyPath([in] BSTR newVal);
HRESULT WebSSOIdpPublicKeyPath([out, retval] BSTR* pVal);
```

EFT v7.3.3

- Property **WebSSOJitUserProvisioning** As Boolean

```
HRESULT WebSSOJitUserProvisioning([out, retval] VARIANT_
BOOL* pVal);
HRESULT WebSSOJitUserProvisioning([in] VARIANT_BOOL newVal);
```

EFT v7.4.13

- Property **WebSSOJitUserProvisioningEmailAttribute** As String

```
HRESULT WebSSOJitUserProvisioningEmailAttribute([out,
retval] BSTR* pVal);
HRESULT WebSSOJitUserProvisioningEmailAttribute([in] BSTR
newVal);
```

EFT v8

- Property **WebSSOJitUserProvisioningEmailLocation** As Long

```
HRESULT WebSSOJitUserProvisioningEmailLocation([out, retval]
long* pVal);
HRESULT WebSSOJitUserProvisioningEmailLocation([in] long
newVal);
```

EFT v8

- Property **WebSSOJitUserProvisioningSettingsTemplate** As String

```
HRESULT WebSSOJitUserProvisioningSettingsTemplate([out,
retval] BSTR* pVal);
HRESULT WebSSOJitUserProvisioningSettingsTemplate([in] BSTR
newVal);
```

EFT v7.4.13

- Property **WebSSONameOrAttribute** As [WebSSONameOrAttribute](#)

```
HRESULT WebSSONameorAttribute([in] WebSSONameOrAttribute
newVal);
HRESULT WebSSONameorAttribute([out, retval]
WebSSONameOrAttribute *pVal);
```

EFT v7.3.3

See also [WebSSONameOrAttribute](#) enum.

- Property **WebSSONameOrAttributeIdentifierFormat** As [WebSSONameOrAttributeIdentifierFormat](#)

```
HRESULT WebSSONameOrAttributeIdentifierFormat([out, retval]
WebSSONameOrAttributeIdentifierFormat *pVal);
HRESULT WebSSONameOrAttributeIdentifierFormat([in]
WebSSONameOrAttributeIdentifierFormat newVal);
```

EFT v7.3.3

See also [WebSSONameOrAttributeIdentifierFormat](#) enum.

- Property **WebSSORedirectToSsoServiceBypassingLoginPage** As Boolean

```
HRESULT WebSSORedirectToSsoServiceBypassingLoginPage([in]
VARIANT_BOOL newVal);
HRESULT WebSSORedirectToSsoServiceBypassingLoginPage([out,
retval] VARIANT_BOOL* pVal);
```

EFT v8.0.4

- Property **WebSSOSpEntityId** As String

```
HRESULT WebSSOSpEntityId([in] BSTR newVal);
HRESULT WebSSOSpEntityId([out, retval] BSTR* pVal);
```

EFT v7.3.3

- Property **WebSSOSpReservedPath** As String

```
HRESULT WebSSOSpReservedPath([in] BSTR newVal);
HRESULT WebSSOSpReservedPath([out, retval] BSTR* pVal);
```

EFT v7.3.3

- Property **WebSSOUserNameRegex** As String

```
HRESULT WebSSOUserNameRegex([in] BSTR newVal);
HRESULT WebSSOUserNameRegex([out, retval] BSTR* pVal);
```

EFT v7.3.3

In the [Enum Reference](#):

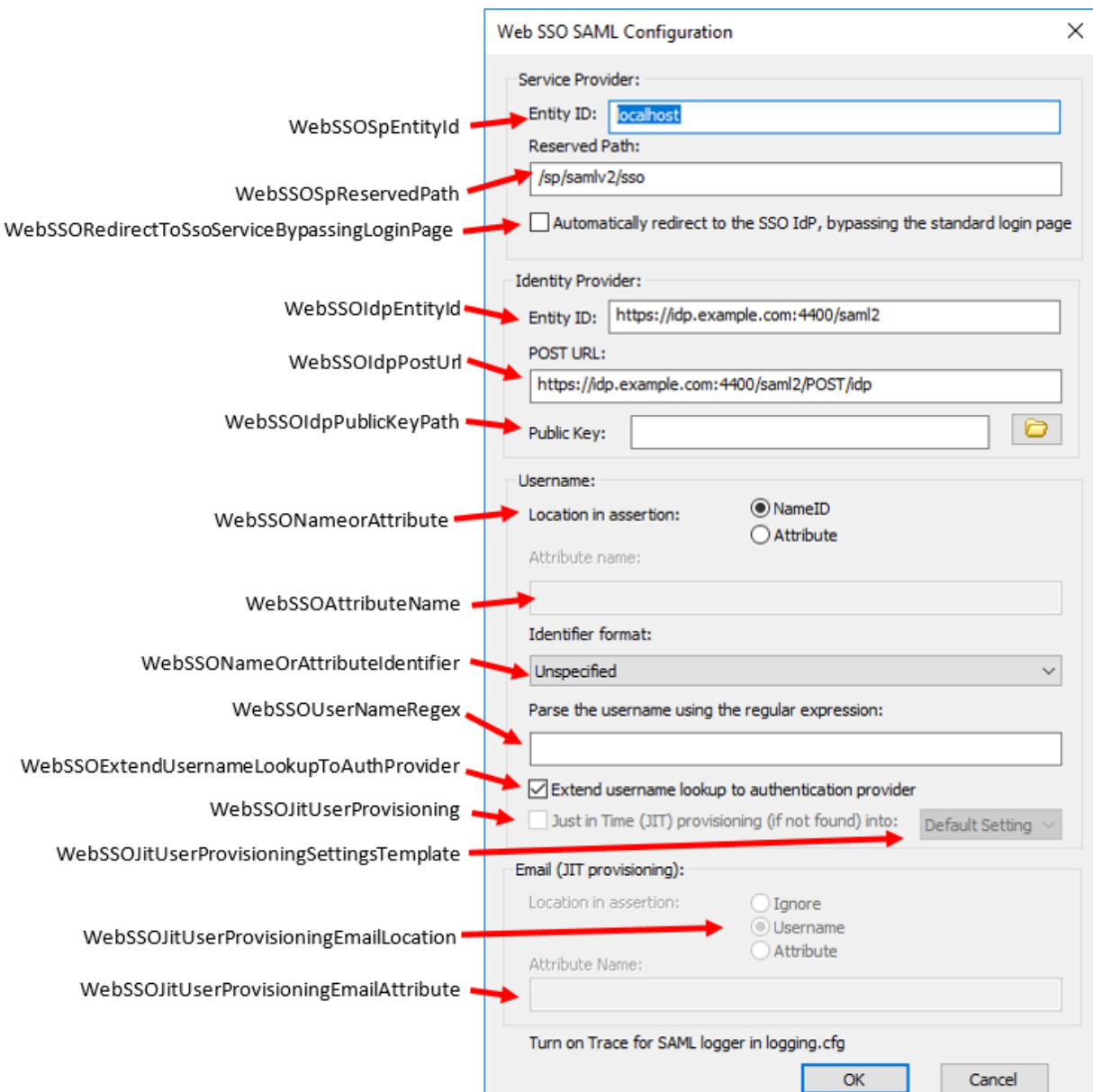
WebSSONameOrAttribute (v7.3.3 and later)

- WebSSONameld = 0,
- WebSSOAttribute = 1

WebSSONameOrAttributeIdentifierFormat (v7.3.3 and later)

- Unspecified,
- EmailAddress,
- X509SubjectName,
- WindowsDomainQualifiedName,
- KerberosPrincipalName,
- EntityIdentifier,
- URIReference,
- Basic

Dialog Box Equivalents



Cloud-Related Interfaces

This chapter contains cloud-related COM objects.

ICICloudAWSS3GeneralParams

The ICICloudAWSS3GeneralParams interface is used to define or retrieve settings for the AWS S3 storage. Used in [ICICloudTransferActionParams](#):GetAWSS3GeneralParams and SetAWSS3GeneralParams.

- Property **AccessKey** As String

```
HRESULT AccessKey([out, retval] BSTR* pVal);
HRESULT AccessKey([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **AuthType** As [AWSS3AuthType](#)

```
HRESULT AuthType([out, retval] AWSS3AuthType* pVal);
HRESULT AuthType([in] AWSS3AuthType newVal);
```

EFT v7.4.6 and later

- Property **BucketAccessStyle** As [AWSS3BucketAccessStyle](#)

```
HRESULT BucketAccessStyle([out, retval]
AWSS3BucketAccessStyle* pVal);
HRESULT BucketAccessStyle([in] AWSS3BucketAccessStyle
newVal);
```

EFT v7.4.6 and later

- Property **BucketName** As String

```
HRESULT BucketName([out, retval] BSTR* pVal);
HRESULT BucketName([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ClientEncryptionPassword** As String

```
HRESULT ClientEncryptionPassword([out, retval] BSTR *pVal);
HRESULT ClientEncryptionPassword([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ConnectionRetryAttempts** As Long

```
HRESULT ConnectionRetryAttempts([out, retval] LONG *pVal);
HRESULT ConnectionRetryAttempts([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ConnectionTimeoutSeconds** As Long

```
HRESULT ConnectionTimeoutSeconds([out, retval] LONG *pVal);
HRESULT ConnectionTimeoutSeconds([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **EncryptionType** As [AWS3EncryptionType](#)

```
HRESULT EncryptionType([out, retval] AWS3EncryptionType*
pVal);
HRESULT EncryptionType([in] AWS3EncryptionType newVal);
```

EFT v7.4.6 and later

- Property **Endpoint** As String

```
HRESULT Endpoint([out, retval] BSTR *pVal);
HRESULT Endpoint([in] BSTR newVal);
```

EFT v7.4.6 and later

- Function **GetMetadataKeys()** As Variant

```
HRESULT GetMetadataKeys([out, retval] VARIANT* pVal);
```

EFT v7.4.6 and later

- Function **GetMetadataValue(key As String)** As String

```
HRESULT GetMetadataValue([in] BSTR key, [out, retval] BSTR
*bstrValue);
```

- Property **MaxConcurrentThreads** As Long

```
HRESULT MaxConcurrentThreads([out, retval] LONG *pVal);
HRESULT MaxConcurrentThreads([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ProxyHostName** As String

```
HRESULT ProxyHostName([out, retval] BSTR *pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ProxyPassword** As String

```
HRESULT ProxyPassword([out, retval] BSTR *pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ProxyPort** As Long

```
HRESULT ProxyPort([out, retval] LONG *pVal);
HRESULT ProxyPort([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ProxyUserName** As String

```
HRESULT ProxyUserName([out, retval] BSTR *pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **Region** As String

```
HRESULT Region([out, retval] BSTR* pVal);
HRESULT Region([in] BSTR newVal);
```

EFT v7.4.6 and later

- Sub **RemoveMetadata**(key As String)

```
HRESULT RemoveMetadata([in] BSTR key);
```

EFT v7.4.6 and later

- Property **RequestorPays** As Boolean

```
HRESULT RequestorPays([out, retval] BSTR* pVal);
HRESULT RequestorPays([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **RetryDelaySeconds** As Long

```
HRESULT RetryDelaySeconds([out, retval] LONG *pVal);
HRESULT RetryDelaySeconds([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **SecretKey** As String

```
HRESULT SecretKey([out, retval] BSTR* pVal);
HRESULT SecretKey([in] BSTR newVal);
```

EFT v7.4.6 and later

- Sub **SetMetadataValue**(key As String, bstrValue As String)

```
HRESULT SetMetadataValue([in] BSTR key, [in] BSTR
bstrValue);
```

EFT v7.4.6 and later

- Property **StorageClass** As [AWSS3StorageClass](#)

```
HRESULT StorageClass([out, retval] AWSS3StorageClass* pVal);
HRESULT StorageClass([in] AWSS3StorageClass newVal);
```

EFT v7.4.6 and later

- Property **TryMultipart** As Boolean

```
HRESULT TryMultipart([out, retval] VARIANT_BOOL *pVal);
HRESULT TryMultipart([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **UseProxy** As Boolean

```
HRESULT UseProxy([out, retval] VARIANT_BOOL *pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **VerifyIntegrity** As Boolean

```
HRESULT VerifyIntegrity([out, retval] VARIANT_BOOL *pVal);
HRESULT VerifyIntegrity([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

ICICloudAWSS3UploadParams

The ICICloudAWSS3UploadParams interface is used to define or retrieve settings relating to uploading to the AWS S3 storage.

- Property **DeleteSourceAfterTransfer** As Boolean

```
HRESULT DeleteSourceAfterTransfer([out, retval] VARIANT_BOOL* pVal);
HRESULT DeleteSourceAfterTransfer([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **LocalPath** As String

```
HRESULT LocalPath([out, retval] BSTR* pVal);
HRESULT LocalPath([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **MatchingFileNamePolicy** As [AWSS3MatchingFileNamePolicy](#)

```
HRESULT MatchingFileNamePolicy([out, retval]
AWSS3MatchingFileNamePolicy* pVal);
HRESULT MatchingFileNamePolicy([in]
AWSS3MatchingFileNamePolicy newVal);
```

EFT v7.4.6 and later

- Property **RemotePath** As String

```
HRESULT RemotePath([out, retval] BSTR* pVal);
HRESULT RemotePath([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_BOOL* pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

ICICloudAWSS3DownloadParams

The ICICloudAWSS3DownloadParams interface is used to define or retrieve settings relating to downloading from the AWS S3 storage.

- Property **DeleteSourceAfterTransfer** As Boolean fterTransfer

```
HRESULT DeleteSourceAfterTransfer([out, retval] VARIANT_BOOL* pVal);
HRESULT DeleteSourceAfterTransfer([in] VARIANT_BOOL newVal);
```

EFT v7.4.6

- Property **LocalPath** As String

```
HRESULT LocalPath([out, retval] BSTR* pVal);
HRESULT LocalPath([in] BSTR newVal);
```

EFT v7.4.6

- Property **MatchingFileNamePolicy** As [AWSS3MatchingFileNamePolicy](#)

```
HRESULT MatchingFileNamePolicy([out, retval]
AWSS3MatchingFileNamePolicy* pVal);
HRESULT MatchingFileNamePolicy([in]
AWSS3MatchingFileNamePolicy newVal);
```

- Property **NoDeleteWhenSameName** As Boolean

```
HRESULT NoDeleteWhenSameName([out, retval] VARIANT_BOOL*
pVal);
HRESULT NoDeleteWhenSameName([in] VARIANT_BOOL newVal);
```

EFT v7.4.6

- Property **RemotePath** As String

```
HRESULT RemotePath([out, retval] BSTR* pVal);
HRESULT RemotePath([in] BSTR newVal);
```

EFT v7.4.6

- Property **RenameTransferredFile** As Boolean

```
HRESULT RenameTransferredFile([out, retval] VARIANT_BOOL*
pVal);
HRESULT RenameTransferredFile([in] VARIANT_BOOL newVal);
```

EFT v7.4.6

- Property **RenameTransferredFileTo** As String

```
HRESULT RenameTransferredFileTo([out, retval] BSTR* pVal);
HRESULT RenameTransferredFileTo([in] BSTR newVal);
```

EFT v7.4.6

ICloudAzureBlobGeneralParams

The ICloudAzureBlobGeneralParams interface is used to define or retrieve settings relating to the Azure storage. Used in

[ICloudTransferActionParams](#)GetAzureBlobGeneralParams and Sub SetAzureBlobGeneralParams.

- Property **AdditionalHeaders** As String

```
HRESULT AdditionalHeaders([out, retval] BSTR *pVal);
HRESULT AdditionalHeaders([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **AuthType** As [AzureBlobAuthType](#)

```
HRESULT AuthType([out, retval] AzureBlobAuthType* pVal);
HRESULT AuthType([in] AzureBlobAuthType newVal);
```

EFT v7.4.6 and later

- Property **ConnectionRetryAttempts** As Long

```
HRESULT ConnectionRetryAttempts([out, retval] LONG *pVal);
HRESULT ConnectionRetryAttempts([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ConnectionTimeoutSeconds** As Long

```
HRESULT ConnectionTimeoutSeconds([out, retval] LONG *pVal);
HRESULT ConnectionTimeoutSeconds([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ContainerURL** As String

```
HRESULT ContainerURL([out, retval] BSTR* pVal);
HRESULT ContainerURL([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **MaxConcurrentThreads** As Long

```
HRESULT MaxConcurrentThreads([out, retval] LONG *pVal);
HRESULT MaxConcurrentThreads([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ProxyHostName** As String

```
HRESULT ProxyHostName([out, retval] BSTR *pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ProxyPassword** As String

```
HRESULT ProxyPassword([out, retval] BSTR *pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **ProxyPort** As Long

```
HRESULT ProxyPort([out, retval] LONG *pVal);
```

```
HRESULT ProxyPort([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **ProxyUserName** As String

```
HRESULT ProxyUserName([out, retval] BSTR *pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **RetryDelaySeconds** As Long

```
HRESULT RetryDelaySeconds([out, retval] LONG *pVal);
HRESULT RetryDelaySeconds([in] LONG newVal);
```

EFT v7.4.6 and later

- Property **SASToken** As String

```
HRESULT SASToken([out, retval] BSTR* pVal);
HRESULT SASToken([in] BSTR newVal);
```

- EFT 7.4.9 and later

- Property **SharedKey** As String

```
HRESULT SharedKey([out, retval] BSTR* pVal);
HRESULT SharedKey([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **UseProxy** As Boolean

```
HRESULT UseProxy([out, retval] VARIANT_BOOL *pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **VerifyIntegrity** As Boolean

```
HRESULT VerifyIntegrity([out, retval] VARIANT_BOOL *pVal);
HRESULT VerifyIntegrity([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

ICloudAzureBlobUploadParams

The ICloudAzureBlobUploadParams interface is used to define or retrieve settings relating to uploading to the Azure blob storage.

- Property **DeleteSourceAfterTransfer** As Boolean

```
HRESULT DeleteSourceAfterTransfer([out, retval] VARIANT_BOOL* pVal);
HRESULT DeleteSourceAfterTransfer([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **LocalPath** As String

```
HRESULT LocalPath([out, retval] BSTR* pVal);
HRESULT LocalPath([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **MatchingFileNamePolicy** As [AzureBlobMatchingFileNamePolicy](#)

```
HRESULT MatchingFileNamePolicy([out, retval]
AzureBlobMatchingFileNamePolicy* pVal);
HRESULT MatchingFileNamePolicy([in]
AzureBlobMatchingFileNamePolicy newVal);
```

EFT v7.4.6 and later

- Property **RemotePath** As String

```
HRESULT RemotePath([out, retval] BSTR* pVal);
HRESULT RemotePath([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_
BOOL* pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL
newVal);
```

EFT v7.4.6 and later

ICICloudAzureBlobDownloadParams

The ICICloudAzureBlobDownloadParams interface is used to define or retrieve settings relating to downloading from the Azure blob storage.

- Property **DeleteSourceAfterTransfer** As Boolean

```
HRESULT DeleteSourceAfterTransfer([out, retval] VARIANT_
BOOL* pVal);
HRESULT DeleteSourceAfterTransfer([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **LocalPath** As String

```
HRESULT LocalPath([out, retval] BSTR* pVal);
HRESULT LocalPath([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **MatchingFileNamePolicy** As [AzureBlobMatchingFileNamePolicy](#)

```
HRESULT MatchingFileNamePolicy([out, retval]
AzureBlobMatchingFileNamePolicy* pVal);
HRESULT MatchingFileNamePolicy([in]
AzureBlobMatchingFileNamePolicy newVal);
```

EFT v7.4.6 and later

- Property **NoDeleteWhenSameName** As Boolean

```
HRESULT NoDeleteWhenSameName([out, retval] VARIANT_BOOL*
pVal);
HRESULT NoDeleteWhenSameName([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **RemotePath** As String

```
HRESULT RemotePath([out, retval] BSTR* pVal);
HRESULT RemotePath([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **RenameTransferredFile** As Boolean

```
HRESULT RenameTransferredFile([out, retval] VARIANT_BOOL*
pVal);
HRESULT RenameTransferredFile([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **RenameTransferredFileTo** As String

```
HRESULT RenameTransferredFileTo([out, retval] BSTR* pVal);
HRESULT RenameTransferredFileTo([in] BSTR newVal);
```

EFT v7.4.6 and later

ICICloudDownloadActionParams

The ICICloudDownloadActionParams interface is used to define or retrieve settings relating to related to the Download file from cloud storage Event Rule Action.

[ICIUploadActionParams](#) and [ICIDownloadActionParams](#) are subclasses that inherited from the parent ICITransferActionParams IDispatch interface.

- Property **CloudProviderType** As [CloudProviderType](#)

```
HRESULT CloudProviderType([out, retval] CloudProviderType*
pVal);
HRESULT CloudProviderType([in] CloudProviderType newVal);
```

EFT v7.4.6

Property **ConnectionProfileName** As String

```
HRESULT ConnectionProfileName([out, retval] BSTR *pValName);
HRESULT ConnectionProfileName([in] BSTR newValName);
```

EFT v7.4.6

- Function **GetAWSS3GeneralParams()** As [CICloudAWSS3GeneralParams](#) and Sub SetAWSS3GeneralParams(pVal As [CICloudAWSS3GeneralParams](#))

```
HRESULT GetAWSS3GeneralParams([out, retval]
    ICICloudAWSS3GeneralParams** ppVal);
HRESULT SetAWSS3GeneralParams([in]
    ICICloudAWSS3GeneralParams* pVal);
```

EFT v7.4.6

- Function **GetAWSS3DownloadParams()** As [CICloudAWSS3DownloadParams](#) and Sub SetAWSS3DownloadParams(pVal As [CICloudAWSS3DownloadParams](#))

```
HRESULT GetAWSS3UploadParams([out, retval]
    ICICloudAWSS3UploadParams** ppVal);
HRESULT SetAWSS3UploadParams([in] ICICloudAWSS3UploadParams*
    pVal);
```

EFT v7.4.6

Function **GetAzureBlobGeneralParams()** As [CICloudAzureBlobGeneralParams](#) and Sub SetAzureBlobGeneralParams(pVal As [CICloudAzureBlobGeneralParams](#))

```
HRESULT GetAzureBlobGeneralParams([out, retval]
    ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureBlobGeneralParams([in]
    ICICloudAzureBlobGeneralParams* pVal);
```

EFT v7.4.6

- Function **GetAzureBlobDownloadParams()** As [CICloudAzureBlobDownloadParams](#) and Sub SetAzureBlobDownloadParams(pVal As [CICloudAzureBlobDownloadParams](#))

```
HRESULT GetAzureBlobUploadParams([out, retval]
    ICICloudAzureBlobUploadParams** ppVal);
HRESULT SetAzureBlobUploadParams([in]
    ICICloudAzureBlobUploadParams* pVal);
```

EFT v7.4.6

ICICloudStorageMonitorEventRuleParams

The ICICloudStorageMonitorEventRuleParams interface is used to specify the cloud storage options in the Cloud Connector module.

Members (listed alphabetically)

- Property ArchiveProcessedFiles As Boolean

```
HRESULT ArchiveProcessedFiles([out, retval] VARIANT_BOOL
    *pVal);
HRESULT ArchiveProcessedFiles([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property ArchiveSubFolderName As String

```
HRESULT ArchiveSubFolderName([out, retval] BSTR *pVal);
HRESULT ArchiveSubFolderName([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property CloudProviderType As [CloudProviderType](#)

```
HRESULT CloudProviderType([out, retval] CloudProviderType*
    pVal);
HRESULT CloudProviderType([in] CloudProviderType newVal);
```

EFT v7.4.6 and later

- Property ConnectionProfileName As String

```
HRESULT ConnectionProfileName([out, retval] BSTR *pValName);
HRESULT ConnectionProfileName([in] BSTR newValName);
```

EFT v7.4.6 and later

- Function GetAWSS3GeneralParams() As [ICICloudAWSS3GeneralParams](#) and Sub SetAWSS3GeneralParams(pVal As [ICICloudAWSS3GeneralParams](#))

```
HRESULT GetAWSS3GeneralParams([out, retval]
    ICICloudAWSS3GeneralParams** ppVal);
HRESULT SetAWSS3GeneralParams([in]
    ICICloudAWSS3GeneralParams* pVal);
```

EFT v7.4.6 and later

- Function GetAzureBlobGeneralParams() As [CICloudAzureBlobGeneralParams](#) and Sub SetAzureBlobGeneralParams(pVal As [CICloudAzureBlobGeneralParams](#))

```
HRESULT GetAzureBlobGeneralParams([out, retval]
ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureBlobGeneralParams([in]
ICICloudAzureBlobGeneralParams* pVal);
```

EFT v7.4.6 and later

- Function GetAzureDLSgen2GeneralParams() As [CICloudAzureBlobGeneralParams](#) and Sub SetAzureDLSgen2GeneralParams(pVal As [CICloudAzureBlobGeneralParams](#))

```
HRESULT GetAzureDLSgen2GeneralParams([out, retval]
ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureDLSgen2GeneralParams([in]
ICICloudAzureBlobGeneralParams* pVal);
```

EFT v8.0.1 and later

- Property IncludeSubfolders As Boolean

```
HRESULT IncludeSubfolders([out, retval] VARIANT_BOOL *pVal);
HRESULT IncludeSubfolders([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property IncludeTimestampInArchivedFileNames As Boolean

```
HRESULT IncludeTimestampInArchivedFileNames([out, retval]
VARIANT_BOOL *pVal);
HRESULT IncludeTimestampInArchivedFileNames([in] VARIANT_
BOOL newVal);
```

EFT v7.4.6 and later

- Property Path As String

```
HRESULT Path([out, retval] BSTR *pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property ScanInterval As Long

```
HRESULT ScanInterval([out, retval] long *pVal);
HRESULT ScanInterval([in] long newVal);
```

EFT v7.4.6 and later

- Property **ScanIntervalType** As [CloudStorageMonitorScanIntervalUnitType](#)

```
HRESULT ScanIntervalType([out, retval]
CloudStorageMonitorScanIntervalUnitType *pVal);
HRESULT ScanIntervalType([in]
CloudStorageMonitorScanIntervalUnitType newVal);
```

EFT v7.4.6 and later

ICICloudTransferActionParams

The ICICloudTransferActionParams interface is used to define or retrieve settings related to the Copy/Move/Download file from cloud storage Event Rule Action.

- Property **CloudProviderType** As [CloudProviderType](#)

```
HRESULT CloudProviderType([out, retval] CloudProviderType*
pVal);
HRESULT CloudProviderType([in] CloudProviderType newVal);
```

EFT v7.4.6

- Property **ConnectionProfileName** As String

```
HRESULT ConnectionProfileName([out, retval] BSTR *pValName);
HRESULT ConnectionProfileName([in] BSTR newValName);
```

EFT v7.4.6

- Function **GetAWSS3GeneralParams()** As [CICloudAWSS3GeneralParams](#) and Sub **SetAWSS3GeneralParams(pVal As [CICloudAWSS3GeneralParams](#))**

```
HRESULT GetAWSS3GeneralParams([out, retval]
ICICloudAWSS3GeneralParams** ppVal);
HRESULT SetAWSS3GeneralParams([in]
ICICloudAWSS3GeneralParams* pVal);
```

EFT v7.4.6

- Function **GetAzureBlobGeneralParams()** As [CICloudAzureBlobGeneralParams](#) and Sub **SetAzureBlobGeneralParams(pVal As [CICloudAzureBlobGeneralParams](#))**

```
HRESULT GetAzureBlobGeneralParams([out, retval]
ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureBlobGeneralParams([in]
```

```
ICICloudAzureBlobGeneralParams* pVal);
```

EFT v7.4.6

ICICloudUploadActionParams

The ICICloudUploadParams interface is used to define or retrieve settings related to the Copy/Move file from cloud storage Event Rule Action. [ICIUploadActionParams](#) and [ICIDownloadActionParams](#) are subclasses that inherited from the parent [ICITransferActionParams](#) IDispatch interface.

- Property **CloudProviderType** As [CloudProviderType](#)

```
HRESULT CloudProviderType([out, retval] CloudProviderType*
pVal);
HRESULT CloudProviderType([in] CloudProviderType newVal);
```

EFT v7.4.5

Property **ConnectionProfileName** As String

```
HRESULT ConnectionProfileName([out, retval] BSTR *pValName);
HRESULT ConnectionProfileName([in] BSTR newValName);
```

EFT v7.4.5

- Function **GetAWSS3GeneralParams()** As [CICloudAWSS3GeneralParams](#) and Sub SetAWSS3GeneralParams(pVal As [CICloudAWSS3GeneralParams](#))

```
HRESULT GetAWSS3GeneralParams([out, retval]
ICICloudAWSS3GeneralParams** ppVal);
HRESULT SetAWSS3GeneralParams([in]
ICICloudAWSS3GeneralParams* pVal);
```

EFT v7.4.5

- Function **GetAWSS3UploadParams()** As [CICloudAWSS3UploadParams](#) and Sub SetAWSS3UploadParams(pVal As [CICloudAWSS3UploadParams](#))

```
HRESULT GetAWSS3UploadParams([out, retval]
ICICloudAWSS3UploadParams** ppVal);
HRESULT SetAWSS3UploadParams([in] ICICloudAWSS3UploadParams*
pVal);
```

EFT v7.4.5

- Function **GetAzureBlobGeneralParams()** As [CICloudAzureBlobGeneralParams](#) and Sub SetAzureBlobGeneralParams(pVal As [CICloudAzureBlobGeneralParams](#))

```
HRESULT GetAzureBlobGeneralParams([out, retval]
ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureBlobGeneralParams([in]
ICICloudAzureBlobGeneralParams* pVal);
```

EFT v7.4.5

- Function **GetAzureBlobUploadParams()** As [CICloudAzureBlobUploadParams](#) and Sub SetAzureBlobUploadParams(pVal As [CICloudAzureBlobUploadParams](#))

```
HRESULT GetAzureBlobUploadParams([out, retval]
ICICloudAzureBlobUploadParams** ppVal);
HRESULT SetAzureBlobUploadParams([in]
ICICloudAzureBlobUploadParams* pVal);
```

EFT v7.4.5

ICICConnectionProfileParams

The ICICConnectionProfileParams interface is used to define the Connection Profiles used in Event Rules and Cloud Connector module.

Members (Listed alphabetically)

- Property **AutoLogin** As Boolean

```
HRESULT AutoLogin([out, retval] VARIANT_BOOL *pVal);
HRESULT AutoLogin([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **CloudProviderType** As [CloudProviderType](#)

```
HRESULT Type([out, retval] ConnectionProfileType *pVal);
HRESULT Type([in] ConnectionProfileType newVal);
```

EFT v7.4.6 and later

- Property **ConnectionRetryAttempts** As Long

```
HRESULT ConnectionRetryAttempts([out, retval] LONG *pVal);
HRESULT ConnectionRetryAttempts([in] LONG newVal);
```

EFT v7.2 and later

- Property **ConnectionTimeoutSeconds** As Long

```
HRESULT ConnectionTimeoutSeconds([out, retval] LONG *pVal);
HRESULT ConnectionTimeoutSeconds([in] LONG newVal);
```

EFT v7.2 and later

- Property **Description** As String

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

EFT v7.2 and later

- Property **FTPDataConnectionMode** As [FTPDataConnectionMode](#)

```
HRESULT FTPDataConnectionMode([out, retval]
FTPDataConnectionMode *pVal);
HRESULT FTPDataConnectionMode([in] FTPDataConnectionMode
newVal);
```

EFT v7.2 and later

- Property **FTPDataConnectionPortMax** As Long

```
HRESULT FTPDataConnectionPortMax([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMax([in] LONG newVal);
```

EFT v7.2 and later

- Property **FTPDataConnectionPortMin** As Long

```
HRESULT FTPDataConnectionPortMin([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMin([in] LONG newVal);
```

EFT v7.2 and later

- Property **FTPSClearCommandChannel** As Boolean

```
HRESULT FTPSClearCommandChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT FTPSClearCommandChannel([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **FTPSClearDataChannel** As Boolean

```
HRESULT FTPSClearDataChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT FTPSClearDataChannel([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Function **GetACL()** As Object

```
HRESULT GetACL([out, retval] IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetAWSS3GeneralParams()** As [CICloudAWSS3GeneralParams](#) and Sub **SetAWSS3GeneralParams**(pVal As [CICloudAWSS3GeneralParams](#))

```
HRESULT GetAWSS3GeneralParams([out, retval]
ICICloudAWSS3GeneralParams** ppVal);
HRESULT SetAWSS3GeneralParams([in]
ICICloudAWSS3GeneralParams* pVal);
```

EFT v7.4.6 and later

- Function **GetAzureBlobGeneralParams()** As [CICloudAzureBlobGeneralParams](#) and Sub **SetAzureBlobGeneralParams**(pVal As [CICloudAzureBlobGeneralParams](#))

```
HRESULT GetAzureBlobGeneralParams([out, retval]
ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureBlobGeneralParams([in]
ICICloudAzureBlobGeneralParams* pVal);
```

EFT v7.4.6 and later

- Property **GUID** As String

```
HRESULT GUID([out, retval] BSTR *pVal);
HRESULT GUID([in] BSTR *pVal);
```

EFT v7.4.11 and later

- Property **Host** As String

```
HRESULT Host([out, retval] BSTR *pVal);
HRESULT Host([in] BSTR newVal);
```

EFT v7.2 and later

- Property **MaxConcurrentThreads** As Long

```
HRESULT MaxConcurrentThreads([out, retval] LONG *pVal);
HRESULT MaxConcurrentThreads([in] LONG newVal);
```

EFT v7.2 and later

- Property **Name As String**

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Password As String**

```
HRESULT Password([out, retval] BSTR *pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Port As Long**

```
HRESULT Port([out, retval] long *pVal);
HRESULT Port([in] long newVal);
```

EFT v7.2 and later

- Property **PreserveLocalFileTime As Boolean**

```
HRESULT PreserveLocalFileTime([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreserveLocalFileTime([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **PreserveRemoteFileTime As Boolean**

```
HRESULT PreserveRemoteFileTime([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreserveRemoteFileTime([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **PrivateKeyPassword As String**

```
HRESULT PrivateKeyPassword([out, retval] BSTR *pVal);
HRESULT PrivateKeyPassword([in] BSTR newVal);
```

EFT v7.2 and later

- Property **PrivateKeyPath** As String

```
HRESULT PrivateKeyPath([out, retval] BSTR *pVal);
HRESULT PrivateKeyPath([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Protocol** As [NetworkProtocol](#)

```
HRESULT Protocol([out, retval] NetworkProtocol *pVal);
HRESULT Protocol([in] NetworkProtocol newVal);
```

EFT v7.2 and later

- Property **ProxyAdvCustomCmds** As String

```
HRESULT ProxyAdvCustomCmds([out, retval] BSTR *pVal);
HRESULT ProxyAdvCustomCmds([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyAdvSeparator** As String

```
HRESULT ProxyAdvSeparator([out, retval] BSTR *pVal);
HRESULT ProxyAdvSeparator([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyAuthenticationType** As [ProxyAuthenticationType](#)

```
HRESULT ProxyAuthenticationType([out, retval]
ProxyAuthenticationType *pVal);
HRESULT ProxyAuthenticationType([in] ProxyAuthenticationType
newVal);
```

EFT v7.2 and later

- Property **ProxyHostName** As String

```
HRESULT ProxyHostName([out, retval] BSTR *pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyPassword** As String

```
HRESULT ProxyPassword([out, retval] BSTR *pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyPort** As Long

```
HRESULT ProxyPort([out, retval] LONG *pVal);
HRESULT ProxyPort([in] LONG newVal);
```

EFT v7.2 and later

- Property **ProxyType** As [ProxyType](#)

```
HRESULT ProxyType([out, retval] ProxyType *pVal);
HRESULT ProxyType([in] ProxyType newVal);
```

EFT v7.2 and later

- Property **ProxyUserName** As String

```
HRESULT ProxyUserName([out, retval] BSTR *pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **PublicKeyPath** As String

```
HRESULT PublicKeyPath([out, retval] BSTR *pVal);
HRESULT PublicKeyPath([in] BSTR newVal);
```

EFT v7.2 and later

- Property **RetryDelaySeconds** As Long

```
HRESULT RetryDelaySeconds([out, retval] LONG *pVal);
HRESULT RetryDelaySeconds([in] LONG newVal);
```

EFT v7.2 and later

- Sub **SetACL**(pVal As Object)

```
HRESULT SetACL([in] IDispatch* pVal);
```

EFT v7.2 and later

- Property **SOCKSHostName** As String

```
HRESULT SOCKSHostName([out, retval] BSTR *pVal);
HRESULT SOCKSHostName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **SOCKSPassword** As String

```
HRESULT SOCKSPassword([out, retval] BSTR *pVal);  
HRESULT SOCKSPassword([in] BSTR newVal);
```

EFT v7.2 and later

- Property **SOCKSPort** As Long

```
HRESULT SOCKSPort([out, retval] LONG *pVal);  
HRESULT SOCKSPort([in] LONG newVal);
```

EFT v7.2 and later

- Property **SOCKSType** As [SOCKSType](#)

```
HRESULT SOCKSType([out, retval] SOCKSType *pVal);  
HRESULT SOCKSType([in] SOCKSType newVal);
```

EFT v7.2 and later

- Property **SOCKSUseAuthentication** As Boolean

```
HRESULT SOCKSUseAuthentication([out, retval] VARIANT_BOOL  
*pVal);  
HRESULT SOCKSUseAuthentication([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **SOCKSUserName** As String

```
HRESULT SOCKSUserName([out, retval] BSTR *pVal);  
HRESULT SOCKSUserName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **TextFileTypes** As String

```
HRESULT TextFileTypes([out, retval] BSTR *pVal);  
HRESULT TextFileTypes([in] BSTR newVal);
```

EFT v7.2 and later

- Property **TransferEncoding** As [TransferEncodingType](#)

```
HRESULT TransferEncoding([out, retval] TransferEncodingType  
*pVal);  
HRESULT TransferEncoding([in] TransferEncodingType newVal);
```

EFT v7.2 and later

- Property **TrustedPublicKeyPath** As String

```
HRESULT TrustedPublicKeyPath([out, retval] BSTR *pVal);
HRESULT TrustedPublicKeyPath([in] BSTR newVal);
```

EFT v7.2 and later

- Property **type** As [ConnectionProfileType](#)

```
HRESULT Type([out, retval] ConnectionProfileType *pVal);
[propput, id(1), helpstring("property Type")]
HRESULT Type([in] ConnectionProfileType newVal);
```

EFT v7.4.6 and later

- Property **UseProxy** As Boolean

```
HRESULT UseProxy([out, retval] VARIANT_BOOL *pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **User** As String

```
HRESULT User([out, retval] BSTR *pVal);
HRESULT User([in] BSTR newVal);
```

EFT v7.2 and later

- Property **UseSOCKS** As Boolean

```
HRESULT UseSOCKS([out, retval] VARIANT_BOOL *pVal);
HRESULT UseSOCKS([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **ValidateIntegrity** As Boolean

```
HRESULT ValidateIntegrity([out, retval] VARIANT_BOOL *pVal);
HRESULT ValidateIntegrity([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

Content Integrity Control Objects

This chapter contains CIC-related COM objects.

In addition to the [ICContentIntegrityControlActionParams](#) and [ICContentIntegrityControlProfile](#) interfaces, Content Integrity Control (CIC) settings can be found in the [ICIServer Interface](#) and [Enum Reference](#).

In the ICIServer interface:

- Sub **AddContentIntegrityControlProfile**(pProfile As Object)

Adds a new profile to use with an ICAP (antivirus and DLP) server.

```
HRESULT AddContentIntegrityControlProfile([in] IDispatch*
pProfile);
```

EFT 7.0.3 and later

- Property **ContentIntegrityControlProfileList** As Variant

```
HRESULT ContentIntegrityControlProfileList([out, retval]
VARIANT *pVal);
```

EFT v7.0.3 and later

- Sub **DeleteContentIntegrityControlProfile**(pProfileName As String)

```
HRESULT DeleteContentIntegrityControlProfile([in] BSTR
pProfileName);
```

(See also the [ICContentIntegrityControlActionParams](#) ProfileName property.)

EFT v7.0.3 and later

- Function **GetContentIntegrityControlProfile**(profileName As String) As Long

```
HRESULT GetContentIntegrityControlProfile([in] BSTR
profileName, [out, retval] long *profileIndex);
```

EFT v7.0.3 and later

- Function **TestContentIntegrityControlProfile**(pProfile As Object) As Boolean

```
HRESULT TestContentIntegrityControlProfile([in] IDispatch*
pProfile, [out, retval] VARIANT_BOOL* pVal);
```

EFT v7.0.3 and later

- Sub **UpdateContentIntegrityControlProfile**(pProfile As Object)

```
HRESULT UpdateContentIntegrityControlProfile([in] IDispatch* pProfile);
```

EFT v7.0.3 and later

In the Enum Reference:

EventActionType

Used in the ICIEventAction Interface [type](#) property.

- ContentIntegrityControl = 8192 (added v7.0.3)

ICAPMode (v7.0.3 and later)

Used in the [ICContentIntegrityControlProfile](#) interface IcapMode property.

- REQMOD
- RESPMOD

IcapMaxScanSizeUnits (v7.0.3 and later)

Used in the [ICContentIntegrityControlProfile](#) interface MaxScanSizeUnits property.

- ScanSizeBytes = 0
- ScanSizeKBytes = 1
- ScanSizeMBytes = 2
- ScanSizeGBytes = 3

ICContentIntegrityControlActionParams

The ICContentIntegrityControlActionParams interface refers to the Content Integrity Control dialog box used in the File Scan Action in Event Rules.

Properties

- Property **AdHocProfile** As Object (accessing this property on anICContentIntegrityControlActionParamsobject that uses a preset profile will cause the script to fail.

```
HRESULT AdHocProfile([out, retval] IDispatch** profile);
HRESULT AdHocProfile([in] IDispatch* profile);
```

EFT v7.0.3

- Property **File** As String

```
HRESULT File([out, retval] BSTR *pVal);
HRESULT File([in] BSTR newVal);
```

EFT v7.0.3

- Property **ProfileName** As String

```
HRESULT ProfileName([out, retval] BSTR *pVal);
HRESULT ProfileName([in] BSTR newVal);
```

EFT v7.0.3

- [ICContentIntegrityControlProfile](#)
- [ICIServer - Server Interface:](#)
- [AddContentIntegrityControlProfile](#)
- [ContentIntegrityControlProfileList](#)
- [DeleteContentIntegrityControlProfile](#)
- [GetContentIntegrityControlProfile](#)
- [TestContentIntegrityControlProfile](#)
- [UpdateContentIntegrityControlProfile](#)

ICContentIntegrityControlProfile

The ICContentIntegrityControlProfile interface is used to define Content Integrity Control profiles used in the File Scan file Control Action in Event Rules. Many of these settings depend on the settings on the ICAP server. (You can also define a profile when defining the Event Rule, as needed.)

Properties

- Property **AuditResponseHeaders** As String

```
HRESULT AuditResponseHeaders([out, retval] BSTR *pVal);
HRESULT AuditResponseHeaders([in] BSTR newVal);
```

EFT v7.0.3 and later

- Property **AuditResponseHeadersEnabled** As Long

```
HRESULT AuditResponseHeadersEnabled([out, retval] BOOL
*pVal);
HRESULT AuditResponseHeadersEnabled([in] BOOL newVal);
```

EFT v7.0.3 and later

- Property ConnectivityErrorsPolicy As [IcapErrorsPolicy](#)

```
HRESULT ConnectivityErrorsPolicy([in] IcapErrorsPolicy
newVal);
HRESULT ConnectivityErrorsPolicy([out, retval]
IcapErrorsPolicy* pVal);
```

EFT v8.0.5 and later

- Property Host As String

```
HRESULT Host([out, retval] BSTR *pVal);
HRESULT Host([in] BSTR newVal);
```

EFT v7.0.3 and later

- Property HTTPErrorsPolicy As [IcapErrorsPolicy](#)

```
HRESULT HTTPErrorsPolicy([in] IcapErrorsPolicy newVal);
HRESULT HTTPErrorsPolicy([out, retval] IcapErrorsPolicy*
pVal);
```

EFT v8.0.5 and later

- Property HTTPServerHost As String

```
HRESULT HTTPServerHost([in] BSTR newVal);
HRESULT HTTPServerHost([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

- Property IcapMode As [IcapMode](#)

```
HRESULT IcapMode([out, retval] IcapMode *pVal);
HRESULT IcapMode([in] IcapMode newVal);
```

EFT v7.0.3 and later

- Property ICAPRedactionsPolicy As [IcapErrorsPolicy](#)

```
HRESULT ICAPRedactionsPolicy([in] IcapErrorsPolicy newVal);
HRESULT ICAPRedactionsPolicy([out, retval] IcapErrorsPolicy*
pVal);
```

- EFT v8.0.5 and later

Property ICAPViolationsPolicy As [IcapErrorsPolicy](#)


```
HRESULT ICAPViolationsPolicy([in] IcapErrorsPolicy newVal);
HRESULT ICAPViolationsPolicy([out, retval] IcapErrorsPolicy*
pVal);
```

EFT v8.0.5 and later

- Property MaxScanSize As Long

```
HRESULT MaxScanSize([out, retval] long *pVal);
HRESULT MaxScanSize([in] long newVal);
```

EFT v7.0.3 and later

- Property MaxScanSizeEnabled As Long

```
HRESULT MaxScanSizeEnabled([out, retval] BOOL *pVal);
HRESULT MaxScanSizeEnabled([in] BOOL newVal);
```

EFT v7.0.3 and later

- Property MaxScanSizeUnits As [IcapMaxScanSizeUnits](#)

```
HRESULT MaxScanSizeUnits([out, retval] IcapMaxScanSizeUnits
*pVal);
HRESULT MaxScanSizeUnits([in] IcapMaxScanSizeUnits newVal);
```

(See also the [IcapMaxScanSizeUnits](#) enum.)

EFT v7.0.3 and later

- Property Name As String

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

EFT v7.0.3 and later

- Property OverrideHTTPServerHost As Boolean

```
HRESULT OverrideHTTPServerHost([in] VARIANT_BOOL newVal);
HRESULT OverrideHTTPServerHost([out, retval] VARIANT_BOOL*
pVal);
```

EFT v8.0.5 and later

- Property OverrideXAuthGroupsHeader As Boolean

```
HRESULT OverrideXAuthGroupsHeader([in] VARIANT_BOOL newVal);
HRESULT OverrideXAuthGroupsHeader([out, retval] VARIANT_BOOL* pVal);
```

EFT v8.0.5 and later

- Property OverrideXAuthUserHeader As Boolean

```
HRESULT OverrideXAuthUserHeader([in] VARIANT_BOOL newVal);
HRESULT OverrideXAuthUserHeader([out, retval] VARIANT_BOOL* pVal);
```

EFT v8.0.5 and later

- Property OverrideXClientIPHeader As Boolean

```
HRESULT OverrideXClientIPHeader([in] VARIANT_BOOL newVal);
HRESULT OverrideXClientIPHeader([out, retval] VARIANT_BOOL* pVal);
```

EFT v8.0.5 and later

- Property OverrideXServerIPHeader As Boolean

```
RESULT OverrideXServerIPHeader([in] VARIANT_BOOL newVal);
HRESULT OverrideXServerIPHeader([out, retval] VARIANT_BOOL* pVal);
```

EFT v8.0.5 and later

- Property OverrideXSubscriberIDHeader As Boolean

```
HRESULT OverrideXSubscriberIDHeader([out, retval] VARIANT_BOOL* pVal);
HRESULT OverrideXSubscriberIDHeader([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

- Property Path As String

```
HRESULT Path([out, retval] BSTR *pVal);
        HRESULT Path([in] BSTR newVal);
```

EFT v7.0.3 and later

- Property Port As Long

```
HRESULT Port([out, retval] long *pVal);
                HRESULT Port([in] long newVal);
```

EFT v7.0.3 and later

- Property ResponseParsingResponseBodyText As String

```
HRESULT ResponseParsingResponseBodyText([out, retval] BSTR
    *pVal);
HRESULT ResponseParsingResponseBodyText([in] BSTR newVal);
```

EFT v7.0.3 and later

- Property ResponseParsingResponseHeadersText As String

```
HRESULT ResponseParsingResponseHeadersText([out, retval]
    BSTR *pVal);
HRESULT ResponseParsingResponseHeadersText([in] BSTR
    newVal);
```

EFT v7.0.3 and later

- Property ViolationIsNonBlocking As Boolean

```
HRESULT ViolationIsNonBlocking([out, retval] VARIANT_BOOL
    *pVal);
HRESULT ViolationIsNonBlocking([in] VARIANT_BOOL newVal);
```

EFT v7.0.3 and later

- Property XAuthGroupsHeader As String

```
HRESULT XAuthGroupsHeader([in] BSTR newVal);
HRESULT XAuthGroupsHeader([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

- Property XAuthUserHeader As String

```
HRESULT XAuthUserHeader([in] BSTR newVal);
                HRESULT XAuthUserHeader([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

- Property XClientIPHeader As String

```
HRESULT XClientIPHeader([in] BSTR newVal);
HRESULT XClientIPHeader([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

- Property XServerIPHeader As String

```
HRESULT XServerIPHeader([in] BSTR newVal);
HRESULT XServerIPHeader([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

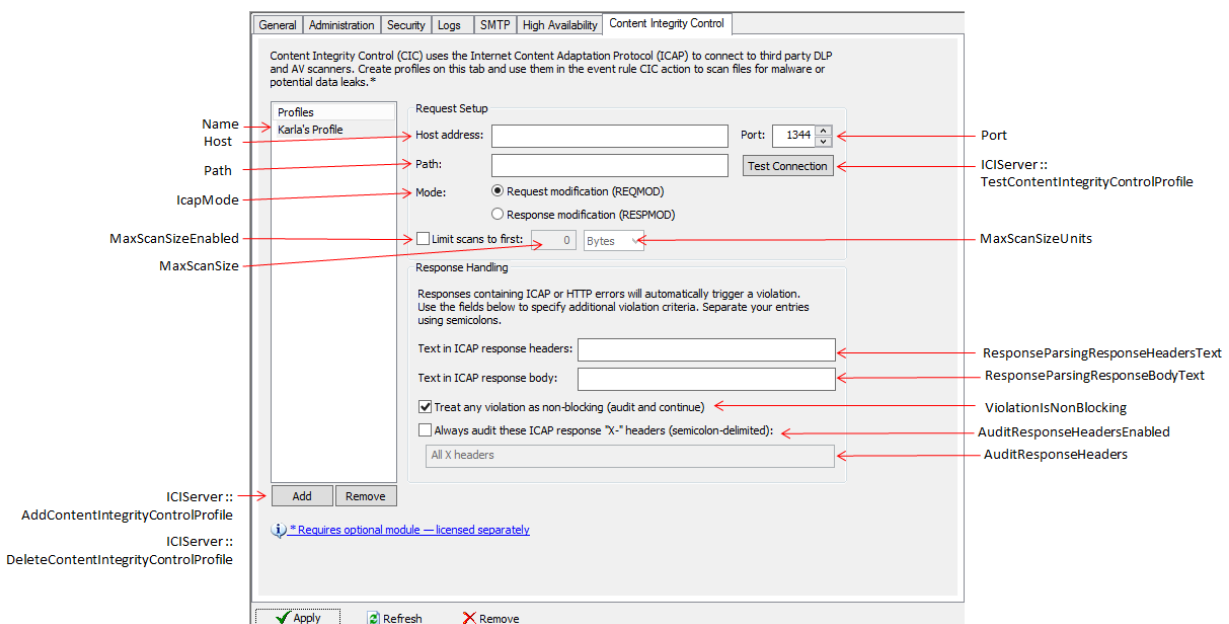
- Property XSubscriberIDHeader As String

```
HRESULT XSubscriberIDHeader([in] BSTR newVal);
HRESULT XSubscriberIDHeader([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

Dialog Box Equivalents

In v8.0.4 and later:



Script Examples

Adding / Updating a Profile

```
.....
''Add profile
.....
```

```

Set newProfile = WScript.CreateObject
("SFTPComInterface.CIContentIntegrityControlProfile")
newProfile.Name = "test3"
newProfile.Host = "localhost3"
newProfile.Port = "3333"
// ....
SFTPComServer.AddContentIntegrityControlProfile newProfile
'Update profile
newProfile.Host = "newHost"
i = SFTPComServer.GetContentIntegrityControlProfile("test3")
SFTPComServer.UpdateContentIntegrityControlProfile newProfile, i

```

Looping Through the Profile List and Printing Basic Information

```

profiles = SFTPComServer.ContentIntegrityControlProfileList
For Each profile In profiles WScript.Echo "Profile: " &
profile.Name & "; Host: " & profile.Host & "; Port: " &
profile.Port
Next

```

Building the Action Parameters, Using Ad Hoc Profiles

```

Set CICActionParams = WScript.CreateObject(
"SFTPComInterface.CIContentIntegrityControlActionParams")
'This ActionParams is now linked to Profile1 in the server list
CICActionParams.ProfileName = "Profile1"
CICActionParams.File = "File"
'Setting an Ad Hoc Profile
Set adHocProfile = WScript.CreateObject
("SFTPComInterface.CIContentIntegrityControlProfile")
adHocProfile.Host = "localhost3"
adHocProfile.Port = "2222"
CICActionParams.AdHocProfile = adHocProfile
'If we set the profile name to a saved profile
and then try to get the ad hoc profile property, we'll get an
error
CICActionParams.ProfileName = "Profile1"
Set AdHocProfile = CICActionParams.AdHocProfile

```

DMZ Gateway-Related Objects

This chapter contains DMZ Gateway-related COM objects.

In addition to the [ICIDMZGateway Interface](#), other objects related to DMZ Gateway are in the [ICISite Interface](#), [ICISiteBackupInfo Interface](#), [ICITransferActionParams Interface](#), [ICIAS2OutboundSettings Interface](#), and [Enum Reference](#).

In the [ICISite Interface](#):

- Property [DMZConnected](#) As Boolean (read only)
- Property [DMZGateway](#) As Object (read-only)

In the [ICISiteBackupInfo Interface](#):

- Property [DMZAddress](#) As String
- Property [DMZEnabled](#) As Boolean
- Property [DMZPort](#) As Long

In the [ICITransferActionParams Interface](#):

- Property [ProxyType](#) As ProxyType

In the [ICIAS2OutboundSettings Interface](#):

- Property [ProxyType](#) As AS2ProxyType

In the [Enum Reference](#):

- [AS2ProxyType](#) (v6.3 and later)
- [ProxyType](#) (v6.2.31 and later)

ICIDMZGateway Interface

Use the ICIDMZGateway interface to edit the DMZ Gateway configuration in EFT. See also the [ICISite::DMZGateway](#) property.

```
interface ICIDMZGateway : IDispatch
```

Properties

- Property AccelerationPort As Long

Retrieves or specifies the acceleration port.

```
HRESULT AccelerationPort([out,retval] LONG* pVal);
HRESULT AccelerationPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Acceleration port: " & CStr(oGW.AccelerationPort)
```

Specifying:

```
oGW.AccelerationPort = 1234
```

EFT v7.3 and later

- Property **AllowPASVMode** As Boolean

(In v6.3.0, it was "AssignPASVMode")

Retrieves or specifies whether to enable PASV mode.

```
HRESULT AllowPASVMode([out,retval] VARIANT_BOOL* pVal);
HRESULT AllowPASVMode(VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Passive mode allow: " & CStr(oGW.AllowPASVMode)
```

Specifying:

```
oGW.AllowPASVMode = True
```

EFT v6.3 and later

- Property **CertificatePath** As String

```
HRESULT CertificatePath([out, retval] BSTR* pVal);
HRESULT CertificatePath([in] BSTR newVal);
```

EFT v7.4.7 and later

- Property **Enable** As Boolean

Retrieves or specifies whether to enable DMZ Gateway.

```
HRESULT Enable([out,retval] VARIANT_BOOL* pVal);
HRESULT Enable([in] VARIANT_BOOL newVal);
```

Example (VB Script):

Retrieving:

```
MsgBox "DMZ Gateway is enabled: " & CStr(oGW.Enable)
```

Specifying:

```
oGW.Enable = False
```

EFT v6.3 and later

- Property **EnableHTTPSAcceleration** As Boolean

Retrieves or specifies whether to enable acceleration.

```
HRESULT EnableHTTPSAcceleration([out,retval] VARIANT_BOOL*
pVal);
HRESULT EnableHTTPSAcceleration(VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Acceleration enabled: " & CStr
(oGW.EnableHTTPSAcceleration)
```

Specifying:

```
oGW.EnableHTTPSAcceleration = True
```

EFT v7.3 and later

- Property **FTPPort** As Long

Retrieves or specifies the FTP port.

```
HRESULT FTPPort([out,retval] LONG* pVal);
HRESULT FTPPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "FTP port for external connections: " & CStr
(oGW.FTPPort)
```

Specifying:

```
oGW.FTPPort = 21
```

EFT v6.3 and later

- Property **FTPSEsplPort** As Long

Retrieves or specifies the FTPS explicit port.

```
HRESULT FTPSEsplPort([out,retval] LONG* pVal);
HRESULT FTPSEsplPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "FTPS explicit port for external connections: " &
CStr(oGW.FTPSEsplPort)
```

Specifying:

```
oGW.FTPSEsplPort = 21
```

EFT v6.3 and later

- Property **FTPSImplPort** As Long

Retrieves or specifies the FTP implicit port.

```
HRESULT FTPSImplPort([out,retval] LONG* pVal);
HRESULT FTPSImplPort([in] LONG newVal)
```

Example (VBScript):

Retrieving:

```
MsgBox "FTPS implicit port for external connections: " &
CStr(oGW.FTPSImplPort)
```

Specifying:

```
oGW.FTPSImplPort = 990
```

EFT v6.3 and later

- Property **GatewayIP** As String

Retrieves or specifies the DMZ Gateway IP address.

```
HRESULT GatewayIP([out,retval] BSTR* pVal);
HRESULT GatewayIP([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "DMZ Gateway IP is: " & oGW.GatewayIP
```

Specifying:

```
oGW.GatewayIP = "198.162.0.1"
```

EFT v6.3 and later

- Property **GatewayPort** As Long

Retrieves or specifies the port on which the DMZ Gateway communicates with EFT.

```
HRESULT GatewayPort([out,retval] LONG* pVal);
HRESULT GatewayPort([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "DMZ Gateway port is: " & CStr(oGW.GatewayPort)
```

Specifying:

```
oGW.GatewayPort = 44500
```

EFT v6.3 and later

- Property **HTTPPort** As Long

Retrieves or specifies the HTTP port.

```
HRESULT HTTPPort([out,retval] LONG* pVal);
HRESULT HTTPPort([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "HTTP port for external connections: " & CStr
(oGW.HTTP)
```

Specifying:

```
oGW.HTTP = 80
```

EFT v6.3 and later

- Property **HTTPSPort** As Long

Retrieves or specifies the HTTPS port.

```
HRESULT HTTPSPort([out,retval] LONG* pVal);
HRESULT HTTPSPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "HTTPS port for external connections: " & CStr
(oGW.HTTPS)
```

Specifying:

```
oGW.HTTPS = 443
```

EFT v6.3 and later

- Property **PASVListenIP** As Long

Retrieves or specifies the PASV IP address.

```
HRESULT PASVListenIP([out,retval] BSTR* pVal);
HRESULT PASVListenIP([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Passive mode listening IP: " & oGW.PASVListenIP
```

Specifying:

```
oGW.PASVListenIP = "192.168.0.88"
```

EFT v6.3 and later

- Property **PASVPortMax** As Long

Retrieves or specifies the upper end of the PASV port range.

```
HRESULT PASVPortMax([out,retval] LONG* pVal);
HRESULT PASVPortMax([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Lower bound of port range is: " & CStr
(oGW.PASVPortMax)
```

Specifying:

```
oGW.PASVPortMax = 30000
```

EFT v6.3 and later

- Property **PASVPortMin** As Long

Retrieves or specifies the lower end of the PASV port range.

```
HRESULT PASVPortMin([out,retval] LONG* pVal);
HRESULT PASVPortMin([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Lower bound of port range is: " & CStr
(oGW.PASVPortMin)
```

Specifying:

```
oGW.PASVPortMin = 28000
```

EFT v6.3 and later

- Property **RemoteAdminMX** As Boolean

```
HRESULT RemoteAdminMX([out, retval] VARIANT_BOOL* pVal);
HRESULT RemoteAdminMX([in] VARIANT_BOOL newVal);
```

EFT v7.4.7 and later

- Property **RemoteAdminMXPort** As Long

```
HRESULT RemoteAdminMXPort([out, retval] VARIANT_BOOL* pVal);
HRESULT RemoteAdminMXPort([in] VARIANT_BOOL newVal);
```

EFT v7.4.7 and later

- Property **RouteFTP** As Boolean

Retrieves or specifies whether to enable FTP.

```
HRESULT RouteFTP([out,retval] VARIANT_BOOL* pVal);
HRESULT RouteFTP([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
if True = oGW.RouteFTP then
  MsgBox "FTP is routing."
end if
```

Specifying:

```
oGW.RouteFTP = False
```

EFT v6.3 and later

- Property **RouteFTPSExp1** As Boolean

Retrieves or specifies whether to enable FTPS explicit.

```
HRESULT RouteFTPSExp1([out,retval] VARIANT_BOOL* pVal);
HRESULT RouteFTPSExp1([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
if True = oGW.RouteFTPSExp1 then
  MsgBox "FTPS explicit is routing."
end if
```

Specifying:

```
oGW.RouteFTPSExp1 = False
```

EFT v6.3 and later

- Property **RouteFTPSImpl** As Boolean

Retrieves or specifies whether to enable FTPS implicit.

```
HRESULT RouteFTPSImpl([out,retval] VARIANT_BOOL* pVal);
HRESULT RouteFTPSImpl([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
if True = oGW.RouteFTPSImpl then
MsgBox "FTPS implicit is routing."
end if
```

Specifying:

```
oGW.RouteFTPSImpl = False
```

EFT v6.3 and later

- Property **RouteHTTP** As Boolean

Retrieves or specifies whether to enable HTTP.

```
HRESULT RouteHTTP([out,retval] VARIANT_BOOL* pVal);
HRESULT RouteHTTP([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
if True = oGW.RouteHTTP then
MsgBox "HTTP is routing."
end if
```

Specifying:

```
oGW.RouteHTTP = False
```

EFT v6.3 and later

- Property **RouteHTTPS** As Boolean

Retrieves or specifies whether to enable HTTPS.

```
HRESULT RouteHTTPS([out,retval] VARIANT_BOOL* pVal);
HRESULT RouteHTTPS([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
if True = oGW.RouteHTTPS then
  MsgBox "HTTPS is routing."
end if
```

Specifying:

```
oGW.RouteHTTPS = False
```

EFT v6.3 and later

- Property **RouteSFTP** As Boolean

Retrieves or specifies whether to enable SFTP.

```
HRESULT RouteSFTP([out,retval] VARIANT_BOOL* pVal);
HRESULT RouteSFTP([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
if True = oGW.RouteSFTP then
  MsgBox "SFTP is routing."
end if
```

Specifying:

```
oGW.RouteSFTP = False
```

EFT v6.3 and later

- Property **SecurePNC** As Boolean

Retrieves or specifies whether Secure PNC is enabled.

```
HRESULT SecurePNC([out, retval] VARIANT_BOOL* pVal);
HRESULT SecurePNC([in] VARIANT_BOOL newVal);
```

EFT v7.4.6 and later

- Property **SFTPPort** As Long

Retrieves or specifies the SFTP port.

```
HRESULT SFTPPort([out,retval] LONG* pVal);
HRESULT SFTPPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP port for external connections: " & CStr
(OGW.SFTP)
```

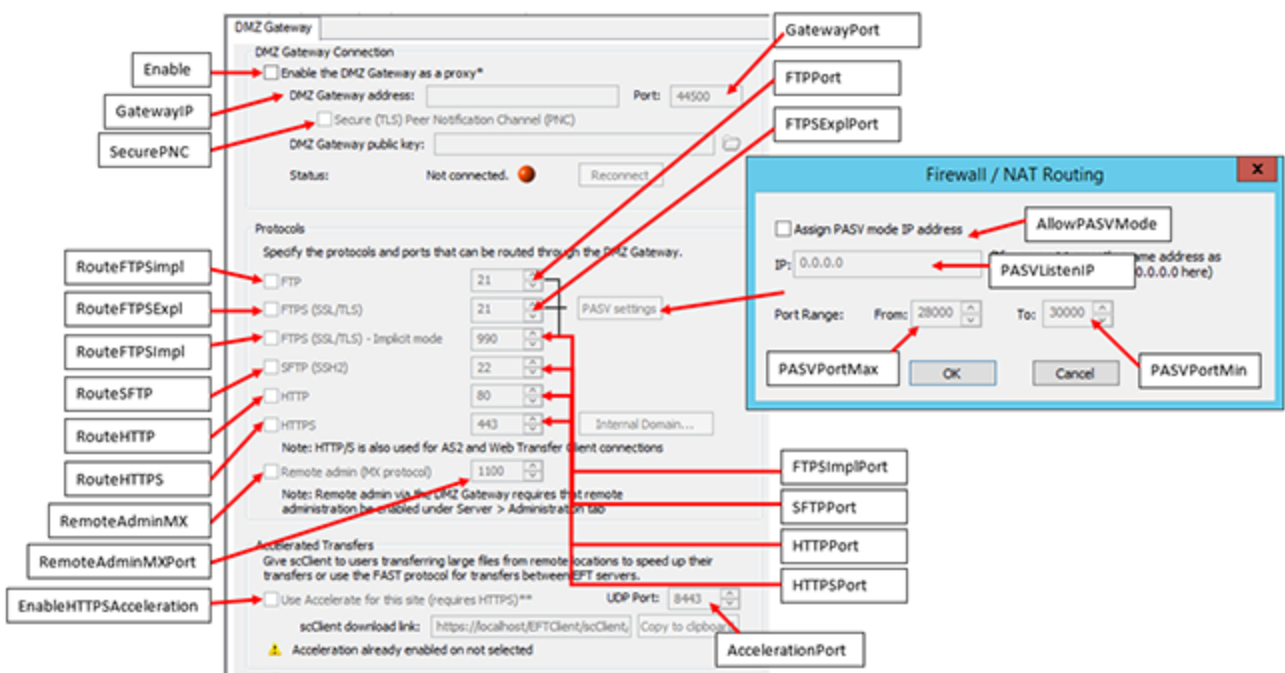
Specifying:

```
OGW.SFTP = 22
```

EFT v6.3 and later

Dialog Box Equivalents

The ICIDMZGateway interface properties correlate to the following fields and controls in the EFT administration interface:



Event Rule-Related Interfaces

This chapter contains Event Rule-related COM objects.

ICIEventRule Interface

Use the ICIEventRule interface to make changes to the Event Rule settings.

```
interface ICIEventRule : IDispatch
```

Examples (VBScript):

```
'Add "If File name matches "*.txt" or "*.exe" condition:
cond = Rule.AddIfStatement(0, FileName, Match, Array("*.txt",
"*.exe"))
Set Params = Rule.GetParams
'rename rule
Set Params = Rule.GetParams
Params.Name = Params.Name & "_renamed"
Rule.SetParams Params
```

NOTE: "Function" and "Sub" indicate that the member is a method.

See also [ICIEventRules Interface](#), [ICIEventAction Interface](#), [ICIEventActions Interface](#), [ICIEventInfo Interface](#), [ICIEventRuleParams Interface](#).

- Function **AddActionStatement**(lIndex As Long, pdispParams As Object) As Object
For an example using this method, refer to the EventRuleCreation script in [Script Examples](#).

Adds an Action statement at a specified position.

```
HRESULT AddActionStatement([in] long lIndex,
[in] IDispatch* pdispParams,
[out, retval] IDispatch** ppdispActionStatement);
```

Example (C#):

```
rule.AddActionStatement(rule.StatementsCount(),
eventLogActionParams);
```

EFT v5.2 and later

- Function **AddIfStatement**(lIndex As Long, property As [EventProperty](#), op As [ConditionOperator](#), varConditionValue, bNot As Boolean) As Object

Adds an If-statement at a specified position.

```
HRESULT AddIfStatement([in] long lIndex,
[in] EventProperty property,
[in] ConditionOperator op,
[in] VARIANT varConditionValue,
[in] VARIANT_BOOL bNot,
[out, retval] IDispatch** ppdispIfStatement);
```

EFT v5.2 and later

- Function **AddLoopDatasetStatement**(lIndex As Long, pdispParams As Object) As Object

Used to define a Loop through Dataset action in an Event Rule.

```
HRESULT AddLoopDatasetStatement(
[in] long lIndex,
[in] IDispatch* pdispParams,
[out, retval] IDispatch** ppdispLoopDatasetStatement);
```

EFT v8.0.4 and later

- Property **ChangeDescription** As String

```
HRESULT ChangeDescription([in] BSTR newVal);
```

EFT v7.2

- Sub **DeleteStatement**(lIndex As Long)

Delete Event Rule's statement by index.

```
HRESULT DeleteStatement([in] long lIndex);
```

EFT v5.2 and later

- Sub **Export**(bstrDstFile As String)

```
HRESULT Export([in] BSTR bstrDstFile);
```

EFT v7.2

- Function **GetACL**() As Object and Sub **SetACL**(pVal As Object)

```
HRESULT GetACL([out, retval] IDispatch** ppVal);
HRESULT SetACL([in] IDispatch* pVal);
```

EFT v7.2 and later

- Function **GetHASchedulingParams()** As Object and Sub **SetHASchedulingParams**(newVal As Object)

Returns/specifies an object of type [ICIRoundRobinEventRuleSchedulingParams](#) that can be used to manage the nodes listed in the “Run on One of” list for a load-balanced event rule.

```
HRESULT GetHASchedulingParams([out, retval] IDispatch** ppdisp);
HRESULT SetHASchedulingParams([in] IDispatch* newVal);
```

EFT v7.0 and later

- Function **GetParams()** As Object and Sub **SetParams**(newVal As Object)

Retrieves/specifies Event Rule’s parameters. The actual object depends on the Event Rule’s type. (Refer to [ICIEventRuleParams](#), [ICITimerEventRuleParams](#), [ICIFolderMonitorEventRuleParams](#) interfaces.)

```
HRESULT GetParams([out, retval] IDispatch* *pVal);
HRESULT SetParams([in] IDispatch* newVal);
```

EFT v6.2 and later

- Function **Statement**(lIndex As Long) As Object

Retrieves a statement object by index. (Refer to [ICIEventRuleStatement](#) interface.)

```
HRESULT Statement([in] long lIndex,
[out, retval] IDispatch** ppdispStatement);
```

EFT v5.2 and later

- Function **StatementsCount()** As Long

For an example using this method, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves the number of listed statements.

```
HRESULT StatementsCount([out, retval] long* plCount);
```

Example (C#):

```
rule.AddActionStatement(rule.StatementsCount(),
    eventLogActionParams);
```

EFT v5.2 and later

- [ICIEventAction - Event Action Interface](#)
- [ICIEventActions - Event Actions Interface](#)
- [ICIEventInfo - Event Information Interface](#)
- [ICIEventRuleParams - Event Rule Parameters Interface](#)
- [ICIEventRules - Event Rules Interface](#)
- [ICIEventRuleStatement - Event Rule Statement Interface](#)
- [ICIEventRuleFolder](#)
- [ICIEventRuleFolders](#)
- [ICIOBJECTAccessRights](#)
- [ICIOBJECTACE](#)
- [ICIOBJECTACL](#)

ICIEventRules Interface

The ICIEventRules interface allows you to make changes to the Event Rule settings.

```
interface ICIEventRules : IDispatch
```

NOTE: "Function" and "Sub" indicate that the member is a method.

Methods

- Function **Add**(lIndex As Long, pdispParams As Object) As Object

Add an Event Rule at a specified position.

```
HRESULT Add(
    [in]long lIndex,
    [in] IDispatch* pdispParams,
    [out, retval] IDispatch** ppdispNewRule);
```

EFT v5.2 and later

- Function **Count**() As Long

Retrieves the number of Event Rules.

```
HRESULT Count([out, retval] long* plCount);
```

EFT v5.2 and later

- Sub **Delete**(lIndex As Long)

Deletes an Event Rule by index.

```
HRESULT Delete([in] long lIndex);
```

EFT v5.2 and later

- Function **Find**(strName As String) As Object

Find an Event Rule by name.

```
HRESULT Find([in] BSTR strName, [out, retval] IDispatch  
**ppdisp);
```

EFT v5.2 and later

- Function **Item**(lIndex As Long) As Object

Retrieves or specifies an Event Rule object by index. (Refer to [ICIEventRule](#) interface.)

```
HRESULT Item([in] long lIndex, [out, retval] IDispatch  
**ppdispRule);
```

EFT v5.2 and later

- [ICIEventAction - Event Action Interface](#)
- [ICIEventActions - Event Actions Interface](#)
- [ICIEventInfo - Event Information Interface](#)
- [ICIEventRuleParams - Event Rule Parameters Interface](#)
- [ICIEventRule - Event Rule Interface](#)
- [ICIEventRuleStatement - Event Rule Statement Interface](#)

ICIEventAction Interface

The ICIEventAction interface allows you to make changes to the Event Rule Action settings.

```
interface ICIEventAction : IDispatch
```

NOTE: "Function" and "Sub" indicate that the member is a method; "Property" indicates that it is a property.

Members

- Function **GetParams()** As Object
Retrieves the event action parameters.

```
HRESULT GetParams([out, retval] IDispatch** ppdispParams);
```

EFT v6.1 and later

- Sub **SetParams**(pdispParams As Object)
Specifies the event action parameters.

```
HRESULT SetParams([in] IDispatch* pdispParams);
```

EFT v6.1 and later

- Property **type** As [EventActionType](#) read-only

```
HRESULT Type([out, retval] EventActionType *pVal);
```

EFT v5.2 and later

- [ICIEventActions - Event Actions Interface](#)
- [ICIEventInfo - Event Information Interface](#)
- [ICIEventRuleParams - Event Rule Parameters Interface](#)
- [ICIEventRule - Event Rule Interface](#)
- [ICIEventRules - Event Rules Interface](#)
- [ICIEventRuleStatement - Event Rule Statement Interface](#)

ICIEventActions Interface

The ICIEventActions interface allows you to make changes to the Event Rule Action settings.

```
interface ICIEventActions : IDispatch
```

NOTE: "Function" and "Sub" indicate that the member is a method.

Members

- Function **Add**(lIndex As Long, pdispParams As Object) As Object
Adds an event action at specified position.

```
HRESULT Add(
    [in] long lIndex,
    [in] IDispatch* pdispParams,
    [out, retval] IDispatch** ppdispAction);
```

EFT v5.2 and later

- Function **Count()** As Long

Retrieves a number of listed event actions.

```
HRESULT Count([out, retval] long* plCount);
```

EFT v5.2 and later

- Sub **Delete**(lIndex As Long)

Deletes event action by index.

```
HRESULT Delete([in] long lIndex);
```

EFT v5.2 and later

- Function **Item**(lIndex As Long) As Object

Retrieves or specifies an event action object by index. (Refer to [ICIEventAction](#) interface.)

```
HRESULT Item(
    [in] long lIndex,
    [out, retval] IDispatch** ppdispAction);
```

EFT v5.2 and later

- [ICIEventAction - Event Action Interface](#)
- [ICIEventInfo - Event Information Interface](#)
- [ICIEventRuleParams - Event Rule Parameters Interface](#)
- [ICIEventRule - Event Rule Interface](#)
- [ICIEventRules - Event Rules Interface](#)
- [ICIEventRuleStatement - Event Rule Statement Interface](#)

ICIEventInfo Interface

The ICIEventInfo interface is used to retrieve information about EFT's events. (Also refer to [ICIServer::AvailableEvents](#) property.) The interface provides the set of read-only properties.

```
interface ICIEventInfo : IDispatch
```

Properties

- Property **Name** As String read-only

Specifies the name of the event.

```
HRESULT Name([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Event: " & Event.Name
```

EFT v6.1 and later

- Property **type** As [EventType](#) read-only

Specifies the type of event.

```
HRESULT Type([out, retval] EventType* pVal);
```

Example (VBScript):

```
If Event.Type = OnTimer Then
MsgBox "Event type: Timer"
End If
```

EFT v6.1 and later

- [ICIEventAction - Event Action Interface](#)
- [ICIEventInfo - Event Information Interface](#)
- [ICIEventRuleParams - Event Rule Parameters Interface](#)
- [ICIEventRule - Event Rule Interface](#)
- [ICIEventRules - Event Rules Interface](#)
- [ICIEventRuleStatement - Event Rule Statement Interface](#)

ICIEventRuleParams Interface

Use the ICIEventRuleParams interface to make changes to the Event Rule Condition settings.

```
interface ICIEventRuleParams : IDispatch
```


Members

- Property **Description** As String

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the description of the Event Rule.

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

Example (C#):

```
objParams.Description = "This is a test event rule";
```

EFT v5.2 and later

- Property **Enabled** As Boolean

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies whether the Event Rule is enabled.

```
HRESULT Enabled([out, retval] VARIANT_BOOL *pVal);
HRESULT Enabled([in] VARIANT_BOOL newVal);
```

Example (C#):

```
objParams.Enabled = true;
```

EFT v5.2 and later

- Property **Name** As String

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the name of the Event Rule.

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

Example (C#):

```
objParams.Name = "TestEventRule";
```

EFT v5.2 and later

- [ICIEventAction - Event Action Interface](#)
- [ICIEventActions - Event Actions Interface](#)
- [ICIEventInfo - Event Information Interface](#)
- [ICIEventRule - Event Rule Interface](#)
- [ICIEventRules - Event Rules Interface](#)
- [ICIEventRuleStatement - Event Rule Statement Interface](#)

ICIEventRuleFolder

The Event Rule Folder interface is used to manage Event Rule folders in the Event Rules node.

- Sub **Add**(lIndex As Long, strRuleName As String)

```
HRESULT Add([in] long lIndex, [in] BSTR strRuleName);
```

EFT v7.2

- Function **Count**() As Long

```
HRESULT Count([out, retval] long* plCount);
```

EFT v7.2

- Sub **Delete**(lIndex As Long)

```
HRESULT Delete([in] long lIndex);
```

EFT v7.2

- Sub **Export**(bstrDstFile As String)

```
HRESULT Export([in] BSTR bstrDstFile);
```

EFT v7.2

- Function **Find**(strRuleName As String) As Boolean

```
HRESULT Find([in] BSTR strRuleName, [out, retval] VARIANT_BOOL* pbResult);
```

EFT v7.2

- Function **GetACL**() As Object

```
HRESULT GetACL([out, retval] IDispatch** ppVal);
```

EFT v7.2

- Sub **Import**(bstrSrcFile As String)

```
HRESULT Import([in] BSTR bstrSrcFile);
```

EFT v7.2

- Function **Item**(lIndex As Long) As String

```
HRESULT Item([in] long lIndex, [out, retval] BSTR*  
pstrRuleName);
```

EFT v7.2

- Property **Name** As String

```
HRESULT Name([out, retval] BSTR *pVal);  
HRESULT Name([in] BSTR newVal);
```

EFT v7.2

- Sub **SetACL**(pVal As Object)

```
HRESULT SetACL([in] IDispatch* pVal);
```

EFT v7.2

ICIEventRuleFolders

The Event Rule Folders interface is used to define Event Rule folders to contain Event Rules.

- Sub **Add**(lIndex As Long, strName As String)

```
HRESULT Add([in] long lIndex, [in] BSTR strName);
```

EFT v7.2 and later

- Function **Count**() As Long

```
HRESULT Count([out, retval] long* plCount);
```

EFT v7.2 and later

- Sub **Delete**(lIndex As Long, bForce As Boolean)

```
HRESULT Delete([in] long lIndex, [in] VARIANT_BOOL bForce);
```

EFT v7.2 and later

- Function **Find**(strName As String) As Object

```
HRESULT Find([in] BSTR strName, [out, retval] IDispatch
**ppdisp);
```

EFT v7.2 and later

- Function **Item**(lIndex As Long) As Object

```
HRESULT Item([in] long lIndex, [out, retval] IDispatch
**ppdispRule);
```

EFT v7.2 and later

ICIEventRuleStatement Interface

Use the ICIEventRuleStatement interface to make changes to the Event Rule settings.

```
interface ICIEventRuleStatement : IDispatch
```

Property

- Property **type**

Retrieves the Event Rule Statement type.

```
HRESULT Type([out, retval] EventRuleStatementType *pVal);
```

EventRuleStatementType

- ActionStatement = 0
- IfStatement = 1

EFT v6.0 and later

- [ICIEventAction Interface](#)
- [ICIEventActions Interface](#)
- [ICIEventInfo Interface](#)
- [ICIEventRuleParams Interface](#)
- [ICIEventRule Interface](#)
- [ICIEventRules Interface](#)

ICIActionStatement Interface

The ICIActionStatement interface allows you to make changes to the Event Rule settings. (See also [ICIBackupActionParams - Backup Action Parameters.](#))

```
interface ICIActionStatement : ICIEventRuleStatement
```

Example (VBScript):

```
If ActionStatement.Action.Type = BackupAction Then
Set BackupParams = ActionStatement.Action.Params
' Modify backup parameters and assign them back to the action
' . . .
ActionStatement.Action.Params = BackupParams
End If
```

NOTE: "Function" indicates that the member is a method; "Property" indicates that it is a property.

- Function **Action()** As Object

Retrieves or specifies the action.

```
HRESULT Action([out, retval] IDispatch** ppdispAction);
```

EFT v5.2 and later

- Function **FailSection()** As Object

Retrieves or specifies what occurs if the Event Rule fails.

```
HRESULT FailSection([out, retval] IDispatch**
ppdispActions);
```

EFT v5.2 and later

- Property **type** As [EventRuleStatementType](#)

(read-only); Retrieves or specifies the type of Event Rule statement (action or conditional).

EFT v5.2 and later

ICIActionStatements Interface

The ICIActionStatements interface allows you to make changes to the Event Rule settings.

```
interface ICIActionStatements : IDispatch
```

NOTE: "Function" and "Sub" indicate that the member is a method.

Methods

- Function **Add**(lIndex As Long, pdispActionParams As Object) As Object
Adds action statement at specified position.

```
HRESULT Add(  
[in] long lIndex,  
[in] IDispatch* pdispActionParams,  
[out, retval] IDispatch** ppdispActionStatement);
```

EFT v5.2 and later

- Function **AddActionStatement**(lIndex As Long, pdispActionParams As Object) As Object

```
HRESULT AddActionStatement(  
[in] long lIndex,  
[in] IDispatch* pdispActionParams,  
[out, retval] IDispatch** ppdispActionStatement);
```

EFT v8.0.4

- Function **AddIfStatement**(lIndex As Long, property As EventProperty, op As ConditionOperator, varConditionValue, bNot As Boolean) As Object

```
HRESULT AddIfStatement(  
[in] long lIndex,  
[in] EventProperty property,  
[in] ConditionOperator op,  
[in] VARIANT varConditionValue,  
[in] VARIANT_BOOL bNot,  
[out, retval] IDispatch** ppdispIfStatement);
```

EFT v8.0.4

- Function **AddLoopDatasetStatement**(lIndex As Long, pdispParams As Object) As Object

```
HRESULT AddLoopDatasetStatement(  
[in] long lIndex,
```

```
[in] IDispatch* pdispParams,  
[out, retval] IDispatch** ppdispLoopDatasetStatement);
```

EFT v8.0.4

- Function **Count()** As Long

Retrieves a number of listed action statements.

```
HRESULT Count([out, retval] long* plCount);
```

EFT v5.2 and later

- Sub **Delete**(lIndex As Long)

Deletes action statement by index.

```
HRESULT Delete([in] long lIndex);
```

EFT v5.2 and later

- Function **Item**(lIndex As Long) As Object

Retrieves or specifies an action statement object by index. (Refer to [ICIActionStatement](#) interface.)

```
HRESULT Item([in] long lIndex,[out, retval] IDispatch**  
ppdispActionStatement);
```

EFT v5.2 and later

IAdhocMessage Interface

This interface is used as an input parameter to [ICIAudit](#) to log ad hoc transfer activity from Mail Express to record "ad hoc" protocol activity and trigger file-upload/download Event Rules. See also [ISATMessage](#) interface.

- Property **BytesTransferred** As Signed Double Long

Retrieves the number of bytes transferred.

```
HRESULT BytesTransferred([in] LONGLONG newVal);  
HRESULT BytesTransferred([out, retval] LONGLONG *pVal);
```

EFT v6.5 - 7.4.x

- Property **DisplayName** As String

Retrieves the display name of the agent responsible for the transfer.

```
HRESULT DisplayName([in] BSTR newVal);
HRESULT DisplayName([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **EmailAddress** As String

Retrieves the email address of the agent responsible for the transfer.

```
HRESULT EmailAddress([in] BSTR newVal);
HRESULT EmailAddress([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **ExternalRef** As String

Retrieves the reference identifier within the ad hoc database.

```
HRESULT ExternalRef([in] BSTR newVal);
HRESULT ExternalRef([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **FileSize** As Signed Double Long

Retrieves the size of the item transferred.

```
HRESULT FileSize([in] LONGLONG newVal);
HRESULT FileSize([out, retval] LONGLONG *pVal);
```

EFT v6.5 - 7.4.x

- Property **IsDispatchEvent** As Boolean

Retrieves whether an event should be dispatched for this transfer.

```
HRESULT IsDispatchEvent([in] VARIANT_BOOL newVal);
HRESULT IsDispatchEvent([out, retval] VARIANT_BOOL *pVal);
```

EFT v6.5 - 7.4.x

- Property **IsUpload** As Boolean

Retrieves the whether the adhoc transfer was inbound or outbound.


```
HRESULT IsUpload([in] VARIANT_BOOL newVal);
HRESULT IsUpload([out, retval] VARIANT_BOOL *pVal);
```

EFT v6.5 - 7.4.x

- Property **LocalAddr** As Boolean

Retrieves the address of the service facilitating the transfer.

```
HRESULT LocalAddr([in] BSTR newVal);
HRESULT LocalAddr([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **LocalPort** As Int

Retrieves the port of the service facilitating the transfer.

```
HRESULT LocalPort([in] INT newVal);
HRESULT LocalPort([out, retval] INT *pVal);
```

EFT v6.5 - 7.4.x

- Property **PhysicalDirectory** As String

Retrieves the physical directory of the item transferred.

```
HRESULT PhysicalDirectory([in] BSTR newVal);
HRESULT PhysicalDirectory([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **PhysicalFileName** As String

Retrieves the physical filename of the item transferred.

```
HRESULT PhysicalFileName([in] BSTR newVal);
HRESULT PhysicalFileName([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **PhysicalPath** As String

Retrieves the physical path of the item transferred.

```
HRESULT PhysicalPath([in] BSTR newVal);
HRESULT PhysicalPath([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **RemoteAddr** As String

Retrieves the address of the agent responsible for the transfer.

```
HRESULT RemoteAddr([in] BSTR newVal);
HRESULT RemoteAddr([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **RemotePort** As Int

Retrieves the port of the agent responsible for the transfer.

```
HRESULT RemotePort([in] INT newVal);
HRESULT RemotePort([out, retval] INT *pVal);
```

EFT v6.5 - 7.4.x

- Property **SiteName** As String

Retrieves the Site on which the ad hoc transfer occurred.

```
HRESULT SiteName([in] BSTR newVal);
HRESULT SiteName([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **Timestamp** As Date

Retrieves the timestamp of the ad hoc transfer.

```
HRESULT Timestamp([in] DATE newVal);
HRESULT Timestamp([out, retval] DATE *pVal);
```

EFT v6.5 - 7.4.x

- Property **UserName** As String

Retrieves the username of the agent responsible for the transfer.

```
HRESULT UserName([in] BSTR newVal);
HRESULT UserName([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **VirtualDirectory** As String

Retrieves the virtual directory of the item transferred.

```
HRESULT VirtualDirectory([in] BSTR newVal);
HRESULT VirtualDirectory([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **VirtualFileName** As String

Retrieves the virtual filename of the item transferred.

```
HRESULT VirtualFileName([in] BSTR newVal);
HRESULT VirtualFileName([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

- Property **VirtualPath** As String

Retrieves the virtual path of the item transferred.

```
HRESULT VirtualPath([in] BSTR newVal);
HRESULT VirtualPath([out, retval] BSTR *pVal);
```

EFT v6.5 - 7.4.x

ICIAudit Interface

Use the ICIAudit interface to log audit information related to ad hoc transfers. (Use the [ICIServer : Auditor](#) method to retrieve the ICIAudit interface.)

```
interface ICIAudit : IDispatch
```

NOTE: "Sub" indicates that the member is a method.

Methods

- Sub **AuditAdhoc**(msg As [AdhocMessage](#))

Audit/event-process an ad hoc file transfer.

```
HRESULT AuditAdhoc([in] IAdhocMessage *msg);
```

EFT v6.5 and later

- Sub **AuditSAT**(msg As SATMessage)

Audit a new SAT transaction through ARM. (Called by Mail Express; similar to **AuditSATTransaction**, but also provides an explicit timestamp for when the event occurs.)

```
HRESULT AuditSAT([in] ISATMessage *msg);
```

EFT v6.5 and later

- Sub **AuditSATEmail**(Email As String, EmailType As Long, TransGUID As String)

Auditing a SAT e-mail in ARM.

```
HRESULT AuditSATEmail(
[in] BSTR Email,
[in] LONG EmailType,
[in] BSTR TransGUID);
```

EFT Server v6.0 and later

- Sub **AuditSATFile**(FileName As String, FileSize As Signed Double Long, TransGUID As String)

Auditing a SAT file in ARM.

```
HRESULT AuditSATFile(
[in] BSTR FileName,
[in] LONGLONG FileSize,
[in] BSTR TransGUID);
```

EFT Server v6.0 and later

- Sub **AuditSATTransaction**(From As String, Subject As String, Body As String, TempUserName As String, TempPassword As String, SiteName As String, ExpireDays As Long, TransGUID As String, Reserved1 As String, Reserved2 As String)

Auditing a new SAT transaction through ARM (called by SAT module).

```
HRESULT AuditSATTransaction([in] BSTR From, [in] BSTR
Subject,
[in] BSTR Body,
[in] BSTR TempUserName,
[in] BSTR TempPassword,
[in] BSTR SiteName,
[in] LONG ExpireDays,
[in] BSTR TransGUID,
```

```
[in] BSTR Reserved1,
[in] BSTR Reserved2);
```

EFT Server v6.0 and later

ICIBackupActionParams Interface

Use ICIBackupActionParams::Folder to create, modify, and retrieve the Backup Server Configuration Event Rule Action. (See also [ICIActionStatement Interface](#).)

```
interface ICIBackupActionParams : IDispatch
```

Members

- Property **Folder** As String

Retrieve or specify the path to the folder to store the backup archive.

```
HRESULT Folder([out, retval] BSTR* pVal);
HRESULT Folder([in] BSTR newVal);
```

EFT v6.1 and later

Examples (VBScript):

```
If ActionStatement.Action.Type = BackupAction Then
Set BackupParams = ActionStatement.Action.Params
\ Modify backup parameters and assign them back to the action
\ . . .
ActionStatement.Action.Params = BackupParams
End If
```

The CIBackupActionParams co-class supports the creation of the Backup Server Configuration Event Rule Action. The co-class implements ICIBackupActionParams interface.

```
Set BackupParams = CreateObject
("SFTPCOMInterface.CIBackupActionParams")
\ Setup backup parameters
\ . . .
Rule.AddActionStatement(0, BackupParams)
```

Retrieving:

```
MsgBox "Configuration snapshot will be stored in " &
BackupParams.Folder
```

Specifying:

```
BackupParams.Folder = "C:\Temp"
```

ICICallSubroutineActionParams Interface

The ICICallSubroutineActionParams interface is used in EFT Event Rules.

Members

- Property SubroutineName As String

```
HRESULT SubroutineName([out, retval] BSTR *pVal);
HRESULT SubroutineName([in] BSTR newVal);
```

EFT v8 and later

- Property Variables As Variant

```
HRESULT Variables([out, retval] VARIANT *pVal);
HRESULT Variables([in] VARIANT newVal);
```

EFT v8 and later

- Property WaitUntilCompletes As Boolean

Specify whether the Event Rule should wait for this action to complete.

```
HRESULT WaitUntilCompletes([out, retval] VARIANT_BOOL
*pVal);
HRESULT WaitUntilCompletes([in] VARIANT_BOOL newVal);
```

EFT v8.0.5

ICICleanupActionParams Interface

In EFT Enterprise, you can configure an EFT Event Rule to clean up a specified folder at regularly scheduled intervals. The Cleanup in Folder Action is available only with the On Timer Server event. At the interval that you specify, EFT compares the filter parameters of the Cleanup Action to the files in the designated folder, then determines the creation time of the file and deletes ("cleans up") files that match the cleanup parameters. For example, if you specify to cleanup files that are older than 7 days named dailyreport*.doc in the folder D:\WorkFolder\Sales\Daily Reports, any Microsoft Word files in that folder with daily report in the file name are deleted.

```
interface ICICleanupActionParams : IDispatch
```

Members

- Property **DaysToKeepFiles** As Long

Retrieves or specifies the number of days to keep files prior to cleanup.

```
HRESULT DaysToKeepFiles([out, retval] long *pVal);
HRESULT DaysToKeepFiles([in] long newVal);
```

Example:

```
cleanupParams.DaysToKeepFiles = 14 'Remove files older than
2 weeks
```

EFT v6.0 and later

- Property **ExcludeFileMask** As Boolean

Retrieves or specifies whether the **FileMask** property specifies files to remove or exclude from cleanup (e.g., remove all the files except for those matching **FileMask**).

```
HRESULT ExcludeFileMask([out, retval] VARIANT_BOOL *pVal);
HRESULT ExcludeFileMask([in] VARIANT_BOOL newVal);
```

Example:

```
cleanupParams.ExcludeFileMask = False 'Remove only files
matching FileMask during cleanup
```

True = Include; **False** = Exclude

EFT v6.0 and later

- Property **FileMask** As String

Retrieves or specifies the files to remove or keep, depending on **ExcludeFileMask** property value.

```
HRESULT FileMask([out, retval] BSTR *pVal);
HRESULT FileMask([in] BSTR newVal);
```

Example:

```
cleanupParams.FileMask = "*.txt"
```

EFT v6.0 and later

- Property **FileTimestampType** As [CleanupTimestampType](#)

```
HRESULT FileTimestampType([out, retval] CleanupTimestampType
*pVal);
HRESULT FileTimestampType([in] CleanupTimestampType newVal);
```

EFT v7.4.13

- Property **Folder** As String

Retrieves or specifies the folder to cleanup.

```
HRESULT Folder([out, retval] BSTR *pVal);
HRESULT Folder([in] BSTR newVal);
```

Example:

```
cleanupParams.Folder = "C:\EFT\Folder_to_cleanup"
```

EFT v6.0 and later

- Property **Recursive** As Boolean

Retrieves or specifies whether to cleanup all subfolders recursively.

```
HRESULT Recursive([out, retval] VARIANT_BOOL *pVal);
HRESULT Recursive([in] VARIANT_BOOL newVal)
```

Example:

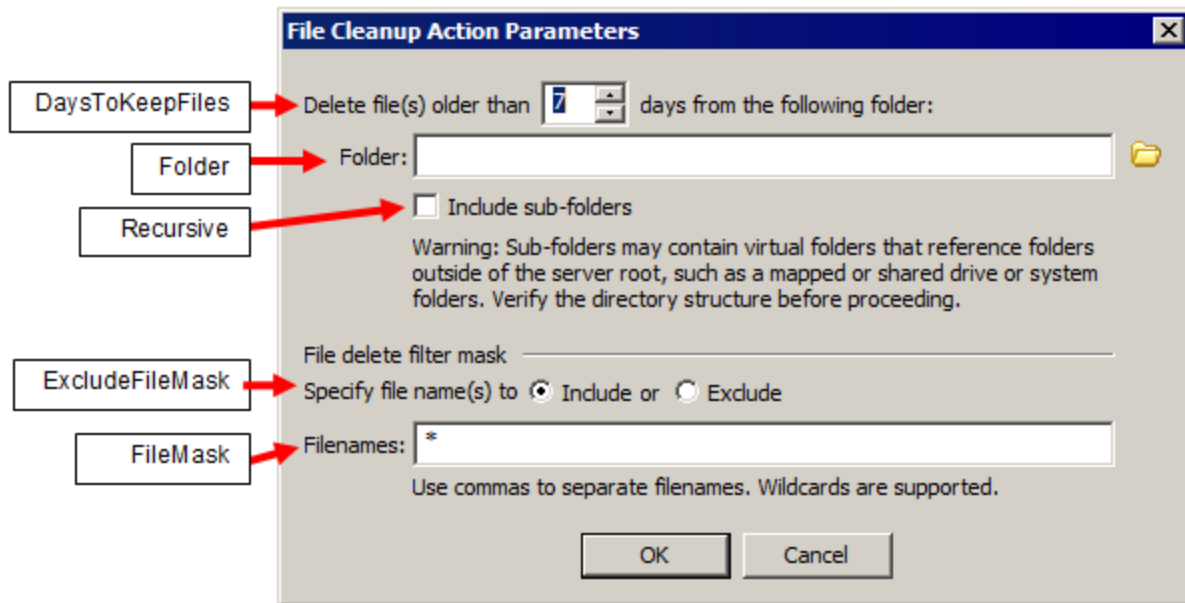
```
cleanupParams.Recursive = False
```

True = Include; **False** = Exclude

EFT v6.0 and later

Dialog Box Equivalents

The ICICleanupActionParams interface properties correlate to the following fields and controls in the EFT administration interface:



IC ICommandActionParams Interface

The IC ICommandActionParams interface allows you to make changes to the settings of Execute Command Event Action. (Access the IC ICommandActionParams interface with [ICIEventAction](#) when its Type property returns CommandAction).

```
interface IC ICommandActionParams
: IDispatch
```

Example:

```
If action.Type = 1 then 'CommandAction
Set commandParams = action.Params
EndIf
```

Members

- Property **Command** As String

Retrieves or specifies command to execute.

```
HRESULT Command([out, retval] BSTR *pVal);
HRESULT Command([in] BSTR newVal);
```

Example:

```
commandParams.Command = "My_command"
```

EFT v5.2 and later

- Property **Parameters** As String

Retrieves or specifies parameters required to execute command.

```
HRESULT Parameters([out, retval] BSTR *pVal);
HRESULT Parameters([in] BSTR newVal);
```

Example:

```
commandParams.Parameters = "D:\MyFolder\MyScript.vbs -A -B"
```

EFT v5.2 and later

- Property **WaitUntilCompletes** As Boolean

Specify whether the Event Rule should wait for this action to complete.

```
HRESULT WaitUntilCompletes([out, retval] VARIANT_BOOL
*pVal);
HRESULT WaitUntilCompletes([in] VARIANT_BOOL newVal);
```

EFT v8.0.5

- Property **WorkingFolder** As String

Retrieves or specifies working directory, if different from executable path.

```
HRESULT WorkingFolder([out, retval] BSTR *pVal);
HRESULT WorkingFolder([in] BSTR newVal);
```

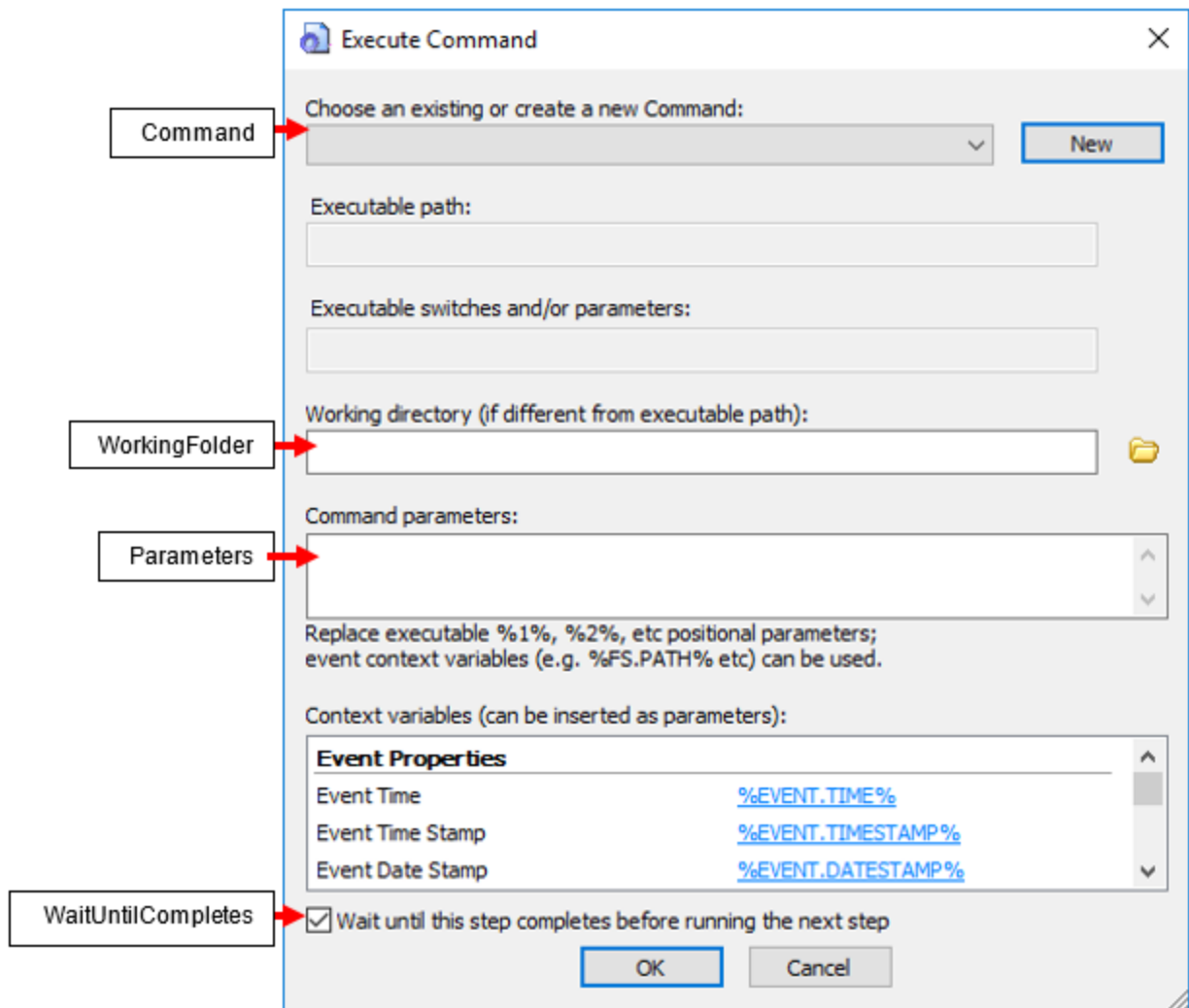
Example:

```
commandParams.WorkingFolder = "D:\MyFolder"
```

EFT v5.2 and later

Dialog Box Equivalents

The dialog box below is labeled with the properties that are available via the API.



IC ICommandSettings Interface

Commands can execute programs, scripts, or batch files with or without command line arguments. Commands can be invoked directly by a user from their client (if permitted by the Server administrator) or as an automated action from the Server's Event Rules. When the Event Rule is triggered, the Server executes the specified custom Command. To configure the Server to execute Commands, you first create the command, then add the command to an Event Rule.

The IC ICommandSettings interface allows you to make changes to the settings of a Site's custom Commands. Access the IC ICommandSettings interface with the [ICISite : GetCommandSettings](#) method. When you change a custom command with the IC ICommandSettings interface, you must call [ICIServer's ApplyChanges](#) method for the changes to take effect.

```
interface IC ICommandSettings : IDispatch
```

Example

```
Set cs = s.GetCommandSettings(aCommands(i))
```

Members

NOTE: "Function" and "Sub" indicate that the member is a method; "Property" indicates that it is a property.

- Sub **AddUserPermission**(bstrUser As String)

Adds user to list of users or Settings Templates who are permitted to execute the command.

```
HRESULT AddUserPermission([in] BSTR bstrUser);
```

All versions

- Property **Description** As String

Retrieves or specifies the description of a command.

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

All versions

- Property **EnableProcessTimeout** As Boolean

Retrieves or specifies whether process timeout is enabled.

```
HRESULT EnableProcessTimeout([out, retval] VARIANT_BOOL
*pVal);
HRESULT EnableProcessTimeout([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled

All versions

- Property **Executable** As String

Retrieves or specifies the name of file executed with the command.

```
HRESULT Executable([out, retval] BSTR *pVal);
HRESULT Executable([in] BSTR newVal);
```

All versions

- Function **GetACL()** As Object

```
HRESULT GetACL([out, retval] IDispatch** ppVal);
```

See also [SetACL](#).

EFT v7.2

- Function **GetUserPermissions()** As Variant

Retrieves an array of users or Settings Templates who have permission to execute the command.

```
HRESULT GetUserPermissions([out, retval] VARIANT *aUsers);
```

All versions

- Property **IsEnabled** As Boolean

Retrieves or specifies whether command is enabled.

```
HRESULT IsEnabled([out, retval] VARIANT_BOOL *pVal);
HRESULT IsEnabled([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled

All versions

- Property **LogPath** As String

Retrieves or specifies the path to the log for the command.

```
HRESULT LogPath([out, retval] BSTR *pVal);
HRESULT LogPath([in] BSTR newVal);
```

Examples (VBScript):

Retrieving:

```
logPath = Command.LogPath
```

Specifying:

```
Command.LogPath = "C:\" & Command.Name & ".log"
```

EFT v6.1 and later

- Property **MinNumOfParams** As Long

Retrieves or specifies the minimum number of parameters required to execute the command.

```
HRESULT MinNumOfParams([out, retval] long *pVal);
HRESULT MinNumOfParams([in] long newVal);
```

All versions

- Property **MinNumOfParamsMsg** As String

Retrieves or changes the message sent when the wrong number of parameters is provided to execute a command.

```
HRESULT MinNumOfParamsMsg([out, retval] BSTR *pVal);
HRESULT MinNumOfParamsMsg([in] BSTR newVal);
```

All versions

- Property **Name** As String

Retrieves or specifies the name of the command. (Do not confuse with **Executable**.)

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

All versions

- Property **Parameters** As String

Retrieves or specifies the list of parameters required to execute command.

```
HRESULT Parameters([out, retval] BSTR *pVal);
HRESULT Parameters([in] BSTR newVal);
```

All versions

- Property **ProcessTimeout** As Long

Retrieves or specifies the number of seconds to before timeout if command fails to execute.

```
HRESULT ProcessTimeout([out, retval] long *pVal);
HRESULT ProcessTimeout([in] long newVal);
```

All versions

- Property **RedirectOutputToClient** As Boolean

Retrieve or specify whether to redirect output to the client.

```
HRESULT RedirectToClient([out, retval] VARIANT_BOOL
*pVal);
HRESULT RedirectToClient([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled

All versions

- Property **RedirectOutputToLog** As Boolean

Retrieves or specifies whether to redirect output to the log.

```
HRESULT RedirectToLog([out, retval] VARIANT_BOOL
*pVal);
HRESULT RedirectToLog([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled

All versions

- Sub **RemoveUserPermission**(bstrUser As String)

Removes a user or Settings Template from list of users or Settings Templates who are permitted to execute the command.

```
HRESULT RemoveUserPermission([in] BSTR bstrUser);
```

All versions

- Property **RequireParams** As Boolean

Retrieves or specifies whether parameters are required to execute the command.

```
HRESULT RequireParams([out, retval] VARIANT_BOOL *pVal);
HRESULT RequireParams([in] VARIANT_BOOL newVal);
```

TRUE=Require parameters; **FALSE**=Do not require parameters

All versions

- Sub **SetACL**(pVal As Object)

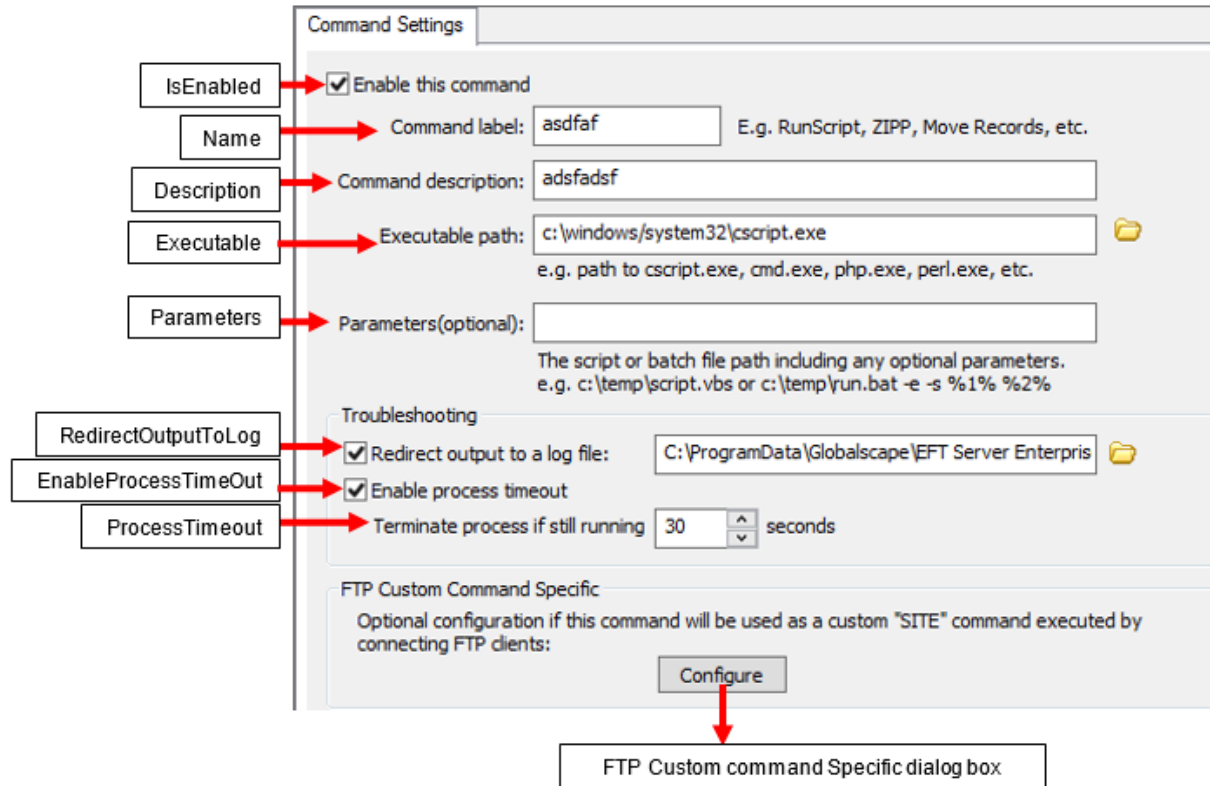
```
HRESULT SetACL([in] IDispatch* pVal);
```

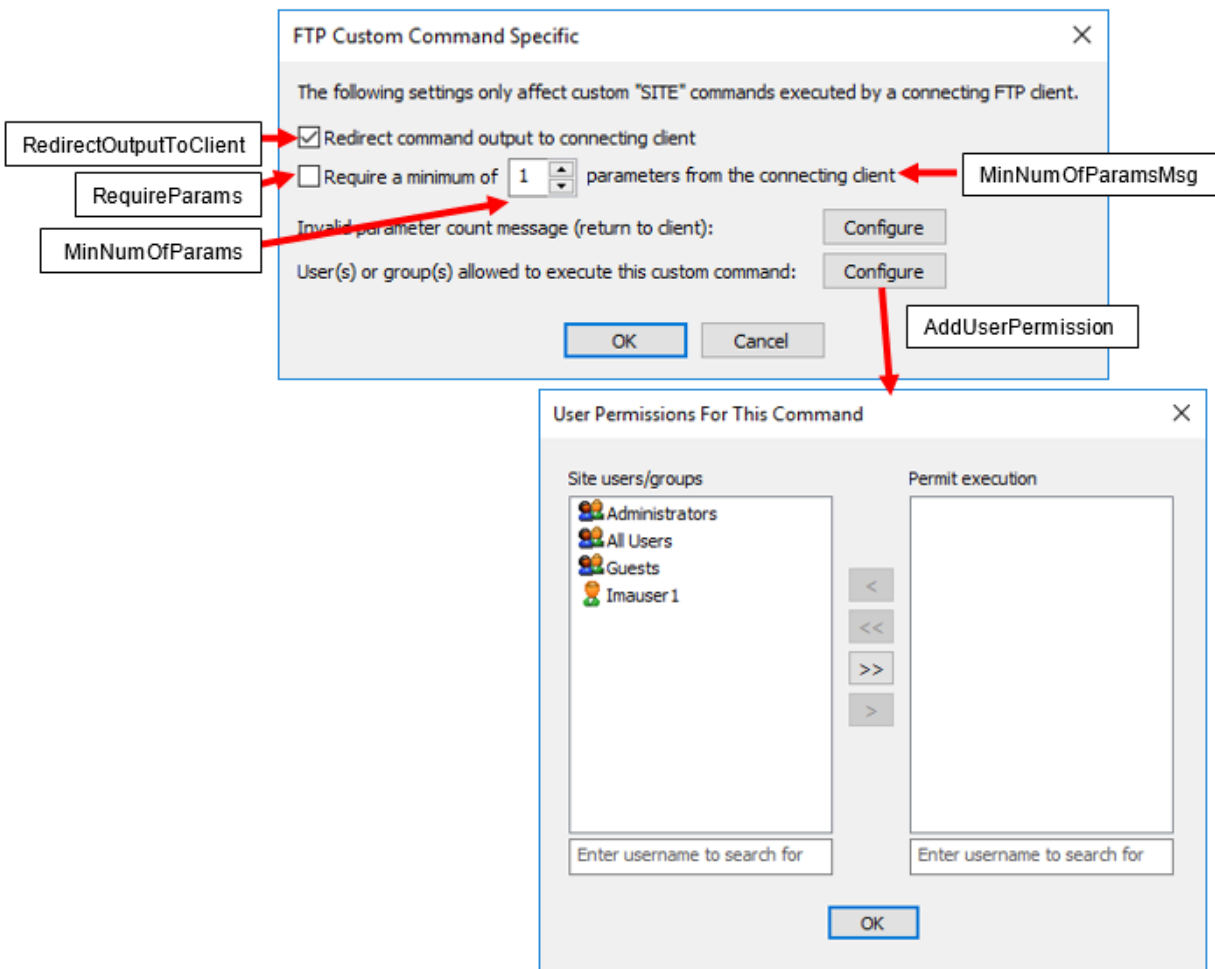
See also [GetACL](#).

EFT v7.2

Dialog Box Equivalents

The ICICCommandSettings interface methods and properties correlate to the following fields and controls in the EFT administration interface. (The FTP Custom Command Specific dialog box appears when you click Configure.)





ICICompoundCondition Interface

The ICICompoundCondition interface allows you to make changes to the Event Rule Condition settings. (See also [ICISimpleCondition - Simple Condition Interface](#).)

```
interface ICICompoundCondition : IDispatch
```

Members

- Function **Add**(property As [EventProperty](#), op As [ConditionOperator](#), varValue, bNot As Boolean) As Object

Add a compound Condition.

```
HRESULT Add(  
    [in] EventProperty property,  
    [in] ConditionOperator op,  
    [in] VARIANT varValue,
```

```
[in] VARIANT_BOOL bNot,  
[out, retval] IDispatch** ppdispSimpleCondition);
```

EFT v5.2 and later

- Function **Count()** As Long

Retrieve the number of listed conditions.

```
HRESULT Count([out, retval] long *plCount);
```

EFT v5.2 and later

- Sub **Delete**(lIndex As Long)

Deletes a Condition.

```
HRESULT Delete([in] long lIndex);
```

EFT v5.2 and later

- Function **Item**(lIndex As Long) As Object

Retrieves a simple condition object by index.

```
HRESULT Item([in] long lIndex,[out, retval] IDispatch**  
ppdispSimpleCondition);
```

EFT v5.2 and later

- Property **Operator** As [LogicalOperator](#)

Retrieves or specifies a logical operation used in calculating the compound condition result (grouping of simple condition items).

```
HRESULT Operator([out, retval] LogicalOperator *pVal);  
HRESULT Operator([in] LogicalOperator newVal);
```

EFT v5.2 and later

IClCompressActionParams

The IClCompressActionsParams interface is used in Event Rules.

Members

- Property **Destination** As String

```
HRESULT Destination([out, retval] BSTR* pVal);
HRESULT Destination([in] BSTR newVal);
```

EFT v7.2

- Property **Format** As [CompressFormat](#)

```
HRESULT Format([out, retval] CompressFormat* pVal);
HRESULT Format([in] CompressFormat newVal);
```

EFT v7.2

- Property **IncludeSubfolders** As Boolean

```
HRESULT IncludeSubfolders([out, retval] VARIANT_BOOL* pVal);
HRESULT IncludeSubfolders([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **Level** As Long

```
HRESULT Level([out, retval] long* pVal);
HRESULT Level([in] long newVal);
```

EFT v7.2

- Property **Method** As [CompressMethod](#)

```
HRESULT Method([out, retval] CompressMethod* pVal);
HRESULT Method([in] CompressMethod newVal);
```

EFT v7.2

- Property **Operation** As [CompressOperation](#)

```
HRESULT Operation([out, retval] CompressOperation* pVal);
HRESULT Operation([in] CompressOperation newVal);
```

EFT v7.2

- Property **OverwritePolicy** As [CompressOverwritePolicy](#)

```
HRESULT OverwritePolicy([out, retval]
CompressOverwritePolicy* pVal);
```

```
HRESULT OverwritePolicy([in] CompressOverwritePolicy
newVal);
```

EFT v7.2

- Property **OverwriteReadOnlyFiles** As Boolean

```
HRESULT OverwriteReadOnlyFiles([out, retval] VARIANT_BOOL*
pVal);
HRESULT OverwriteReadOnlyFiles([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **PassPhrase** As String

```
HRESULT Passphrase([out, retval] BSTR* pVal);
HRESULT Passphrase([in] BSTR newVal);
```

EFT v7.2

- Property **RemoveSourceFiles** As Boolean

```
HRESULT RemoveSourceFiles([out, retval] VARIANT_BOOL* pVal);
HRESULT RemoveSourceFiles([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **Source** As String

```
HRESULT Source([out, retval] BSTR* pVal);
HRESULT Source([in] BSTR newVal);
```

EFT v7.2

ICICConnectionProfileParams

The ICICConnectionProfileParams interface is used to define the Connection Profiles used in Event Rules and Cloud Connector module.

Members (Listed alphabetically)

- Property **AutoLogin** As Boolean

```
HRESULT AutoLogin([out, retval] VARIANT_BOOL *pVal);
HRESULT AutoLogin([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **CloudProviderType** As [CloudProviderType](#)

```
HRESULT Type([out, retval] ConnectionProfileType *pVal);
HRESULT Type([in] ConnectionProfileType newVal);
```

EFT v7.4.6 and later

- Property **ConnectionRetryAttempts** As Long

```
HRESULT ConnectionRetryAttempts([out, retval] LONG *pVal);
HRESULT ConnectionRetryAttempts([in] LONG newVal);
```

EFT v7.2 and later

- Property **ConnectionTimeoutSeconds** As Long

```
HRESULT ConnectionTimeoutSeconds([out, retval] LONG *pVal);
HRESULT ConnectionTimeoutSeconds([in] LONG newVal);
```

EFT v7.2 and later

- Property **Description** As String

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

EFT v7.2 and later

- Property **FTPDataConnectionMode** As [FTPDataConnectionMode](#)

```
HRESULT FTPDataConnectionMode([out, retval]
FTPDataConnectionMode *pVal);
HRESULT FTPDataConnectionMode([in] FTPDataConnectionMode
newVal);
```

EFT v7.2 and later

- Property **FTPDataConnectionPortMax** As Long

```
HRESULT FTPDataConnectionPortMax([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMax([in] LONG newVal);
```

EFT v7.2 and later

- Property **FTPDataConnectionPortMin** As Long

```
HRESULT FTPDataConnectionPortMin([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMin([in] LONG newVal);
```

EFT v7.2 and later

- Property **FTPSClearCommandChannel** As Boolean

```
HRESULT FTPSClearCommandChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT FTPSClearCommandChannel([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **FTPSClearDataChannel** As Boolean

```
HRESULT FTPSClearDataChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT FTPSClearDataChannel([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Function **GetACL()** As Object

```
HRESULT GetACL([out, retval] IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetAWSS3GeneralParams()** As [CICloudAWSS3GeneralParams](#) and Sub **SetAWSS3GeneralParams**(pVal As [CICloudAWSS3GeneralParams](#))

```
HRESULT GetAWSS3GeneralParams([out, retval]
ICICloudAWSS3GeneralParams** ppVal);
HRESULT SetAWSS3GeneralParams([in]
ICICloudAWSS3GeneralParams* pVal);
```

EFT v7.4.6 and later

- Function **GetAzureBlobGeneralParams()** As [CICloudAzureBlobGeneralParams](#) and Sub **SetAzureBlobGeneralParams**(pVal As [CICloudAzureBlobGeneralParams](#))

```
HRESULT GetAzureBlobGeneralParams([out, retval]
ICICloudAzureBlobGeneralParams** ppVal);
HRESULT SetAzureBlobGeneralParams([in]
ICICloudAzureBlobGeneralParams* pVal);
```

EFT v7.4.6 and later

- Property **GUID As String**

```
HRESULT GUID([out, retval] BSTR *pVal);
HRESULT GUID([in] BSTR *pVal);
```

EFT v7.4.11 and later

- Property **Host As String**

```
HRESULT Host([out, retval] BSTR *pVal);
HRESULT Host([in] BSTR newVal);
```

EFT v7.2 and later

- Property **MaxConcurrentThreads As Long**

```
HRESULT MaxConcurrentThreads([out, retval] LONG *pVal);
HRESULT MaxConcurrentThreads([in] LONG newVal);
```

EFT v7.2 and later

- Property **Name As String**

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Password As String**

```
HRESULT Password([out, retval] BSTR *pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Port As Long**

```
HRESULT Port([out, retval] long *pVal);
HRESULT Port([in] long newVal);
```

EFT v7.2 and later

- Property **PreserveLocalFileTime As Boolean**

```
HRESULT PreserveLocalFileTime([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreserveLocalFileTime([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **PreserveRemoteFileTime** As Boolean

```
HRESULT PreserveRemoteFileTime([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreserveRemoteFileTime([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **PrivateKeyPassword** As String

```
HRESULT PrivateKeyPassword([out, retval] BSTR *pVal);
HRESULT PrivateKeyPassword([in] BSTR newVal);
```

EFT v7.2 and later

- Property **PrivateKeyPath** As String

```
HRESULT PrivateKeyPath([out, retval] BSTR *pVal);
HRESULT PrivateKeyPath([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Protocol** As [NetworkProtocol](#)

```
HRESULT Protocol([out, retval] NetworkProtocol *pVal);
HRESULT Protocol([in] NetworkProtocol newVal);
```

EFT v7.2 and later

- Property **ProxyAdvCustomCmds** As String

```
HRESULT ProxyAdvCustomCmds([out, retval] BSTR *pVal);
HRESULT ProxyAdvCustomCmds([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyAdvSeparator** As String

```
HRESULT ProxyAdvSeparator([out, retval] BSTR *pVal);
HRESULT ProxyAdvSeparator([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyAuthenticationType** As [ProxyAuthenticationType](#)

```
HRESULT ProxyAuthenticationType([out, retval]
ProxyAuthenticationType *pVal);
```



```
HRESULT ProxyAuthenticationType([in] ProxyAuthenticationType
newVal);
```

EFT v7.2 and later

- Property **ProxyHostName** As String

```
HRESULT ProxyHostName([out, retval] BSTR *pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyPassword** As String

```
HRESULT ProxyPassword([out, retval] BSTR *pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

EFT v7.2 and later

- Property **ProxyPort** As Long

```
HRESULT ProxyPort([out, retval] LONG *pVal);
HRESULT ProxyPort([in] LONG newVal);
```

EFT v7.2 and later

- Property **ProxyType** As [ProxyType](#)

```
HRESULT ProxyType([out, retval] ProxyType *pVal);
HRESULT ProxyType([in] ProxyType newVal);
```

EFT v7.2 and later

- Property **ProxyUserName** As String

```
HRESULT ProxyUserName([out, retval] BSTR *pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **PublicKeyPath** As String

```
HRESULT PublicKeyPath([out, retval] BSTR *pVal);
HRESULT PublicKeyPath([in] BSTR newVal);
```

EFT v7.2 and later

- Property **RetryDelaySeconds** As Long

```
HRESULT RetryDelaySeconds([out, retval] LONG *pVal);
HRESULT RetryDelaySeconds([in] LONG newVal);
```

EFT v7.2 and later

- Sub **SetACL**(pVal As Object)

```
HRESULT SetACL([in] IDispatch* pVal);
```

EFT v7.2 and later

- Property **SOCKSHostName** As String

```
HRESULT SOCKSHostName([out, retval] BSTR *pVal);
HRESULT SOCKSHostName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **SOCKSPassword** As String

```
HRESULT SOCKSPassword([out, retval] BSTR *pVal);
HRESULT SOCKSPassword([in] BSTR newVal);
```

EFT v7.2 and later

- Property **SOCKSPort** As Long

```
HRESULT SOCKSPort([out, retval] LONG *pVal);
HRESULT SOCKSPort([in] LONG newVal);
```

EFT v7.2 and later

- Property **SOCKSType** As [SOCKSType](#)

```
HRESULT SOCKSType([out, retval] SOCKSType *pVal);
HRESULT SOCKSType([in] SOCKSType newVal);
```

EFT v7.2 and later

- Property **SOCKSUseAuthentication** As Boolean

```
HRESULT SOCKSUseAuthentication([out, retval] VARIANT_BOOL
*pVal);
HRESULT SOCKSUseAuthentication([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **SOCKSUserName** As String

```
HRESULT SOCKSUserName([out, retval] BSTR *pVal);
HRESULT SOCKSUserName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **TextFileTypes** As String

```
HRESULT TextFileTypes([out, retval] BSTR *pVal);
HRESULT TextFileTypes([in] BSTR newVal);
```

EFT v7.2 and later

- Property **TransferEncoding** As [TransferEncodingType](#)

```
HRESULT TransferEncoding([out, retval] TransferEncodingType
*pVal);
HRESULT TransferEncoding([in] TransferEncodingType newVal);
```

EFT v7.2 and later

- Property **TrustedPublicKeyPath** As String

```
HRESULT TrustedPublicKeyPath([out, retval] BSTR *pVal);
HRESULT TrustedPublicKeyPath([in] BSTR newVal);
```

EFT v7.2 and later

- Property **type** As [ConnectionProfileType](#)

```
HRESULT Type([out, retval] ConnectionProfileType *pVal);
[propput, id(1), helpstring("property Type")]
HRESULT Type([in] ConnectionProfileType newVal);
```

EFT v7.4.6 and later

- Property **UseProxy** As Boolean

```
HRESULT UseProxy([out, retval] VARIANT_BOOL *pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **User** As String

```
HRESULT User([out, retval] BSTR *pVal);
HRESULT User([in] BSTR newVal);
```

EFT v7.2 and later

- Property **UseSOCKS** As Boolean

```
HRESULT UseSOCKS([out, retval] VARIANT_BOOL *pVal);
HRESULT UseSOCKS([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **ValidateIntegrity** As Boolean

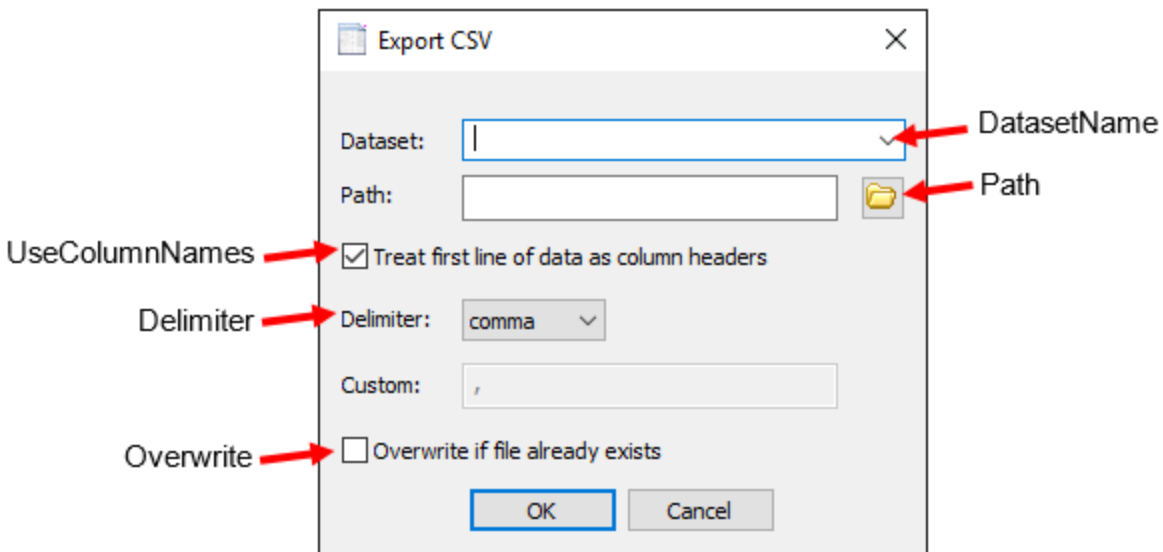
```
HRESULT ValidateIntegrity([out, retval] VARIANT_BOOL *pVal);
HRESULT ValidateIntegrity([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

ICICsvExportActionParams

The ICICsvExportActionParams interface is used to manage parameters for the CSV Export from Dataset to Path Action in Event Rules.

See also [ICICsvImportActionParams](#).



Properties

- Property **DatasetName** As String

```
HRESULT DatasetName([out, retval] BSTR *pVal);
HRESULT DatasetName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Delimiter** As String

```
HRESULT Delimiter([out, retval] BSTR *pVal);
HRESULT Delimiter([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Overwrite** As Boolean

```
HRESULT Overwrite([out, retval] VARIANT_BOOL *pVal);
HRESULT Overwrite([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **Path** As String

```
HRESULT Path([out, retval] BSTR *pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **UseColumnNames** As Boolean

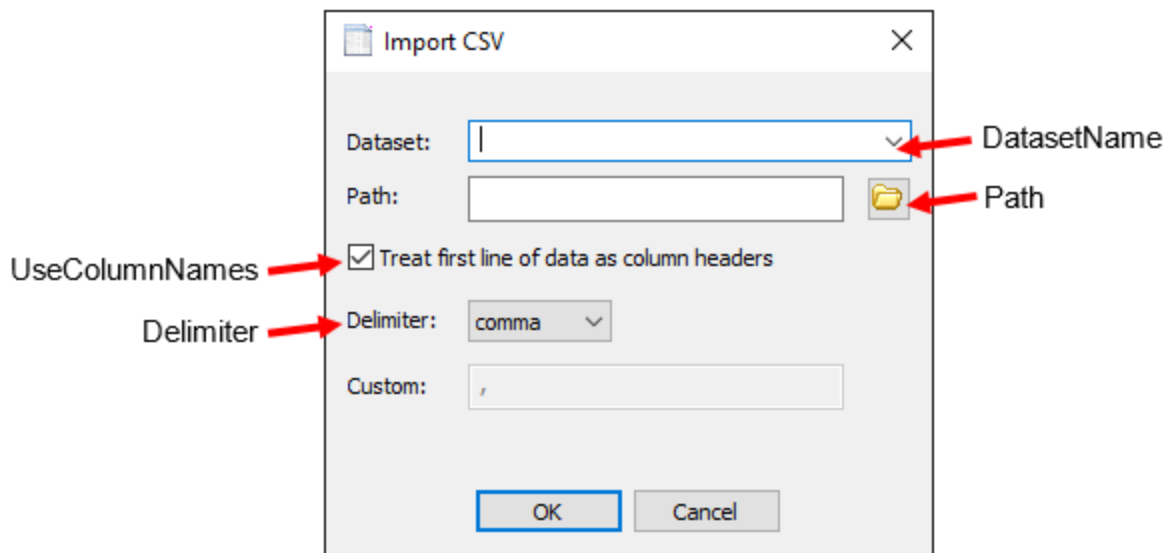
```
HRESULT UseColumnNames([out, retval] VARIANT_BOOL *pVal);
HRESULT UseColumnNames([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

ICICsvImportActionParams

The ICICsvImportActionParams interface is used to manage parameters for the CSV Import to Dataset to Path Action in Event Rules.

See also [ICICsvExportActionParams](#).



Properties

- Property DatasetName As String

```
HRESULT DatasetName([out, retval] BSTR *pVal);
HRESULT DatasetName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property Delimiter As String

```
HRESULT Delimiter([out, retval] BSTR *pVal);
HRESULT Delimiter([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property Path As String

```
HRESULT Path([out, retval] BSTR *pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property UseColumnNames As Boolean

```
HRESULT UseColumnNames([out, retval] VARIANT_BOOL *pVal);
HRESULT UseColumnNames([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

ICIDefineVariableActionParams Interface

The ICIDefineVariableActionParams interface is used to define the Variable Action in Event Rules.

Members

- Property MaskVariable As Boolean

```
HRESULT MaskVariable([out, retval] VARIANT_BOOL *pVal);
HRESULT MaskVariable([in] VARIANT_BOOL newVal);
```

EFT v8 and later

- Property VariableName As String

```
HRESULT VariableName([out, retval] BSTR *pVal);
HRESULT VariableName([in] BSTR newVal);
```

EFT v8 and later

- Property VariableValue As String

```
HRESULT VariableValue([out, retval] BSTR *pVal);
HRESULT VariableValue([in] BSTR newVal);
```

EFT v8 and later

ICIDownloadActionParams Interface

The ICIDownloadActionParams interface (derived from [ICITransferActionParams](#) defining client operation parameters) allows you to make changes to the Event Rule settings.

```
interface ICIDownloadActionParams : ICITransferActionParams
```

Refer to [ICITransferActionParams Interface](#) for details of the methods and properties.

ICIFileActionFilter

The ICIFileActionFilter interface specifies properties of the Event Rules File Operation Action filters.

Members

- Property **ExcludeFiles** As Boolean

Enable or disable the ability to exclude files.

```
HRESULT ExcludeFiles([out, retval] VARIANT_BOOL *pVal);
HRESULT ExcludeFiles([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

- Property **ExclusionMask** As String

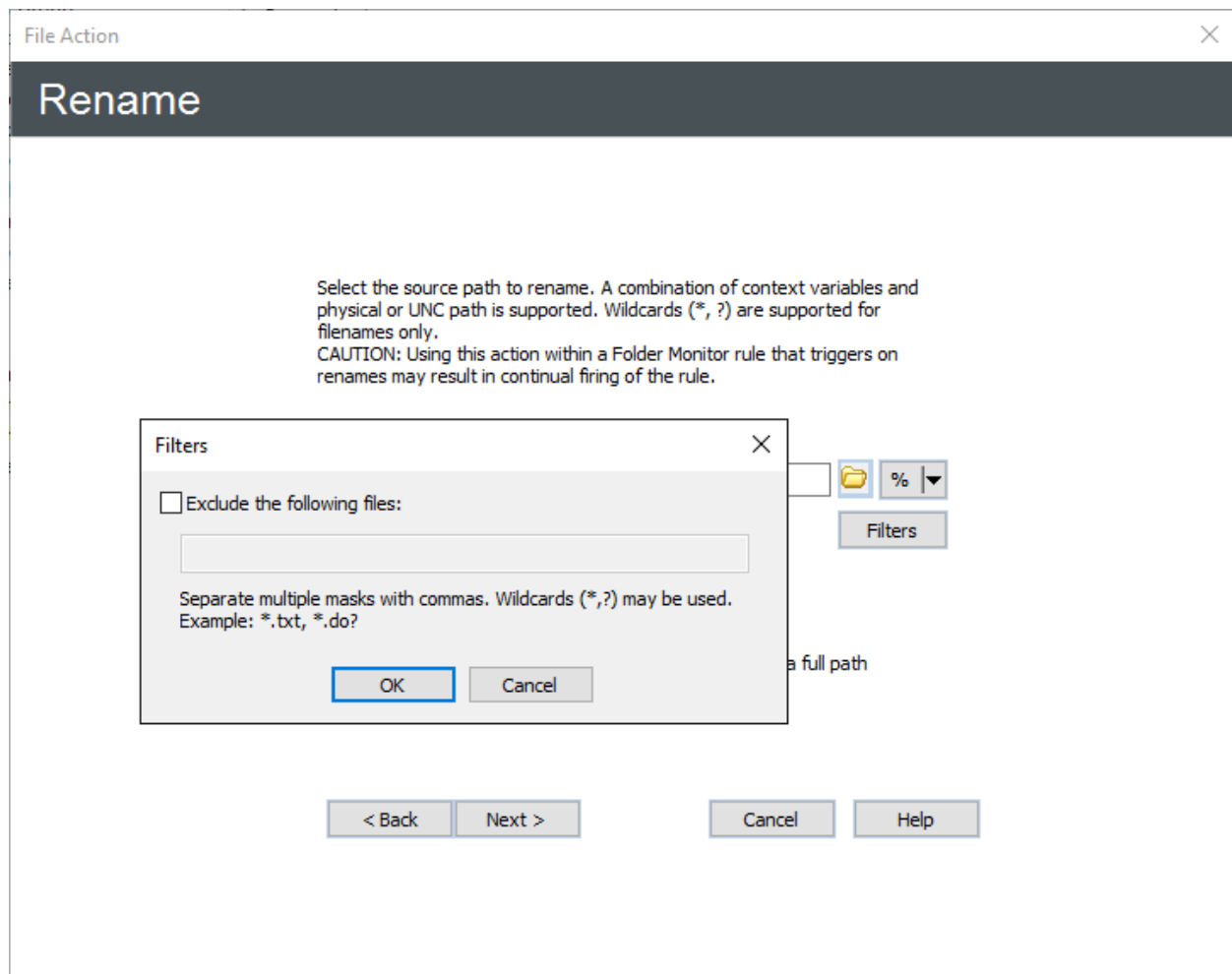
Specify files or types of files to exclude. (ExcludeFiles, above, must be enabled.) You can use wildcards and specify multiple masks with commas, such as:

*.txt, *.do?

```
HRESULT ExclusionMask([out, retval] BSTR* pVal);
HRESULT ExclusionMask([in] BSTR newVal);
```

EFT v8.0.5 and later

Dialog Box Equivalents



ICIFileActionsChecksumParams

The ICIFileActionChecksumParams interface specifies properties of the Event Rules File Operation Action.

- Property **ChecksumAlgorithm** As [ChecksumAlgorithm](#)

Specify the checksum algorithm. Refer to [ChecksumAlgorithm](#) for options.

```
HRESULT ChecksumAlgorithm([out, retval] ChecksumAlgorithm*
pVal);
HRESULT ChecksumAlgorithm([in] ChecksumAlgorithm newVal);
```

EFT v8.0.5 and later

- Property **DestinationVariable** As String

Specify the destination variable to hold the checksum.

```
HRESULT DestinationVariable([out, retval] BSTR* pVal);
HRESULT DestinationVariable([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SourcePath** As String

Specify the source path to perform the checksum on.

```
HRESULT SourcePath([out, retval] BSTR* pVal);
HRESULT SourcePath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

Used to specify the action should be treated as successful if the source file does not exist.

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_
BOOL *pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL
newVal);
```

EFT v8.0.5 and later

ICIFileActionConcatenateParams

The ICIFileActionConcatenateParams interface specifies properties of the Event Rules File Operation Action.

- Property **ConcatenatedPath** As String

```
HRESULT ConcatenatedPath([out, retval] BSTR* pVal);
HRESULT ConcatenatedPath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **MatchingFileOption** As [MatchingFileOption](#)

```
HRESULT MatchingFileOption([out, retval] MatchingFileOption* pVal);
HRESULT MatchingFileOption([in] MatchingFileOption newVal);
```

EFT v8.0.5 and later

- Property **SourcePathA** As String

```
HRESULT SourcePathA([out, retval] BSTR* pVal);
HRESULT SourcePathA([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SourcePathB** As String

```
HRESULT SourcePathB([out, retval] BSTR* pVal);
HRESULT SourcePathB([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_BOOL *pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

ICIFileActionDeleteParams

The ICIFileActionWriteParams interface specifies properties of the Event Rules File Operation Action.

- Function **GetFilter()** As Object

```
HRESULT GetFilter([out, retval] IDispatch* *pVal);
```

EFT v8.0.5 and later

- Property **IncludeSubdirectories** As Boolean

```
HRESULT IncludeSubdirectories([out, retval] VARIANT_BOOL *pVal);
HRESULT IncludeSubdirectories([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

- Sub **SetFilter(newVal As Object)**

```
HRESULT SetFilter([in] IDispatch* newVal);
```

EFT v8.0.5 and later

- Property **SourcePath** As String

```
HRESULT SourcePath([out, retval] BSTR* pVal);
HRESULT SourcePath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_BOOL *pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

ICIFileActionParams

The ICIFileActionParams interface specifies properties of the Event Rules Perform File Operation Action.

- Property **Contents** As String

```
HRESULT Contents([out, retval] BSTR* pVal);
HRESULT Contents([in] BSTR newVal);
```

EFT v7.2 - v8.0.4.x

- Property **Encoding** As [FileEncodingType](#)

```
HRESULT Encoding([out, retval] FileEncodingType* pVal);
HRESULT Encoding([in] FileEncodingType newVal);
```

EFT v7.2 - v8.0.4.x

- Property **ExclusionMasks** As String

```
HRESULT ExclusionMasks([out, retval] BSTR* pVal);
HRESULT ExclusionMasks([in] BSTR newVal);
```

EFT v7.2 - v8.0.4.x

- Function **GetParams()** As Object

```
HRESULT GetParams([out, retval] IDispatch* *pVal);
```

EFT v8.0.5

- Property **NewNameMask** As String

```
HRESULT NewNameMask([out, retval] BSTR* pVal);
HRESULT NewNameMask([in] BSTR newVal);
```

EFT v7.2 - v8.0.4.x

- Property **Operation** As [FileOperationType](#)

```
HRESULT Operation([out, retval] FileOperationType* pVal);
HRESULT Operation([in] FileOperationType newVal);
```

EFT v7. - v8.0.4.x

- Property **OverwriteOption** As [FileOverwriteOptions](#)

```
HRESULT OverwriteOption([out, retval] FileOverwriteOptions*
pVal);
HRESULT OverwriteOption([in] FileOverwriteOptions newVal);
```

EFT v7.2 - v8.0.4.x

- Property **Password** As String

```
HRESULT Password([out, retval] BSTR* pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Path** As String

```
HRESULT Path([out, retval] BSTR* pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v7.2 - v8.0.4.x

- Sub **SetParams**(newVal As Object)

```
HRESULT SetParams([in] IDispatch* newVal);
```

EFT v8.0.5

- Property **Type** As [FileOperationType](#) (read only)

```
HRESULT Type([out, retval] FileOperationType* pVal);
```

EFTv8.0.5

- Property **UseCredentialsFlag** As Boolean

```
HRESULT UseCredentialsFlag([out, retval] VARIANT_BOOL*
pVal);
HRESULT UseCredentialsFlag([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **UseExclusionMasksFlag** As Boolean

```
HRESULT UseExclusionMasksFlag([out, retval] VARIANT_BOOL*
pVal);
HRESULT UseExclusionMasksFlag([in] VARIANT_BOOL newVal);
```

EFT v7.2 - v8.0.4.x

- Property **UserName** As String

```
HRESULT UserName([out, retval] BSTR* pVal);
HRESULT UserName([in] BSTR newVal);
```

EFT v7.2 and later

ICIFileActionReadParams

The ICIFileActionReadParams interface specifies properties of the Event Rules File Operation Action.

- Property **DestinationVariable** As String

```
HRESULT DestinationVariable([out, retval] BSTR* pVal);
HRESULT DestinationVariable([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SourcePath** As String

```
HRESULT SourcePath([out, retval] BSTR* pVal);
HRESULT SourcePath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_
BOOL *pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL
newVal);
```

EFT v8.0.5 and later

ICIFileActionRenameParams

The ICIFileActionRenameParams interface specifies properties of the Event Rules File Operation Action.

IMPORTANT: Using this action within a Folder Monitor rule that triggers on renaming a file may result in the continual firing of the rule.

- Property **DestinationPath** As String

Used to specify the destination path for the renamed file.

```
HRESULT DestinationPath([out, retval] BSTR* pVal);
HRESULT DestinationPath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Function **GetFilter()** As Object

```
HRESULT GetFilter([out, retval] IDispatch* *pVal);
```

EFT v8.0.5 and later

- Property **IncludeSubdirectories** As Boolean

Specify whether to include subfolders in the renaming.

```
HRESULT IncludeSubdirectories([out, retval] VARIANT_BOOL
*pVal);
HRESULT IncludeSubdirectories([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

- Property **MatchingFileOption** As [MatchingFileOption](#)

```
HRESULT MatchingFileOption([out, retval] MatchingFileOption*
pVal);
HRESULT MatchingFileOption([in] MatchingFileOption newVal);
```

EFT v8.0.5 and later

- Sub **SetFilter**(newVal As Object)

```
HRESULT SetFilter([in] IDispatch* newVal);
```

EFT v8.0.5 and later

- Property **SourcePath** As String

Used to select the source path to rename.

```
HRESULT SourcePath([out, retval] BSTR* pVal);
HRESULT SourcePath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_
BOOL *pVal);
```

```
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL
newVal);
```

EFT v8.0.5 and later

ICIFileActionWriteParams

The ICIFileActionWriteParams interface specifies properties of the Event Rules File Operation Action.

- Property **Content** As String

Used to specify the source data or content to write to the destination.

```
HRESULT Content([out, retval] BSTR* pVal);
HRESULT Content([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **DestinationEncoding** As [FileEncodingType](#)

Used to specify the destination encoding. Refer to [FileEncodingType](#) for options.

```
HRESULT DestinationEncoding([out, retval] FileEncodingType*
pVal);
HRESULT DestinationEncoding([in] FileEncodingType newVal);
```

EFT v8.0.5 and later

- Property **DestinationPath** As String

Used to specify the destination path to write to.

```
HRESULT DestinationPath([out, retval] BSTR* pVal);
HRESULT DestinationPath([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **MatchingFileOption** As

Used to specify what to do with files that have a matching name in the destination. Refer to [MatchingFileOption](#) for options.

```
HRESULT MatchingFileOption([out, retval] MatchingFileOption*
pVal);
```



```
HRESULT MatchingFileOption([in] MatchingFileOption newVal);
```

EFT v8.0.5 and later

ICIFolderActionParams

The ICIFolderActionParams interface specifies properties of the Event Rules Folder Operation Action.

- Property **DeleteIfEmptyFlag** As Boolean

```
HRESULT DeleteIfEmptyFlag([out, retval] VARIANT_BOOL* pVal);
HRESULT DeleteIfEmptyFlag([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **NewFolderName** As String

```
HRESULT NewFolderName([out, retval] BSTR* pVal);
HRESULT NewFolderName([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Operation** As [FolderOperationType](#)

```
HRESULT Operation([out, retval] FolderOperationType* pVal);
HRESULT Operation([in] FolderOperationType newVal);
```

EFT v7.2 and later

- Property **Password** As String

```
HRESULT Password([out, retval] BSTR* pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v7.2 and later

- Property **Path** As String

```
HRESULT Path([out, retval] BSTR* pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v7.2 and later

- Property **UseCredentialsFlag** As Boolean

```
HRESULT UseCredentialsFlag([out, retval] VARIANT_BOOL*
pVal);
HRESULT UseCredentialsFlag([in] VARIANT_BOOL newVal);
```

EFT v7.2 and later

- Property **UserName** As String

```
HRESULT UserName([out, retval] BSTR* pVal);
HRESULT UserName([in] BSTR newVal);
```

EFT v7.2 and later

ICIFolderMonitorEventRuleParams Interface

The ICIFolderMonitorEventRuleParams interface allows you to make changes to the Event Rule settings.

```
interface ICIFolderMonitorEventRuleParams : ICIEventRuleParams
```

Properties

- Property **AddTimestampToArchivedFileNames** As Boolean

Retrieves or specifies whether to include a timestamp in names of archived files.

```
HRESULT AddTimestampToArchivedDuplicates([out, retval]
VARIANT_BOOL *pVal);
HRESULT AddTimestampToArchivedDuplicates([in] VARIANT_BOOL
newVal);
```

EFT v6.4 and later

- Property **ArchiveLeftovers** As Boolean

Retrieves or specifies whether to archive the file that has triggered the Folder Monitor Event if the file is present in the monitored folder after the Event Rule is completed.

```
HRESULT ArchiveLeftovers([out, retval] VARIANT_BOOL *pVal);
HRESULT ArchiveLeftovers([in] VARIANT_BOOL newVal);
```

EFT v6.4 and later

- Property **ArchiveSubFolderName** As String

Retrieves or specifies the name of the subfolder into which files are moved during Folder Sweep operations. If this property is null, no post processing is performed.

```
HRESULT ArchiveSubFolderName([out, retval] BSTR *pVal);
HRESULT ArchiveSubFolderName([in] BSTR newVal);
```

(For details of using "Folder Sweep" in Event Rules, refer to the EFT documentation.)

EFT v6.4 and later

- Property **CheckHealth** As Boolean

Retrieves or specifies whether **Check health of network share** is enabled for a Folder Monitor rule.

```
HRESULT CheckHealth([out, retval] VARIANT_BOOL *pVal);
HRESULT CheckHealth([in] VARIANT_BOOL newVal);
```

EFT v5.2 and later

- Property **CheckHealthInterval** As Long

Retrieves or specifies how often to check the health of the network share if **CheckHealth** (above) is enabled.

```
HRESULT CheckHealthInterval([out, retval] long *pVal);
HRESULT CheckHealthInterval([in] long newVal);
```

EFT v5.2 and later

- Property **Description** As String

Description of the event.

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

EFT v5.2 and later

- Property **Enabled** As Boolean

Determine whether the Event Rule is enabled.

```
HRESULT Enabled([out, retval] VARIANT_BOOL *pVal);
HRESULT Enabled([in] VARIANT_BOOL newVal);
```

EFT v5.2 and later

- Property **IncludeSubfolders** As Boolean

Retrieves or specifies whether to include subfolders when monitoring the path.

```
HRESULT IncludeSubfolders([out, retval] VARIANT_BOOL *pVal);
HRESULT IncludeSubfolders([in] VARIANT_BOOL newVal);
```

EFT v5.2 and later

- Property **Name** As String

Name of the event.

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

EFT v5.2 and later

- Property **Password** As String

Retrieves or specifies the password used in the Folder Monitor rule.

```
HRESULT Password([out, retval] BSTR *pVal); [in] BSTR
newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox CStr(oRuleParams.Password)
```

Specifying:

```
oRuleParams.Password = "asd@#!"
```

EFT v6.3.1 and later

- Property **Path** As String

Retrieves or specifies the path to the folder to monitor.

```
HRESULT Path([out, retval] BSTR *pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v5.2 and later

- Property **PollInterval**

Retrieves or specifies the polling interval value for Folder Monitor rule.

```
HRESULT PollInterval([out, retval] long *pVal);
HRESULT PollInterval([in] long newVal);
```

Example (VBScript):**Retrieving:**

```
If EventParams.PollInterval = 0 Then
  'do something
End If
```

Specifying:

```
'set polling interval value to 10 hours for FolderMonitor
event
EventParams.PollIntervalType = 2 ' hours
EventParams.PollInterval = 10
```

EFT v6.5 and later

- Property **PollIntervalSeconds** As Long

Retrieves or specifies poll interval used in the Folder Monitor rule. Design is consistent with the administration interface; property cannot be set with value more than 9999 seconds.

```
HRESULT PollIntervalSeconds([out, retval] long *pVal);
HRESULT PollIntervalSeconds([in] long newVal);
```

EFT v6.4 and later

- Property **PollIntervalType**

See also [FolderMonitorPollIntervalType](#) enum.

Retrieves or specifies the polling interval type for Folder Monitor rule.

```
HRESULT PollIntervalType([out, retval]
FolderMonitorPollIntervalType *pVal);
HRESULT PollIntervalType([in] FolderMonitorPollIntervalType
newVal);
```

Example (VBScript):**Retrieving:**

```
If EventParams.PollIntervalType = 0 Then
  `Seconds
End If
```

Specifying:

```
`set polling interval type to hours for FolderMonitor event
EventParams.PollIntervalType = 2
```

EFT v6.5 and later

- Property **UseAlternateCredentials** As Boolean

Retrieves or specifies whether to use alternate credentials.

```
HRESULT UseAlternateCredentials([out, retval] VARIANT_BOOL
*pVal); [in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox CStr(oRuleParams.UseAlternateCredentials)
```

Specifying:

```
oRuleParams.UseAlternateCredentials = False
```

EFT v6.3.1 and later

- Property **UseFileSystemNotifications** As Boolean

Retrieves or specifies whether the file system notifications must be used when triggering monitor folder.

```
HRESULT UseFileSystemNotifications([out, retval] VARIANT_
BOOL *pVal);
HRESULT UseFileSystemNotifications([in] VARIANT_BOOL
newVal);
```

EFT v6.4 and later

- Property **UseInteractiveLogon** As Boolean

Retrieves or specifies whether the interactive logon must be used in the monitor folder rule.

```
HRESULT UseInteractiveLogon([out, retval] VARIANT_BOOL
*pVal);
HRESULT UseInteractiveLogon([in] VARIANT_BOOL newVal);
```

TRUE = not enabled (the [Require Active Directory domain trust relationship](#) check box is *not* selected)

FALSE = enabled (check box selected).

EFT v6.4 and later

- Property **UsePeriodicDirectoryPoll** As Boolean

Retrieves or specifies whether periodic directory poll must be used in the Monitor Folder Rule.

```
HRESULT UsePeriodicDirectoryPoll([out, retval] VARIANT_BOOL
*pVal);
HRESULT UsePeriodicDirectoryPoll([in] VARIANT_BOOL newVal);
```

EFT v6.4 and later

- Property **Username** As String

Retrieves or specifies the username used in the Folder Monitor rule.

```
HRESULT Username([out, retval] BSTR *pVal); [in] BSTR
newVal);
```

Example (VBScript):

Retrieving:

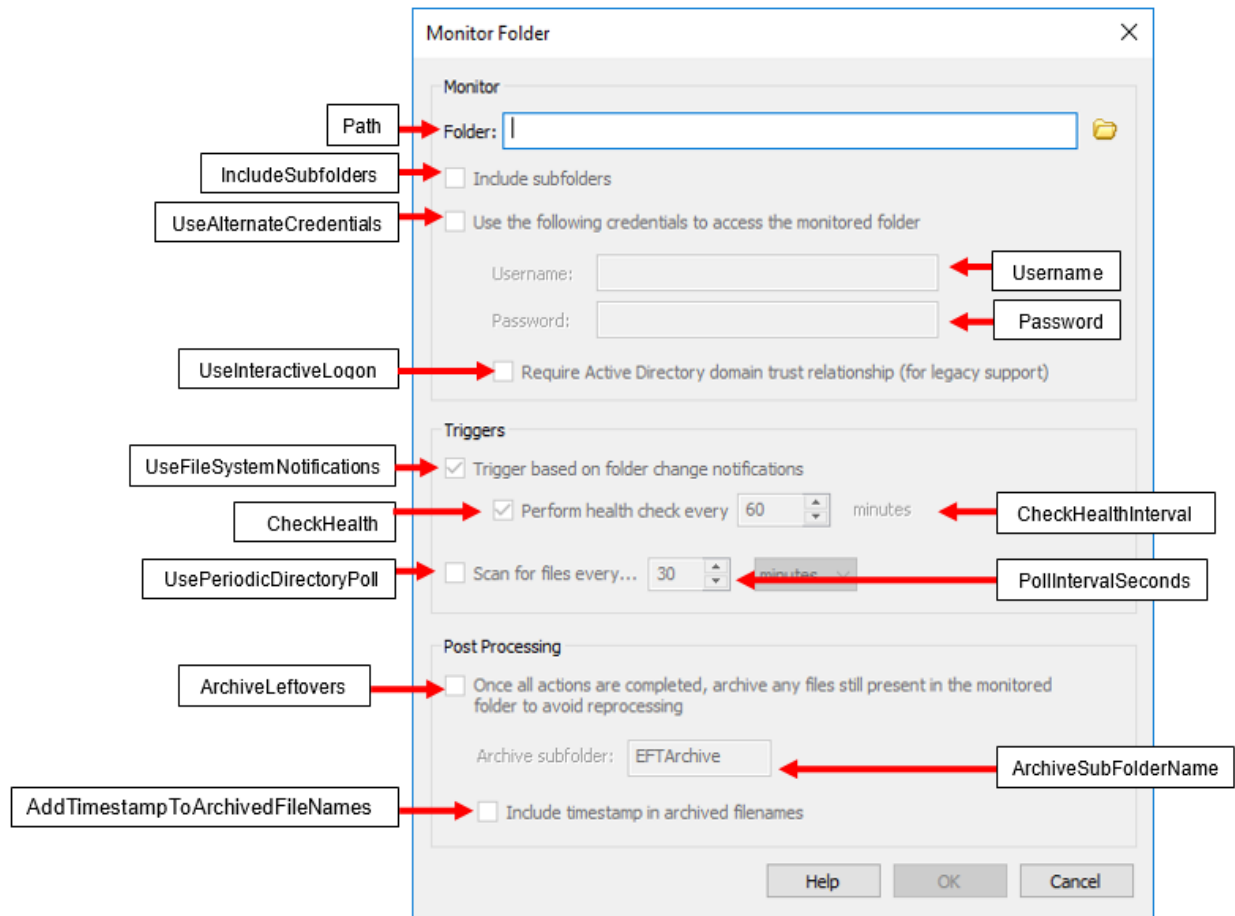
```
MsgBox CStr(oRuleParams.Username)
```

Specifying:

```
oRuleParams.Username = "User"
```

EFT v6.3.1 and later

Dialog Box Equivalents



ICIGetListingActionParams Interface

Properties and Methods

- Property **AutoLogin** As Boolean

```
HRESULT AutoLogin([out, retval] VARIANT_BOOL *pVal);
HRESULT AutoLogin([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **ColumnCount** As Long (read only)

```
HRESULT ColumnCount([out, retval] LONG *pValName);
```

EFT v8.0.4 and later

- Property **ConnectionProfileName** As String


```
HRESULT ConnectionProfileName([out, retval] BSTR *pValName);
HRESULT ConnectionProfileName([in] BSTR newValName);
```

EFT v8.0.4 and later

- Property **ConnectionRetryAttempts** As Long

```
HRESULT ConnectionRetryAttempts([out, retval] LONG *pVal);
HRESULT ConnectionRetryAttempts([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **ConnectionTimeoutSeconds** As Long

```
HRESULT ConnectionTimeoutSeconds([out, retval] LONG *pVal);
HRESULT ConnectionTimeoutSeconds([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **DatasetName** As String

```
HRESULT DatasetName([out, retval] BSTR *pVal);
HRESULT DatasetName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ExcludeFileByExtensionFilter** As String

```
HRESULT ExcludeFileByExtensionFilter([out, retval] BSTR
*pVal);
HRESULT ExcludeFileByExtensionFilter([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **FileDateFormat** As [FileDateFormat](#)

```
HRESULT FileDateFormat([out, retval] FileDateFormat *pVal);
HRESULT FileDateFormat([in] FileDateFormat newVal);
```

EFT v8.0.4 and later

- Property **FTPDataConnectionMode** As [FTPDataConnectionMode](#)

```
HRESULT FTPDataConnectionMode([out, retval]
FTPDataConnectionMode *pVal);
HRESULT FTPDataConnectionMode([in] FTPDataConnectionMode
newVal);
```

EFT v8.0.4 and later

- Property **FTPDataConnectionPortMax** As Long

```
HRESULT FTPDataConnectionPortMax([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMax([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **FTPDataConnectionPortMin** As Long

```
HRESULT FTPDataConnectionPortMin([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMin([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **FTPSClearCommandChannel** As Boolean

```
HRESULT FTPSClearCommandChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT FTPSClearCommandChannel([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **FTPSClearDataChannel** As Boolean

```
HRESULT FTPSClearDataChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT FTPSClearDataChannel([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Function **GetColumnName**(index As Long) As String and Sub **SetColumnName**(index As Long, newVal As String)

```
HRESULT GetColumnName([in] LONG index, [out, retval] BSTR
*pVal);
HRESULT SetColumnName([in] LONG index, [in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Host** As String

```
HRESULT Host([out, retval] BSTR *pVal);
HRESULT Host([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **IncludeSubfolders** As Boolean

```
HRESULT IncludeSubfolders([out, retval] VARIANT_BOOL *pVal);
HRESULT IncludeSubfolders([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **MaxConcurrentThreads** As Long

```
HRESULT MaxConcurrentThreads([out, retval] LONG *pVal);
HRESULT MaxConcurrentThreads([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **Password** As String

```
HRESULT Password([out, retval] BSTR *pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Port** As Long

```
HRESULT Port([out, retval] long *pVal);
HRESULT Port([in] long newVal);
```

EFT v8.0.4 and later

- Property **PortAsString** As String

```
HRESULT PortAsString([out, retval] BSTR *pVal);
HRESULT PortAsString([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **PostTransferCommandParam** As String

```
HRESULT PostTransferCommandParam([out, retval] BSTR *pVal);
HRESULT PostTransferCommandParam([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **PostTransferCommandType** As [PrePostTransferCommand](#)

```
HRESULT PostTransferCommandType([out, retval]
PrePostTransferCommand *pVal);
HRESULT PostTransferCommandType([in] PrePostTransferCommand
newVal);
```

EFT v8.0.4 and later

- Property **PreserveFileTime** As Boolean

```
HRESULT PreserveFileTime([out, retval] VARIANT_BOOL *pVal);
HRESULT PreserveFileTime([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **PreTransferCommandParam** As String

```
HRESULT PreTransferCommandParam([out, retval] BSTR *pVal);
HRESULT PreTransferCommandParam([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **PreTransferCommandType** As [PrePostTransferCommand](#)

```
HRESULT PreTransferCommandType([out, retval]
PrePostTransferCommand *pVal);
HRESULT PreTransferCommandType([in] PrePostTransferCommand
newVal);
```

EFT v8.0.4 and later

- Property **PrivateKeyPassword** As String

```
HRESULT PrivateKeyPassword([out, retval] BSTR *pVal);
HRESULT PrivateKeyPassword([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **PrivateKeyPath** As String

```
HRESULT PrivateKeyPath([out, retval] BSTR *pVal);
HRESULT PrivateKeyPath([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Protocol** As [NetworkProtocol](#)

```
HRESULT Protocol([out, retval] NetworkProtocol *pVal);
HRESULT Protocol([in] NetworkProtocol newVal);
```

EFT v8.0.4 and later

- Property **ProxyAdvCustomCmds** As String

```
HRESULT ProxyAdvCustomCmds([out, retval] BSTR *pVal);
HRESULT ProxyAdvCustomCmds([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ProxyAdvSeparator** As String

```
HRESULT ProxyAdvSeparator([out, retval] BSTR *pVal);
HRESULT ProxyAdvSeparator([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ProxyAuthenticationType** As [ProxyAuthenticationType](#)

```
HRESULT ProxyAuthenticationType([out, retval]
ProxyAuthenticationType *pVal);
HRESULT ProxyAuthenticationType([in] ProxyAuthenticationType
newVal);
```

EFT v8.0.4 and later

- Property **ProxyHostName** As String

```
HRESULT ProxyHostName([out, retval] BSTR *pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ProxyPassword** As String

```
HRESULT ProxyPassword([out, retval] BSTR *pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ProxyPort** As Long

```
HRESULT ProxyPort([out, retval] LONG *pVal);
HRESULT ProxyPort([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **ProxyType** As [ProxyType](#)

```
HRESULT ProxyType([out, retval] ProxyType *pVal);
HRESULT ProxyType([in] ProxyType newVal);
```

EFT v8.0.4 and later

- Property **ProxyUserName** As String

```
HRESULT ProxyUserName([out, retval] BSTR *pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **PublicKeyPath** As String

```
HRESULT PublicKeyPath([out, retval] BSTR *pVal);
HRESULT PublicKeyPath([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **RetryDelaySeconds** As Long

```
HRESULT RetryDelaySeconds([out, retval] LONG *pVal);
HRESULT RetryDelaySeconds([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **SOCKSHostName** As String

```
HRESULT SOCKSHostName([out, retval] BSTR *pVal);
HRESULT SOCKSHostName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **SOCKSPassword** As String

```
HRESULT SOCKSPassword([out, retval] BSTR *pVal);
HRESULT SOCKSPassword([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **SOCKSPort** As Long

```
HRESULT SOCKSPort([out, retval] LONG *pVal);
HRESULT SOCKSPort([in] LONG newVal);
```

EFT v8.0.4 and later

- Property **SOCKSType** As [SOCKSType](#)

```
HRESULT SOCKSType([out, retval] SOCKSType *pVal);
HRESULT SOCKSType([in] SOCKSType newVal);
```

EFT v8.0.4 and later

- Property **SOCKSUseAuthentication** As Boolean

```
HRESULT SOCKSUseAuthentication([out, retval] VARIANT_BOOL
*pVal);
HRESULT SOCKSUseAuthentication([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **SOCKSUserName** As String

```
HRESULT SOCKSUserName([out, retval] BSTR *pVal);
HRESULT SOCKSUserName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **SourcePath** As String

```
HRESULT SourcePath([out, retval] BSTR *pVal);
HRESULT SourcePath([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **TextFileTypes** As String

```
HRESULT TextFileTypes([out, retval] BSTR *pVal);
HRESULT TextFileTypes([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **TransferEncoding** As [TransferEncodingType](#)

```
HRESULT TransferEncoding([out, retval] TransferEncodingType
*pVal);
HRESULT TransferEncoding([in] TransferEncodingType newVal);
```

EFT v8.0.4 and later

- Property **TrustedPublicKeyPath** As String

```
HRESULT TrustedPublicKeyPath([out, retval] BSTR *pVal);
HRESULT TrustedPublicKeyPath([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **UseExcludeFileByExtensionFilter** As Boolean

```
HRESULT UseExcludeFileByExtensionFilter([out, retval]
VARIANT_BOOL *pVal);
```

```
HRESULT UseExcludeFileByExtensionFilter([in] VARIANT_BOOL
newVal);
```

EFT v8.0.4 and later

- Property **UseLocalCodePage** As Boolean

```
HRESULT UseLocalCodePage([out, retval] VARIANT_BOOL *pVal);
HRESULT UseLocalCodePage([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **UseProxy** As Boolean

```
HRESULT UseProxy([out, retval] VARIANT_BOOL *pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **User** As String

```
HRESULT User([out, retval] BSTR *pVal);
HRESULT User([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **UseSOCKS** As Boolean

```
HRESULT UseSOCKS([out, retval] VARIANT_BOOL *pVal);
HRESULT UseSOCKS([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **ValidateIntegrity** As Boolean

```
HRESULT ValidateIntegrity([out, retval] VARIANT_BOOL *pVal);
HRESULT ValidateIntegrity([in] VARIANT_BOOL newVal);
```

ICIIIfStatement Interface

The ICIIIfStatement interface allows you to make changes to the Event Rule settings. See also [ICIEventRule : AddIfStatement](#).

```
interface ICIIIfStatement : ICIEventRuleStatement
```


Properties

- Property **Condition** As Object read-only

Retrieves the Condition object. (Refer to [ICICompoundCondition](#) interface.)

```
HRESULT Condition([out, retval] IDispatch**
ppdispCompoundCondition);
```

EFT v5.2 and later

- Property **ElseSection** As Object read-only

Retrieves the Action statements in Else-section. (Refer to [ICIActionStatements](#) interface.)

```
HRESULT ElseSection([out, retval] IDispatch**
ppdispActionStatements);
```

EFT v5.2 and later

- Property **IfSection** As Object read-only

Retrieves the Action statements in Else-section. (Refer to [ICIActionStatements](#) interface.)

```
HRESULT IfSection([out, retval] IDispatch**
ppdispActionStatements);
```

- Property **type** As [EventRuleStatementType](#) (read-only)

ICILoopDatasetStatement

The ICILoopDatasetStatement interface is used to manage loop statements in Event Rules. Used with the [ICIEventRuleStatement](#) interface and the [EventRuleStatementType](#) enum.

Methods

- Function **AddActionStatement**(lIndex As Long, pdispParams As Object) As Object

```
HRESULT AddActionStatement(
[in] long lIndex,
[in] IDispatch* pdispParams,
[out, retval] IDispatch** ppdispActionStatement);
```

EFT v8.0.4 and later

- Function **AddIfStatement**(lIndex As Long, property As EventProperty, op As ConditionOperator, varConditionValue, bNot As Boolean) As Object

```
HRESULT AddIfStatement(
    [in] long lIndex,
    [in] EventProperty property,
    [in] ConditionOperator op,
    [in] VARIANT varConditionValue,
    [in] VARIANT_BOOL bNot,
    [out, retval] IDispatch** ppdispIfStatement);
```

EFT v8.0.4 and later

- Function **AddLoopDatasetStatement**(lIndex As Long, pdispParams As Object) As Object

```
HRESULT AddLoopDatasetStatement(
    [in] long lIndex,
    [in] IDispatch* pdispParams,
    [out, retval] IDispatch** ppdispLoopDatasetStatement);
```

EFT v8.0.4 and later

- Sub **DeleteStatement**(lIndex As Long)

```
HRESULT DeleteStatement([in] long lIndex);
```

EFT v8.0.4 and later

- Function **GetParams**() As Object

```
HRESULT GetParams([out, retval] IDispatch** ppdispParams);
```

EFT v8.0.4 and later

- Sub **SetParams**(pdispParams As Object)

```
HRESULT SetParams([in] IDispatch* pdispParams);
```

EFT v8.0.4 and later

- Function **Statement**(lIndex As Long) As Object

```
HRESULT Statement([in] long lIndex, [out, retval]
    IDispatch** ppdispStatement);
```

EFT v8.0.4 and later

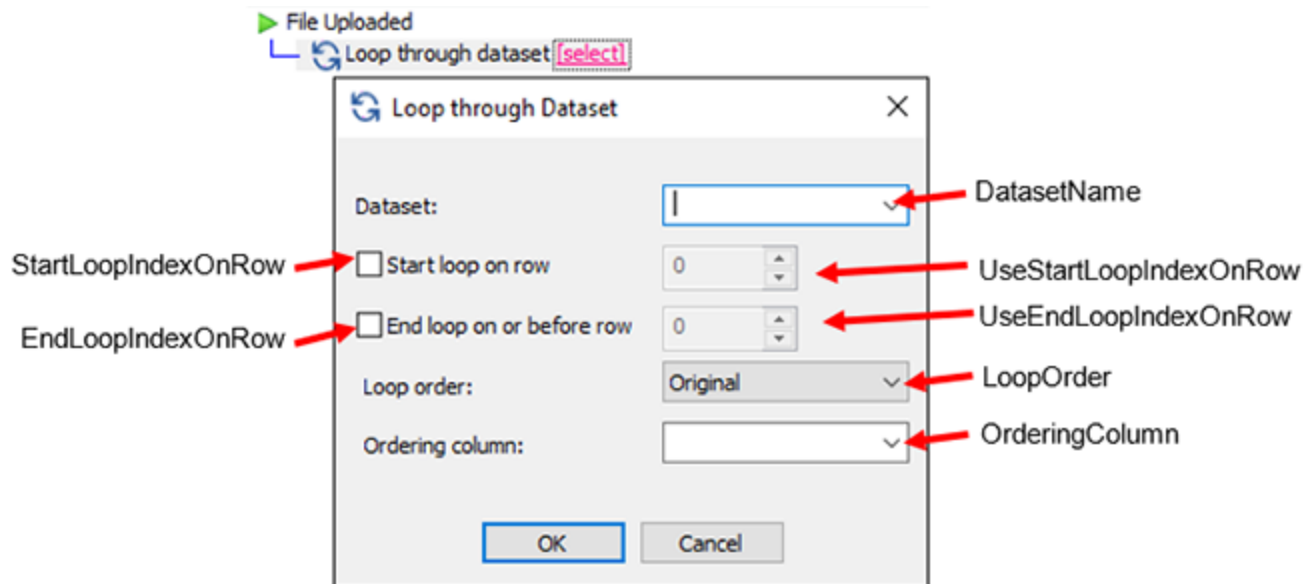
- Function **StatementsCount()** As Long

```
HRESULT StatementsCount([out, retval] long* plCount);
```

EFT v8.0.4 and later

ICILoopDatasetStatementParams

Used to manage the parameters of a Loop through Dataset Action in Event Rules.



Properties

- Property **DatasetName** As String

```
HRESULT DatasetName([out, retval] BSTR *pVal);
HRESULT DatasetName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **EndLoopIndexOnRow** As Long

```
HRESULT EndLoopIndexOnRow([out, retval] long *pVal);
HRESULT EndLoopIndexOnRow([in] long newVal);
```

EFT v8.0.4 and later

- Property **LoopOrder** As [LoopOrderType](#)

```
HRESULT LoopOrder([out, retval] LoopOrderType *pVal);
HRESULT LoopOrder([in] LoopOrderType newVal);
```

EFT v8.0.4 and later

- Property **OrderingColumn** As String

```
HRESULT OrderingColumn([out, retval] BSTR *pVal);  
HRESULT OrderingColumn([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **StartLoopIndexOnRow** As Long

```
HRESULT StartLoopIndexOnRow([out, retval] long *pVal);  
HRESULT StartLoopIndexOnRow([in] long newVal);
```

EFT v8.0.4 and later

- Property **UseEndLoopIndexOnRow** As Boolean

```
HRESULT UseEndLoopIndexOnRow([out, retval] VARIANT_BOOL  
*pVal);  
HRESULT UseEndLoopIndexOnRow([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **UseStartLoopIndexOnRow** As Boolean

```
HRESULT UseStartLoopIndexOnRow([out, retval] VARIANT_BOOL  
*pVal);  
HRESULT UseStartLoopIndexOnRow([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

ICIMailActionParams Interface

You can create an e-mail notification action for Event Rule and AS2 Transaction success and failure notifications.

NOTE: On Sites using AD authentication, EFT must have "Log On as a domain user" permission in order for e-mail notifications to work.

The ICIMailActionParams interface allows you to make changes to the MailAction settings. Access the ICIMailActionParams interface with [ICIEventAction](#) when its EventActionType property returns MailAction.

```
interface ICIMailActionParams : IDispatch
```

Example:

```
If action.Type = 2 then 'MailAction
Set mailParams = action.Params
EndIf
```

VB script example:

```
'Add Mail action to the conditional statement's IF section
Set IfEventActionParams = CreateObject
("SFTPCOMInterface.CIBackupActionParams")
IfEventActionParams.TOAddresses = foobar@com;"
IfEventActionParams.Subject = "Subj"
IfEventActionParams.Template = "<b>text</b>"
Set pMailAction = pIfStatement.IfSection.Add(0, MailAction,
IfEventActionParams)
```

Properties

- Property **Attach As String**

```
HRESULT Attach([out, retval] BSTR *pVal);
HRESULT Attach([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **BCCAddresses As String**

Retrieves or specifies BCC addresses.

```
HRESULT BCCAddresses([out, retval] BSTR *pVal);
HRESULT BCCAddresses([in] BSTR newVal);
```

Example (VBScript):

```
mailParams.BCCAddresses = "rob@globalscape.com;foo@bar.com;"
```

EFT v5.2 and later

- Property **Body** As String

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the body of the message.

```
HRESULT Body([out, retval] BSTR *pVal);
HRESULT Body([in] BSTR newVal);
```

Example (VBScript):

```
mailParams.Body = "<HTML>This message was sent to you  
automatically by the GlobalSCAPE EFT</HTML>"
```

Example (C#):

```
mail.Body = "Test email";
```

EFT v5.2 and later

- Property **CCAddresses** As String

Retrieves or specifies CC addresses.

```
HRESULT CCAddresses([out, retval] BSTR *pVal);
HRESULT CCAddresses([in] BSTR newVal);
```

Example (VBScript):

```
mailParams.CCAddresses = "rob@globalscape.com;foo@bar.com;"
```

EFT v5.2 and later

- Property **CopyToClient** As Boolean

Retrieves or specifies whether to send a copy to the sender.

```
HRESULT CopyToClient([out, retval] VARIANT_BOOL *pVal);
HRESULT CopyToClient([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
mailParams.CopyToClient = True
```

EFT v5.2 and later

- Property **CustomFromAddress** As String

Retrieves or specifies custom a FROM address. (See [UseCustomFromAddress](#) below.)

```
HRESULT CustomFromAddress([out, retval] BSTR *pVal);
HRESULT CustomFromAddress([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
FROMAddr = Server.SMTPSenderAddr 'default
If MailParams.UseCustomFromAddress Then
FROMAddr = MailParams.CustomFromAddress
End If
```

Specifying:

```
MailParams.UseCustomFromAddress = True
MailParams.CustomFromAddress = "custom_email@smtpserver.org"
```

EFT v6.1 and later

- Property **Subject** As String

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the subject line of the message.

```
HRESULT Subject([out, retval] BSTR *pVal);
HRESULT Subject([in] BSTR newVal);
```

Example (VBScript):

```
mailParams.Subject = "Important! Need to talk"
```

Example (C#):

```
mail.Subject = "Test";
```

EFT v5.2 and later

- Property **TOAddresses** As String

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the TO addresses.

```
HRESULT TOAddresses([out, retval] BSTR *pVal);
HRESULT TOAddresses([in] BSTR newVal);
```

Example (VBScript):

```
mailParams.TOAddresses = "rob@globalscape.com;foo@bar.com;"
```

Example (C#):

```
mail.TOAddresses = "youremail@yourdomain.com";
```

EFT v5.2 and later

- Property **UseCustomFromAddress** As Boolean

Retrieves or specifies whether to use a custom FROM address. (See [CustomFromAddress](#) above.)

```
HRESULT UseCustomFromAddress([out, retval] VARIANT_BOOL
*pVal);
HRESULT UseCustomFromAddress([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
FROMAddr = Server.SMTPSenderAddr 'default
If MailParams.UseCustomFromAddress Then
FROMAddr = MailParams.CustomFromAddress
End If
```

Specifying:

```
MailParams.UseCustomFromAddress = False
```


EFT v6.1 and later

- Property **WaitUntilCompletes** As Boolean

Specify whether the Event Rule should wait for this action to complete.

```
HRESULT WaitUntilCompletes([out, retval] VARIANT_BOOL
    *pVal);
HRESULT WaitUntilCompletes([in] VARIANT_BOOL newVal);
```

EFT v8.0.5

ICLObjectAccessRights

The ICIObjectAccessRights interface is used in Event Rules.

- Property **CanDelete** As SFTPAdvBool

```
HRESULT CanDelete([out, retval] SFTPAdvBool *pVal);
HRESULT CanDelete([in] SFTPAdvBool newVal);
```

EFT v7.2

- Property **CanExecute** As SFTPAdvBool

```
HRESULT CanExecute([out, retval] SFTPAdvBool *pVal);
HRESULT CanExecute([in] SFTPAdvBool newVal);
```

EFT v7.2

- Property **CanManagePermissions** As SFTPAdvBool

```
HRESULT CanManagePermissions([out, retval] SFTPAdvBool
    *pVal);
HRESULT CanManagePermissions([in] SFTPAdvBool newVal);
```

EFT v7.2

- Property **CanRead** As SFTPAdvBool

```
HRESULT CanRead([out, retval] SFTPAdvBool *pVal);
HRESULT CanRead([in] SFTPAdvBool newVal);
```

EFT v7.2

- Property **CanWrite** As SFTPAdvBool

```
HRESULT CanWrite([out, retval] SFTPAdvBool *pVal);
HRESULT CanWrite([in] SFTPAdvBool newVal);
```

EFT v7.2

ICObjectACE

The ICObjectACE interface is use in the [ICObjectAccessRights](#) interface.

- Property **AdminID** As [CIAdminID](#) read-only

```
HRESULT AdminID([out, retval] ICAdminID **prop);
```

EFT v7.2 and later

- Property **Access** As [ICObjectAccessRights](#) read-only

```
HRESULT Access([out, retval] ICObjectAccessRights **prop);
```

EFT v7.2 and later

ICObjectACL

The ICObjectACL interface is used with the [ICObjectAccessRights](#) and [ICObjectACE](#) interfaces.

- Function **Add**(pAdminID As Object) As DWord

```
HRESULT Add([in] IDispatch* pAdminID, [out, retval] ULONG*
pnIndex);
```

EFT v7.2

- Property **Count** As DWord read-only

```
HRESULT Count([out, retval] ULONG *pVal);
```

EFT v7.2

- Sub **Delete**(nIndex As DWord)

```
HRESULT Delete([in] ULONG nIndex);
```

EFT v7.2

- Function **Find**(pAdminID As Object) As [ICObjectACE](#)

```
HRESULT Find([in] IDispatch* pAdminID, [out, retval]
ICIObjectACE **ppdisp);
```

EFT v7.2

- Function **GetEffectiveAccessRights**(pAdminID As Object) As [CObjectAccessRights](#)

```
HRESULT GetEffectiveAccessRights([in] IDispatch* pAdminID,
[out, retval] ICIObjectAccessRights** ppVal);
```

EFT v7.2

- Function **Item**(nIndex As DWord) As [CObjectACE](#)

```
HRESULT Item([in] ULONG nIndex, [out, retval] ICIObjectACE**
ppVal);
```

EFT v7.2

ICIPrivacyPolicy Interface

The ICIPrivacyPolicy interface is used to specify the Privacy Policy on the Site for all web portals. The Privacy Policy applies to new account registration, anonymous pickup/dropoff, and after first login.

Members

- Property **AgreementLabels** As String

Retrieves or modifies

```
HRESULT AgreementLabels([out, retval] BSTR *pVal);
HRESULT AgreementLabels([in] BSTR val);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.AgreementLabels = "C:\pplabels.json"
```

EFT v8 and later

- Property **AgreementPath** As String

Retrieves or modifies privacy policy agreement path.

```
HRESULT AgreementPath([out, retval] BSTR *pVal);
HRESULT AgreementPath([in] BSTR val);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.AgreementPath = "C:\pp.json"
```

EFT v8 and later

- Property **EffectiveDate** As Variant

Retrieves or modifies privacy policy effective date.

```
HRESULT EffectiveDate([out, retval] VARIANT *pVal);
HRESULT EffectiveDate([in] VARIANT newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.WScript.echo pp.EffectiveDate
```

EFT v8 and later

- Property **Mode** As [GDPRPrivacyPolicyMode](#)

Retrieves or modifies Privacy Policy mode.

```
HRESULT Mode([out, retval] GDPRPrivacyPolicyMode *pVal);
HRESULT Mode([in] GDPRPrivacyPolicyMode newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.Mode = GDPRPrivacyPolicyMode_Shown
```

EFT v8 and later

- Property **PreventFTPSUntilConsentGranted** As Boolean

Retrieves or modifies "Prevent S/FTP/S until PP is agreed to" option state.

```
HRESULT PreventFTPSUntilConsentGranted([out, retval]
VARIANT_BOOL *pVal);
HRESULT PreventFTPSUntilConsentGranted([in] VARIANT_BOOL
newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.PreventFTPSUntilConsentGranted
```

EFT v8 and later

- Property **RevertToImpliedConsentForNotEUDataSubjects** As Boolean

Retrieves or modifies “Revert to implied consent for users identified as not being EU data subjects” option state.

```
HRESULT RevertToImpliedConsentForNotEUDataSubjects ([out,
retval] VARIANT_BOOL *pVal);
HRESULT RevertToImpliedConsentForNotEUDataSubjects ([in]
VARIANT_BOOL newVal);
```

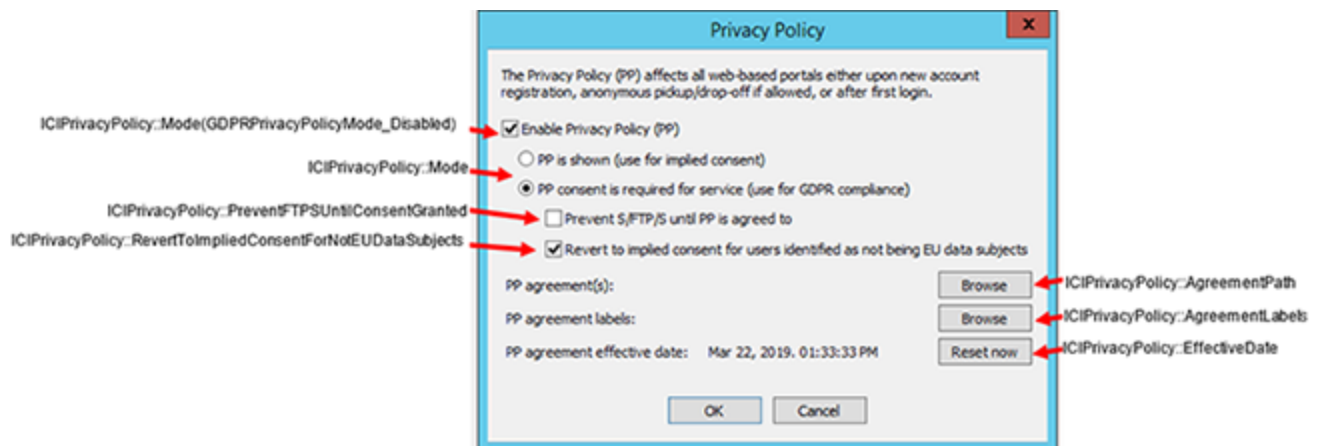
Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.RevertToImpliedConsentForNotEUDataSubjects =False
```

EFT v8 and later

Dialog Box Equivalents

The Privacy Policy dialog box is available from the Site > Web tab.



ICIReportActionParams Interface

The ICIReportActionParams interface allows you to make changes to report actions in Event Rules.

```
interface ICIReportActionParams : IDispatch
```

See also ICIServer Interface: [GenerateReport](#)

Properties

- Property **CustomDate** As [PredefinedReportPeriod](#)

Retrieves or specifies a predefined report period.

```
HRESULT CustomDate([out, retval] PredefinedReportPeriod
*pVal);
HRESULT CustomDate([in] PredefinedReportPeriod newVal);
```

Example:

Retrieving:

```
If ReportParams.CustomDate = AllDates Then
MsgBox "All-time report will be generated"
End If
```

Specifying:

```
ReportParams.CustomDate = Last12Months
```

EFT v5.2 and later

- Property **DateFormat** As [ReportPeriodType](#)

Retrieves or specifies the report period (predefined or a FROM-TO range).

```
HRESULT DateFormat([out, retval] ReportPeriodType *pVal);
HRESULT DateFormat([in] ReportPeriodType newVal);
```

Example:

Retrieving:

```
If ReportParams.DateFormat = Predefined Then
MsgBox "Predefined report range is used"
End If
```

Specifying:

```
ReportParams.DateFormat = Range
```

EFT v5.2 and later

- Property **FilterAndOr** As [ReportFiltersCombiningOperator](#)

Retrieves or specifies a filter.

```
HRESULT FilterAndOr([out, retval]
ReportFiltersCombiningOperator *pVal);
HRESULT FilterAndOr([in] ReportFiltersCombiningOperator
newVal);
```

Example:**Retrieving:**

```
If ReportParams.FilterAndOr = NoCombining Then
MsgBox "Please specify an operator to combine report
filters."
End If
```

Specifying:

```
ReportParams.FilterAndOr = AndCombining
```

EFT v5.2 and later

- Property **FilterField1** As String

Retrieves or specifies a filter.

```
HRESULT FilterField1([out, retval] BSTR *pVal);
HRESULT FilterField1([in] BSTR newVal);
```

EFT v5.2 and later

- Property **FilterField2** As String

Retrieves or specifies a filter.

```
HRESULT FilterField2([out, retval] BSTR *pVal);
HRESULT FilterField2([in] BSTR newVal);
```

EFT v5.2 and later

- Property **FilterOperator1** As [ReportFilterOperator](#)

Retrieves or specifies a filter operator.

```
HRESULT FilterOperator1([out, retval] ReportFilterOperator
*pVal);
HRESULT FilterOperator1([in] ReportFilterOperator newVal);
```

Example:**Retrieving:**

```
If ReportParams.FilterOperator1 = NoFilter Then
MsgBox "Please specify a filter for the report."
End If
```

Specifying:

```
ReportParams.FilterOperator2 = ContainsFilter
```

EFT v5.2 and later

- Property **FilterOperator2** As [ReportFilterOperator](#)

Retrieves or specifies a filter operator.

```
HRESULT FilterOperator2([out, retval] ReportFilterOperator
*pVal);
HRESULT FilterOperator2([in] ReportFilterOperator newVal);
```

EFT v5.2 and later

- Property **FilterValue1** As String

Retrieves or specifies a filter value.

```
HRESULT FilterValue1([out, retval] BSTR *pVal);
HRESULT FilterValue1([in] BSTR newVal);
```

EFT v5.2 and later

- Property **FilterValue2** As String

Retrieves or specifies a filter value.

```
HRESULT FilterValue2([out, retval] BSTR *pVal);
HRESULT FilterValue2([in] BSTR newVal);
```

EFT v5.2 and later

- Property **FromDate** As Variant

Retrieves or specifies the FROM date for the report.


```
HRESULT FromDate([out, retval] VARIANT *pVal);
HRESULT FromDate([in] VARIANT newVal);
```

EFT v5.2 and later

- Property **OptionalParameters** As String

Retrieves or specifies optional parameters used to generate the report.

```
HRESULT OptionalParameters([out, retval] BSTR *pVal);
HRESULT OptionalParameters([in] BSTR newVal);
```

EFT v5.2 and later

- Property **Report** As Object

Retrieves or specifies a report.

```
HRESULT Report([out, retval] IDispatch** pVal);
HRESULT Report([in] IDispatch* newVal);
```

EFT v6.1 and later

- Property **ReportFileFormat** As [ReportFormat](#)

Retrieves or specifies the report format.

```
HRESULT ReportFileFormat([out, retval] ReportFormat *pVal);
HRESULT ReportFileFormat([in] ReportFormat newVal);
```

Example:

Retrieving:

```
If ReportParams.ReportFileFormat = HTMLFormat Then
MsgBox "Report will be generated in HTML format"
End If
```

Specifying:

```
ReportParams.ReportFileFormat = PDFFormat
```

EFT v5.2 and later

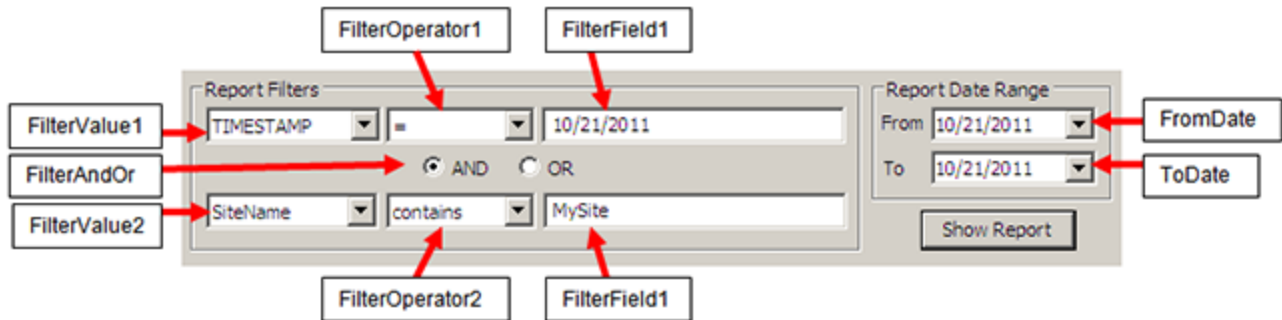
- Property **ToDate** As Variant

Retrieves or specifies the date for the report.

```
HRESULT ToDate([out, retval] VARIANT *pVal);
HRESULT ToDate([in] VARIANT newVal);
```

EFT v5.2 and later

Dialog Box Equivalents



ICIRESTInvocationAuthParams

Used to specify REST authentication type, manage permissions, and get/set a PowerShell script (with [ICIRunPowerShellScriptActionParams Interface](#)).

Members

- Property **Type** As [RESTInvocationAuthType](#)

```
HRESULT Type([out, retval] RESTInvocationAuthType *pVal);
HRESULT Type([in] RESTInvocationAuthType newVal);
```

EFT v8.0.5 and later

- Function **GetUserPermissions()** As Variant

Retrieves an array of users who has a permission to access REST endpoint

```
HRESULT GetUserPermissions([out, retval] VARIANT aUsers);
```

EFT v8.0.5 and later

- Sub **AddUserPermission(bstrUser As String)**

```
HRESULT AddUserPermission([in] BSTR bstrUser);
```

EFT v8.0.5 and later

- Sub **RemoveUserPermission(bstrUser As String)**

```
HRESULT RemoveUserPermission([in] BSTR bstrUser);
```

EFT v8.0.5 and later

- Property **Token** As String

```
HRESULT Token([out, retval] BSTR* pVal);
HRESULT Token([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **Secret** As String

```
HRESULT Secret([out, retval] BSTR* pVal);
HRESULT Secret([in] BSTR newVal);
```

EFT v8.0.5 and later

- Function **GetPowerShellScript()** As [CIRunPowerShellScriptActionParams](#)

```
HRESULT GetPowerShellScript([out, retval]
ICIRunPowerShellScriptActionParams** ppVal);
```

EFT v8.0.5 and later

- Sub **SetPowerShellScript**(pVal As [CIRunPowerShellScriptActionParams](#))

```
HRESULT SetPowerShellScript([in]
ICIRunPowerShellScriptActionParams* pVal);
```

EFT v8.0.5 and later

ICIRESTInvocationEventRuleParams Interface

Used to view or specify REST Invocation Event Rules parameters.

Members

- Property **Endpoint** As String

Set or retrieves the end point for the event rule.

```
HRESULT Endpoint([out, retval] BSTR *pVal);
HRESULT Endpoint([in] BSTR newVal);
```

EFTv8.0.5 and later

- Function **GetAuthParams()** As [CIRESTInvocationAuthParams](#)

Retrieve authentication parameters (`ICIRESTInvocationAuthParams`).

```
HRESULT GetAuthParams([out, retval]
    ICIRESTInvocationAuthParams** ppVal);
```

Example (VBScript), retrieving:

```
Set Rules = site.EventRules(ET_REST_INVOCATION)
Set rule = Rules.Item(0)
Set ruleParams = rule.GetParams
Set authParams = ruleParams.GetAuthParams
```

EFTv8.0.5 and later

- Function **GetVerifyParams()** As [CIRESTInvocationVerifyParams](#)

Retrieve verification parameters.

```
HRESULT GetVerifyParams([out, retval]
    ICIRESTInvocationVerifyParams** ppVal);
```

Example (VBScript) Retrieving:

```
Set Rules = site.EventRules(ET_REST_INVOCATION)
Set rule = Rules.Item(0)
Set ruleParams = rule.GetParams
Set verifyParams = ruleParams.GetVerifyParams
```

EFTv8.0.5 and later

- Property **MinimalNumberOfParamsInQuery** As Int

Sets or retrieves the minimal incoming number of query parameters for the even rule.

```
HRESULT MinimalNumberOfParamsInQuery([out, retval] int
    *pVal);
HRESULT MinimalNumberOfParamsInQuery([in] int newVal);
```

EFTv8.0.5 and later

- Property **Name** As String

Sets or retrieves the name for the event rule.

EFT v8.0.5 and later

- Sub **SetAuthParams**(pVal As [CIRESTInvocationAuthParams](#))

```
HRESULT SetAuthParams([in] ICIRESTInvocationAuthParams*
    pVal);
```

Example:

```
ruleParams.SetAuthParams(authParams)
```

EFTv8.0.5 and later

- Sub **SetVerifyParams**(pVal As [CIRESTInvocationVerifyParams](#))

Sets the verification parameters.

```
HRESULT SetVerifyParams([in] ICIRESTInvocationVerifyParams* pVal);
```

Example (VBScript) Specifying:

```
ruleParams.SetVerifyParams(verifyParams)
```

EFTv8.0.5 and later

ICIRESTInvocationVerifyParams

Used with [ICIRunPowerShellScriptActionParams Interface](#) and the [RESTInvocationVerifyType](#) enum.

Members

- Property **type** As [RESTInvocationVerifyType](#)

```
HRESULT Type([out, retval] RESTInvocationVerifyType *pVal);  
HRESULT Type([in] RESTInvocationVerifyType newVal);
```

EFT v8.0.5 and later

- Function **GetPowerShellScript**() As [CIRunPowerShellScriptActionParams](#)

```
HRESULT GetPowerShellScript([out, retval]  
ICIRunPowerShellScriptActionParams** ppVal);
```

EFT v8.0.5 and later

- Sub **SetPowerShellScript**(pVal As [CIRunPowerShellScriptActionParams](#))

```
HRESULT SetPowerShellScript([in]  
ICIRunPowerShellScriptActionParams* pVal);
```

EFT v8.0.5 and later

ICIRoundRobinEventRuleSchedulingParams Interface

The ICIRoundRobinEventRuleSchedulingParams interface provides the methods and properties used for scheduling the "round robin" process for Event Rules in a high availability, active-active cluster.

- Sub **Add**(newVal As String)

This function adds a new node entry to the run on list for the event rule.

```
HRESULT Add([in] BSTR newVal);
```

EFT v7.0 and later

- Function **Count**() As Long

This function returns the number of nodes in the event rule's run on list.

```
HRESULT Count([out, retval] long* plCount);
```

EFT v7.0 and later

- Sub **Delete**(lIndex As Long)

This function removes the node entry at the specified list index.

```
HRESULT Delete([in] long lIndex);
```

EFT v7.0 and later

- Function **Item**(lIndex As Long) As String

This function returns the node name at the specified list index.

```
HRESULT Item([in] long lIndex, [out, retval] BSTR *pVal);
```

EFT v7.0 and later

ICIRunPowerShellScriptActionParams Interface

The ICIRunPowerShellScriptActionParams interface is used in the Run PowerShell Script Action in Event Rules.

Members

- **Property IsExternal As Boolean**

```
HRESULT IsExternal([out, retval] VARIANT_BOOL *pVal);  
HRESULT IsExternal([in] VARIANT_BOOL newVal);
```

EFT v8 and later

- **Property Script As String**

```
HRESULT Script([out, retval] BSTR *pVal);  
HRESULT Script([in] BSTR newVal);
```

EFT v8 and later

- **Property ScriptPath As String**

```
HRESULT ScriptPath([out, retval] BSTR *pVal);  
HRESULT ScriptPath([in] BSTR newVal);
```

EFT v8

- **Property TimeoutInSec As Int**

```
HRESULT TimeoutInSec([out, retval] int* pVal);  
HRESULT TimeoutInSec([in] int newVal);
```

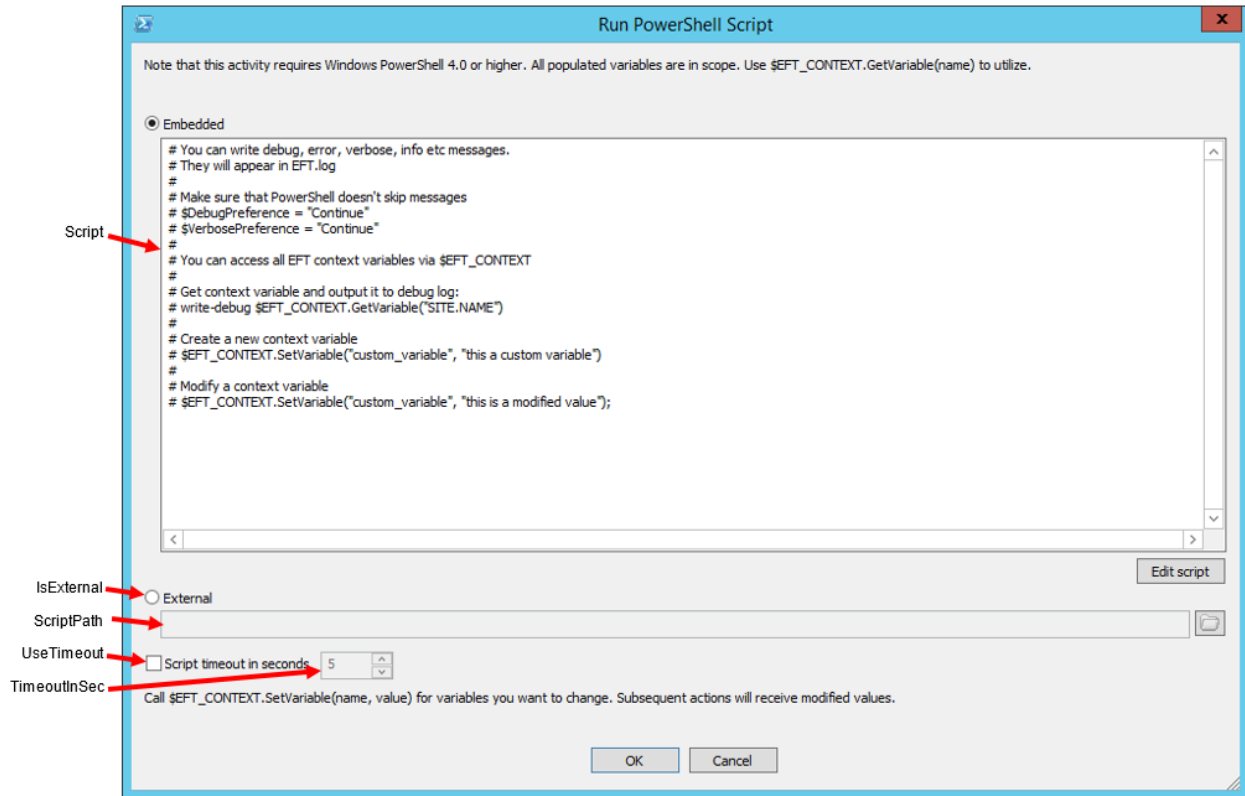
EFT v8

- **Property UseTimeout As Boolean**

```
HRESULT UseTimeout([out, retval] VARIANT_BOOL *pVal);  
HRESULT UseTimeout([in] VARIANT_BOOL newVal);
```

EFT v8

Dialog Box Equivalents



ISATMessage

This interface is used as an input parameter to [ICIAudit](#) to log ad hoc transfer activity. This is conceptually equivalent to the parameter list used in [ICIAudit::AuditSATTransaction](#).

Properties

- Property **Body** As String

Retrieves or specifies the body of the email accompanying the transfer.

```
HRESULT Body([out, retval] BSTR *pVal);
HRESULT Body([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **ExpireDays** As Int

Retrieves or specifies the number of days that the transfer will be accessible.

```
HRESULT ExpireDays([out, retval] INT *pVal);
HRESULT ExpireDays([in] INT newVal);
```


EFT v6.5 -7.4.x

- Property **From** As String

Retrieves or specifies the email address of the user responsible for the transfer.

```
HRESULT From([out, retval] BSTR *pVal);
HRESULT From([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **Reserved1** As String

Retrieves or specifies the field reserved to record additional data associated with a transfer.

```
HRESULT Reserved1([out, retval] BSTR *pVal);
HRESULT Reserved1([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **Reserved2** As String

Retrieves or specifies the field reserved to record additional data associated with a transfer.

```
HRESULT Reserved2([out, retval] BSTR *pVal);
HRESULT Reserved2([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **SiteName** As String

Retrieves or specifies the name of the site under which the transfer occurred.

```
HRESULT SiteName([out, retval] BSTR *pVal);
HRESULT SiteName([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **Subject** As String

Retrieves or specifies the subject line of email accompanying the transfer.

```
HRESULT Subject([out, retval] BSTR *pVal);
HRESULT Subject([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **TempPassword** As String

Retrieves or specifies the password of the account created to hold the transferred files.

```
HRESULT TempPassword([out, retval] BSTR *pVal);
HRESULT TempPassword([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **TempUserName** As String

Retrieves or specifies the account created to hold the transferred files.

```
HRESULT TempUserName([out, retval] BSTR *pVal);
HRESULT TempUserName([in] BSTR newVal);
```

EFT v6.5 -7.4.x

- Property **Timestamp** As Date

Retrieves the timestamp the ad hoc transfer occurred.

```
HRESULT Timestamp([out, retval] DATE *pVal);
HRESULT Timestamp([in] DATE newVal);
```

EFT v6.5 -7.4.x

- Property **TransGUID** As String

Retrieves or specifies the GUID used to correlate this transfer with email recipients and files.

```
HRESULT TransGUID([out, retval] BSTR *pVal);
HRESULT TransGUID([in] BSTR newVal);
```

EFT v6.5 -7.4.x

ICISimpleCondition Interface

The ICISimpleCondition interface allows you to make changes to the Event Rule Condition settings. (See also [ICICompoundCondition - Compound Condition Interface](#).)

```
interface ICISimpleCondition : IDispatch
```

Properties

- Function **GetContextVariable()** As String

```
HRESULT GetContextVariable([out, retval] BSTR
    *bstrContextVariable);
```

EFT v7.2 and later

- Function **GetIsFileSwitch()** As Boolean

```
HRESULT GetIsFileSwitch([out, retval] VARIANT_BOOL *pVal);
```

EFT v7.2 and later

- Property **Not** As Boolean

Retrieves or specifies whether NOT is used in the Condition.

```
HRESULT Not([out, retval] VARIANT_BOOL *pVal);
HRESULT Not([in] VARIANT_BOOL newVal);
```

EFT v5.2 and later

- Property **Operator** As [ConditionOperator](#)

Retrieves or specifies the condition operator as [ConditionOperator](#).

```
HRESULT Operator([out, retval] ConditionOperator *pVal);
HRESULT Operator([in] ConditionOperator newVal);
```

EFT v5.2 and later

- Property **property** As [EventProperty](#) read-only

Retrieve the property as [EventProperty](#).

```
HRESULT Property([out, retval] EventProperty *pVal);
```

EFT v5.2 and later

- Sub **SetContextVariable**(bstrContextVariable As String)

```
HRESULT SetContextVariable([in] BSTR bstrContextVariable);
```

EFT v7.2 and later

- Sub **SetIsFileSwitch**(pVal As Boolean)

```
HRESULT SetIsFileSwitch([in] VARIANT_BOOL pVal);
```

EFT v7.2 and later

- Property **UseRegex** As Boolean

```
HRESULT UseRegex([out, retval] VARIANT_BOOL *pVal);
HRESULT UseRegex([in] VARIANT_BOOL newVal);
```

EFT v8.0.4

- Property **Value** As Variant

Retrieves or specifies the value associated with the Condition.

```
HRESULT Value([out, retval] VARIANT *pVal);
HRESULT Value([in] VARIANT newVal);
```

EFT v5.2 and later

ICISetActionParams Interface

The ICISetActionParams interface allows you to enable/disable the Stop Processing Action in Event Rules.

```
interface ICISetActionParams : IDispatch
```

Members

- Property **Action** As [StopType](#)

Retrieves or specifies the type of stop.

```
HRESULT Action([out, retval] StopType *pVal);
HRESULT Action([in] StopType newVal);
```

EFT v5.2 and later

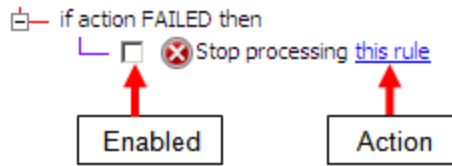
- Property **Enabled** As Boolean

Specifies whether the stop is enabled.

```
HRESULT Enabled([out, retval] VARIANT_BOOL *pVal);
HRESULT Enabled([in] VARIANT_BOOL newVal);
```

EFT v5.2 and later

Dialog Box Equivalents



ICISynchronizeActionParams Interface

Used with the [ICITransferActionParams](#) Interface.

Members

- Property **SyncDirection** As [SyncDirectionType](#)

```
HRESULT SyncDirection([out, retval] SyncDirectionType
*pVal);
HRESULT SyncDirection([in] SyncDirectionType newVal);
```

EFT v8.0.5 and later

- Property **SyncIncludeSubfolders** As Boolean

```
HRESULT SyncIncludeSubfolders([out, retval] VARIANT_BOOL
*pVal);
HRESULT SyncIncludeSubfolders([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

- Property **SyncDeleteFileIfNotPresent** As Boolean

```
HRESULT SyncDeleteFileIfNotPresent([out, retval] VARIANT_
BOOL *pVal);
HRESULT SyncDeleteFileIfNotPresent([in] VARIANT_BOOL
newVal);
```

EFT v8.0.5 and later

- Property **SyncOverwrite** As [SyncOverwriteType](#)

```
HRESULT SyncOverwrite([out, retval] SyncOverwriteType
*pVal);
HRESULT SyncOverwrite([in] SyncOverwriteType newVal);
```

EFT v8.0.5 and later

ICITimerEventRuleParams Interface

Use ICITimerEventRuleParams to make changes to Timer Event Rule settings. (See also [ICIEventRuleParams - Event Rule Parameters Interface](#).)

```
interface ICITimerEventRuleParams : ICIEventRuleParams
```

Members

- Property [DailyDayPeriod](#) As Long

Retrieves or specifies the recurrence in days (e.g., every 5 days).

```
HRESULT DailyDayPeriod([out, retval] long *pVal);
HRESULT DailyDayPeriod([in] long newVal);
```

EFT v6.3 and later

- Property [DailyEveryWeekDay](#) As Boolean

Retrieves or specifies whether the even recurs every weekday or every <n> days.

```
HRESULT DailyEveryWeekDay([out, retval] VARIANT_BOOL *pVal);
HRESULT DailyEveryWeekDay([in] VARIANT_BOOL newVal);
```

True= Every weekday; False= Every <n> days. Use [DailyDayPeriod](#) to specify the <n> days value.

EFT v6.3 and later

- Property [DateEndEnabled](#) As Boolean

Retrieves or specifies whether an end date is specified.

```
HRESULT DateEndEnabled([out, retval] VARIANT_BOOL *pVal);
HRESULT DateEndEnabled([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [DateTimeEnd](#) As Variant

Retrieves or specifies the date and time the event is to stop recurring; local to the server. (Combines end date and end time values.)

```
HRESULT DateTimeEnd([out, retval] VARIANT *pVal);
HRESULT DateTimeEnd([in] VARIANT newVal);
```

EFT v6.3 and later

- Property [DateTimeStart](#) As Variant

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the date and time the event is to start; local to the server.

```
HRESULT DateTimeStart([out, retval] VARIANT *pVal);
HRESULT DateTimeStart([in] VARIANT newVal);
```

Use in every Timer rule to specify the date and time the event is to start.

Example (C#):

```
objParams.DateTimeStart = new DateTime(2011, 1, 1, 0, 0, 1);
```

EFT v6.3 and later

- Property [HolidayCalendar](#) As String

Retrieves or specifies a holiday calendar.

```
HRESULT HolidayCalendar([out, retval] BSTR*pVal)
HRESULT HolidayCalendar([in] BSTR newVal);
```

EFT v6.3 and later

- Property [MonthlyDayPeriod](#) As Long

Retrieves or specifies a specific day of the month the event is to occur.

```
HRESULT MonthlyDayPeriod([out, retval] long *pVal);
HRESULT MonthlyDayPeriod([in] long newVal);
```

EFT v6.3 and later

- Property [MonthlyFixedDay](#) As Boolean

Retrieves or specifies whether the event occurs on the same day every month.

```
HRESULT MonthlyFixedDay([out, retval] VARIANT_BOOL *pVal);
HRESULT MonthlyFixedDay([in] VARIANT_BOOL newVal);
```

True= A specific day of the month (e.g., on the 21st every 3 months); False= The first, second, third, fourth, or last weekday of the month (e.g., The first Monday of the month).

EFT v6.3 and later

- Property [MonthlyMonthPeriod](#) As Long

Retrieves or specifies the monthly frequency of the event (e.g., every 3 months).

```
HRESULT MonthlyMonthPeriod([out, retval] long *pVal);
HRESULT MonthlyMonthPeriod([in] long newVal);
```

EFT v6.3 and later

- Property [MonthlyRelativeWeekday](#) As [Relation](#)

Retrieves or specifies whether the event occurs on the first, second, third, fourth, or last day of the period (e.g., the first Sunday of the month).

```
HRESULT MonthlyRelativeWeekday([out, retval] Relation
*pVal);
HRESULT MonthlyRelativeWeekday([in] Relation newVal);
```

EFT v6.3 and later

- Property [MonthlyWeekday](#) As [Weekday](#)

Retrieves or specifies the day of the week the event is to occur.

```
HRESULT MonthlyWeekday([out, retval] Weekday *pVal);
HRESULT MonthlyWeekday([in] Weekday newVal);
```

EFT v6.3 and later

- Property **NextRun** As Date read-only

```
HRESULT NextRun([out, retval] DATE* pVal);
```

- Property [Recurrence](#) As [Recurrence](#)

For an example using this property, refer to the EventRuleCreation script in [Script Examples](#).

Retrieves or specifies the frequency with which the Timer event is to occur (hourly, daily, weekly, monthly, yearly, once).

```
HRESULT Recurrence([out, retval] Recurrence *pVal);
HRESULT Recurrence([in] Recurrence newVal);
```

Example (C#):

```
objParams.Recurrence = Recurrence.Recurrence_Once;
```


EFT v5.2 and later

- Property [RepeatEnabled](#) As Boolean

Retrieves or specifies whether the event is to repeat.

```
HRESULT RepeatEnabled([out, retval] VARIANT_BOOL *pVal);
HRESULT RepeatEnabled([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [RepeatPattern](#) As [RepeatPattern](#)

Retrieves or specifies the recurrence pattern (e.g., every 4 hours).

```
HRESULT RepeatPattern([out, retval] RepeatPattern *pVal);
HRESULT RepeatPattern([in] RepeatPattern newVal);
```

EFT v6.3 and later

- Property [RepeatRate](#) As Long

Retrieves or specifies the frequency the event is to recur.

```
HRESULT RepeatRate([out, retval] long *pVal);
HRESULT RepeatRate([in] long newVal);
```

EFT v6.3 and later

- Property [RundayCalendar](#) As String

Retrieves or specifies a custom calendar.

```
HRESULT RundayCalendar([out, retval] BSTR *pVal);
HRESULT RundayCalendar([in] BSTR newVal);
```

EFT v6.3 and later

- Property [TimeEndEnabled](#) As Boolean

Retrieves or specifies whether end time is enabled.

```
HRESULT TimeEndEnabled([out, retval] VARIANT_BOOL *pVal);
HRESULT TimeEndEnabled([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklyFriday](#) As Boolean

Retrieves or specifies every week on Friday.

```
HRESULT WeeklyFriday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklyFriday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklyMonday](#) As Boolean

Retrieves or specifies every week on Monday.

```
HRESULT WeeklyMonday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklyMonday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklySaturday](#) As Boolean

Retrieves or specifies every week on Saturday.

```
HRESULT WeeklySaturday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklySaturday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklySunday](#) As Boolean

Retrieves or specifies every week on Sunday.

```
HRESULT WeeklySunday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklySunday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklyThursday](#) As Boolean

Retrieves or specifies every week on Thursday.

```
HRESULT WeeklyThursday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklyThursday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklyTuesday](#) As Boolean

Retrieves or specifies every week on Tuesday.

```
HRESULT WeeklyTuesday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklyTuesday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklyWednesday](#) As Boolean

Retrieves or specifies every week on Wednesday.

```
HRESULT WeeklyWednesday([out, retval] VARIANT_BOOL *pVal);
HRESULT WeeklyWednesday([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [WeeklyWeekPeriod](#) As Long

Retrieves or specifies a frequency in weeks (e.g., every 2 weeks).

```
HRESULT WeeklyWeekPeriod([out, retval] long *pVal);
HRESULT WeeklyWeekPeriod([in] long newVal);
```

EFT v6.3 and later

- Property [YearlyDayPeriod](#) As Long

Retrieves or specifies a frequency (e.g., every 7 days).

```
HRESULT YearlyDayPeriod([out, retval] long *pVal);
HRESULT YearlyDayPeriod([in] long newVal);
```

EFT v6.3 and later

- Property [YearlyFixedDay](#) As Boolean

Retrieves or specifies a specific day (e.g., Every December 31 or the last Saturday of December)..

```
HRESULT YearlyFixedDay([out, retval] VARIANT_BOOL *pVal);
HRESULT YearlyFixedDay([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [YearlyMonth](#) As [Month](#)

Retrieves or specifies the month (e.g., Every December 31 or the last Saturday of December).

```
HRESULT YearlyMonth([out, retval] Month *pVal);
HRESULT YearlyMonth([in] Month newVal);
```

EFT v6.3 and later

- Property [YearlyRelativeWeekday](#) As [Relation](#)

Retrieves or specifies whether the day is the first, second, third, fourth, or last day of the week.

```
HRESULT YearlyRelativeWeekday([out, retval] Relation *pVal);
HRESULT YearlyRelativeWeekday([in] Relation newVal);
```

EFT v6.3 and later

- Property [YearlyWeekday](#) As [Weekday](#)

Retrieves or specifies a day of the week.

```
HRESULT YearlyWeekday([out, retval] Weekday *pVal);
HRESULT YearlyWeekday([in] Weekday newVal);
```

EFT v6.3 and later

ICITimerEventRuleParams in Previous Versions

If you have Timer rules defined in previous versions of COM, note that the Timer event trigger has had major improvements in v6.3 and later; therefore, the API's TimerEventRuleParams interface has also undergone many changes. (Refer to "Scheduler (Timer) Event" in the EFT documentation for details of how this Event is used in the administration interface or [scroll down](#) to see how the methods and properties correlate to the interface fields and options.)

The constants in the Recurrence enumeration were changed, and "Recurrence_Calendar" was added:

- Custom (0) was changed to Recurrence_Continually = 0; The Event starts at a specified (future) date and time and repeats every <n> Hours, Minutes, or Seconds.
- Daily (1) was changed to Recurrence_Daily = 1; The Event runs every <n> days or every weekday, starting at a specified date and time, and ending on a specified date and time or repeating every <n> hours, minutes, or seconds. You can also exclude certain holidays and/or end the recurrence of the event at a specified date and time.
- Weekly (2) was changed to Recurrence_Weekly = 2; The Event runs every <n> weeks on a specified days of the week, starting at a specified date and time and ending on a specified date and time or repeating every <n> hours, minutes, or seconds. You can

also exclude certain holidays and/or end the recurrence of the event at a specified date and time.

- Monthly (3) was changed to `Recurrence_Monthly = 3`; The Event runs on the <n> day of every <n> months or the <nth> day of the week of <n> months starting at a specified date and time and ending on a specified date and time or repeating every <n> hours, minutes, or seconds. You can also exclude certain holidays and/or end the recurrence of the event at a specified date and time.
- Yearly (4) was changed to `Recurrence_Yearly = 4`; The Event runs every <month> <day> or on the <n> <day of the week> of <month> starting at a specified date and time and ending on a specified date and time or repeating every <n> hours, minutes, or seconds. You can also exclude certain holidays and/or end the recurrence of the event at a specified date and time.
- OneTime (5) was changed to `Recurrence_Once = 5`; The Event runs one time at a specified date and time, and never repeats.
- `Recurrence_Calendar = 6` as added; The timer uses the calendar to get the next run day. This value should not be directly set to the Recurrence property of the `TimerEventRuleParams` object. Instead, if you want to use the run calendar in your timer, set the calendar name to the [RundayCalendar](#) property, then the Recurrence is automatically set to the `Recurrence_Calendar` value. You can also exclude certain holidays and/or end the recurrence of the event at a specified date and time.

The following properties in the `TimerEventRuleParams` interface were removed or renamed in EFT v6.3:

- `CustomPeriod`
- `DayIndex`
- `DayOfMonth`
- `DayPeriod`
- `FixedDate`
- `MonthIndex`
- `MonthPeriod`
- `TimeStart`
- `WeekDayIndex`
- `Weekdays`
- `WeekPeriod`

The following properties, enumerators, and constants in the `TimerEventRuleParams` interface were added in EFT Server v6.3:

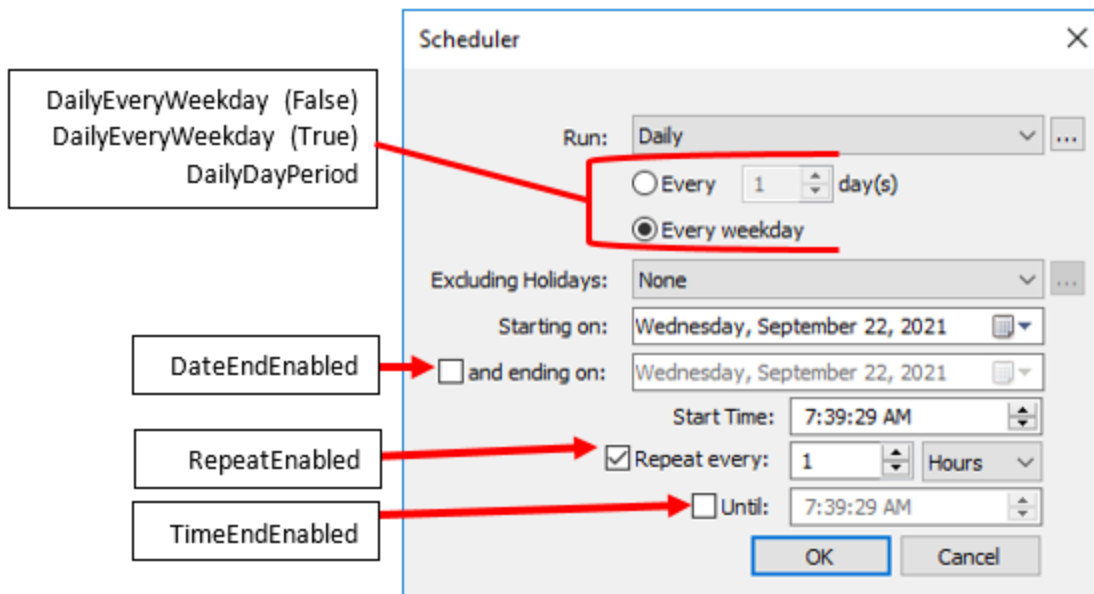
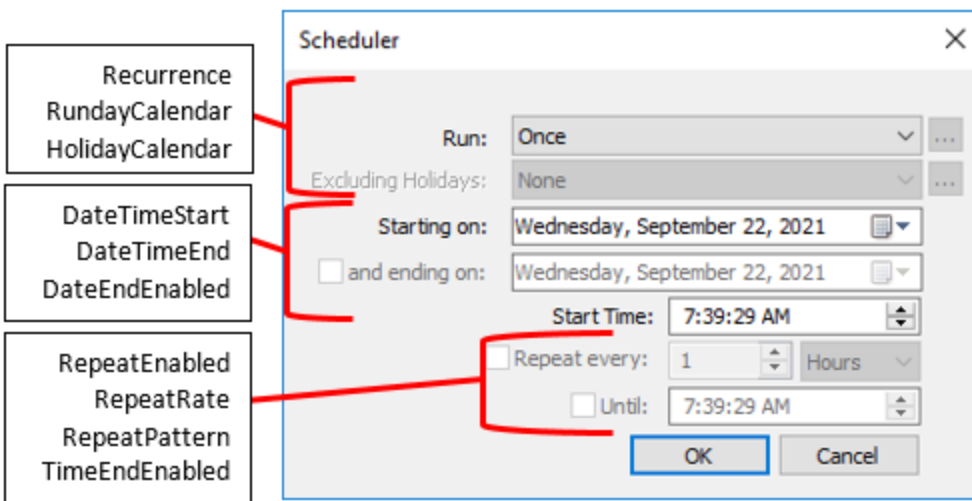
- DailyDayPeriod As Long
- DailyEveryWeekDay As Boolean
- DateEndEnabled As Boolean
- DateTimeEnd As Variant
- DateTimeStart As Variant
- HolidayCalendar As String
- MonthlyDayPeriod As Long
- MonthlyFixedDay As Boolean
- MonthlyMonthPeriod As Long
- MonthlyRelativeWeekday As Relation:
 - Relation_First
 - Relation_Second
 - Relation_Third
 - Relation_Fourth
 - Relation_Last
- RepeatEnabled As Boolean
- RepeatPattern As RepeatPattern:
 - RepeatHours
 - RepeatMinutes
 - RepeatSeconds
- RepeatRate As Long
- RundayCalendar As String
- MonthlyWeekday As Weekday:
 - Weekday_Day = 7
 - Weekday_Friday = 5
 - Weekday_Monday = 1
 - Weekday_Saturday = 6
 - Weekday_Sunday = 0
 - Weekday_Thursday = 4
 - Weekday_Tuesday = 2
 - Weekday_Wednesday = 3
 - Weekday_Weekday = 8
 - Weekday_Weekendday = 9

- TimeEndEnabled As Boolean
- WeeklyFriday As Boolean
- WeeklyMonday As Boolean
- WeeklySaturday As Boolean
- WeeklySunday As Boolean
- WeeklyThursday As Boolean
- WeeklyTuesday As Boolean
- WeeklyWednesday As Boolean
- WeeklyWeekPeriod As Long
- YearlyDayPeriod As Long
- YearlyFixedDay As Boolean
- YearlyMonth As Month:
 - Month_January = 0
 - Month_February = 1
 - Month_March = 2
 - Month_April = 3
 - Month_May = 4
 - Month_June = 5
 - Month_July = 6
 - Month_August = 7
 - Month_September = 8
 - Month_October = 9
 - Month_November = 10
 - Month_December = 11
- YearlyRelativeWeekday As Relation:
 - Relation_First
 - Relation_Second
 - Relation_Third
 - Relation_Fourth
 - Relation_Last
- YearlyWeekday [As Weekday](#)

Dialog Box Equivalents

The ICITimerEventRuleParams interface properties and methods correlate to the following fields and controls in the EFT administration interface. (**DateTimeStart** property,

DateTimeEnd property, **HolidayCalendar**, and **Recurrence** property are used in each.)



Scheduler [X]

Run: Weekly [v] ...

Recur every 1 week(s) on

☒ Sun ☐ Mon ☐ Tues

☐ Wed ☐ Thurs ☐ Fri ☐ Sat

Excluding Holidays: None [v] ...

Starting on: Wednesday, September 22, 2021 [v]

☐ and ending on: Wednesday, September 22, 2021 [v]

Start Time: 7:39:29 AM [v]

☒ Repeat every: 1 [v] Hours [v]

☐ Until: 7:39:29 AM [v]

OK Cancel

WeeklyWeekPeriod

WeeklySunday

WeeklyMonday

WeeklyTuesday

WeeklyWednesday

WeeklyThursday

WeeklyFriday

WeeklySaturday

Scheduler [X]

Run: Monthly [v] ...

☒ Day 1 of every 1 month(s)

☐ The first Sunday of 1 month(s)

Excluding Holidays: None [v] ...

Starting on: Wednesday, September 22, 2021 [v]

☐ and ending on: Wednesday, September 22, 2021 [v]

Start Time: 7:39:29 AM [v]

☒ Repeat every: 1 [v] Hours [v]

☐ Until: 7:39:29 AM [v]

OK Cancel

MonthlyFixedDay (True)

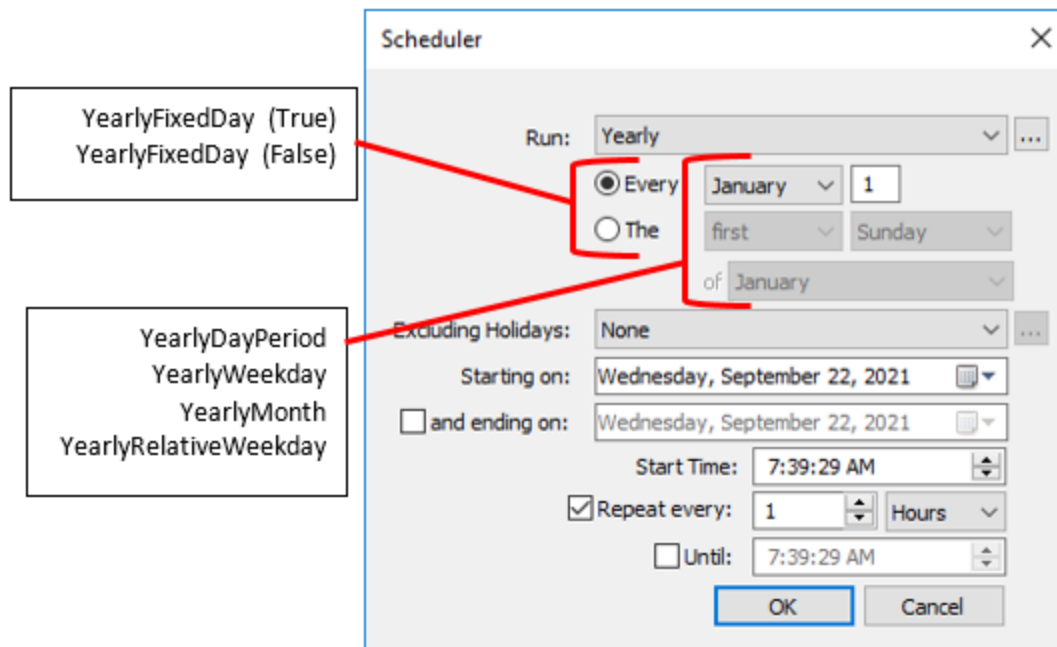
MonthlyFixedDay (False)

MonthlyDayPeriod

MonthlyWeekday

MonthlyRelativeWeekday

MonthlyMonthPeriod



ICITermsOfService Interface

The ICITermsOfService interface provides the properties of the Terms of Service.

Members

- Property **AgreementLabels** As String

Retrieves or modifies Terms of Service labels.

```
HRESULT AgreementLabels([out, retval] BSTR *pVal);
HRESULT AgreementLabels([in] BSTR val);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.AgreementLabels = "C:\toslabels.json"
```

EFT v8 and later

- Property **AgreementPath** As String

Retrieves or modifies Terms of Service agreement path.

```
HRESULT AgreementPath([out, retval] BSTR *pVal);
HRESULT AgreementPath([in] BSTR val);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.AgreementPath = "C:\tos.json"
```

EFT v8 and later

- Property **EffectiveDate** As Variant

Retrieves or modifies Terms of Service effective date.

```
HRESULT EffectiveDate([out, retval] VARIANT *pVal);
HRESULT EffectiveDate([in] VARIANT newVal);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
WScript.echo tos.EffectiveDate
```

EFT v8 and later

- Property **Mode** As [GDPRTermsOfServiceMode](#)

Retrieves or modifies Terms of Service mode.

```
HRESULT Mode([out, retval] GDPRTermsOfServiceMode *pVal);
HRESULT Mode([in] GDPRTermsOfServiceMode newVal);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.Mode = GDPRTermsOfServiceMode_Implied
```

EFT v8 and later

- Property **PreventFTPSUntilAgreed** As Boolean

Retrieves or modifies "Prevent FTPS connection until agreed" option state.

```
HRESULT PreventFTPSUntilAgreed([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreventFTPSUntilAgreed([in] VARIANT_BOOL newVal);
```

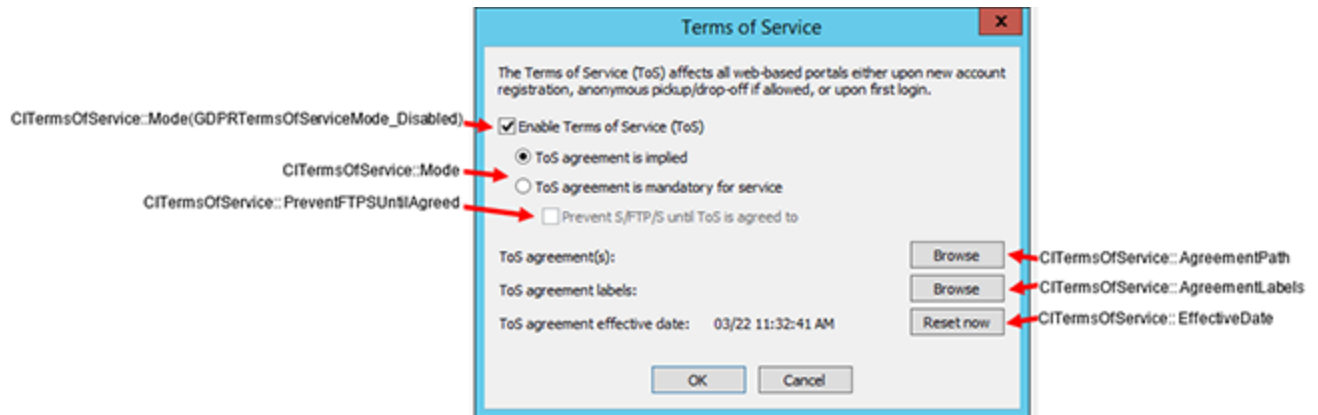
Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.PreventFTPSUntilAgreed =True
```

EFT v8 and later

Dialog Box Equivalents

The Terms of Service dialog box is available from the Site > Web tab.



ICITransferActionParams Interface

The ICITransferActionParams interface allows you to make changes to transfer-related Events, such as when a file is uploaded or downloaded. ICITransferActionParams is a top level IDispatch interface. [ICIUploadActionParams](#) and [ICIDownloadActionParams](#) are subclasses that inherited from the parent ICITransferActionParams IDispatch interface.

```
interface ICITransferActionParams : IDispatch
```

For an example using these properties, refer to the EventRuleCreation script in [Script Examples](#).

Properties

- Property **AutoLogin** As Boolean

Retrieves or specifies whether to automatically login.

```
HRESULT AutoLogin([out, retval] VARIANT_BOOL *pVal);
HRESULT AutoLogin([in] VARIANT_BOOL newVal);
```

EFT v5.2 and later

- Property **ConnectionProfileName** As String

```
HRESULT ConnectionProfileName([out, retval] BSTR *pValName);
HRESULT ConnectionProfileName([in] BSTR newValName);
```

EFT v7.2 and later

- Property **ConnectionRetryAttempts** As Long

Retrieves or specifies the number of times the server will attempt to establish a connection if there are connection problems (e.g., the network is unavailable).

```
HRESULT ConnectionRetryAttempts([out, retval] LONG *pVal);
HRESULT ConnectionRetryAttempts([in] LONG newVal);
```

EFT v6.2.10 and later

- Property **ConnectionTimeoutSeconds** As Long

Retrieves or specifies the number of seconds to wait before connection times out.

```
HRESULT ConnectionTimeoutSeconds([out, retval] LONG *pVal);
HRESULT ConnectionTimeoutSeconds([in] LONG newVal);
```

EFT v6.2.10 and later

- Property **DeleteSourceFile** As Boolean

Retrieves or specifies whether to delete the source file after transfer is complete.

```
HRESULT DeleteSourceFile([out, retval] VARIANT_BOOL *pVal);
HRESULT DeleteSourceFile([in] VARIANT_BOOL newVal);
```

EFT v5.2.5 and later

- Property **DeleteSourceFileIfMoveSkipped** As Boolean

Retrieves or specifies whether the source file should be deleted when an offload (move) or download was skipped due to an existing identical destination file.

```
HRESULT DeleteSourceFileIfMoveSkipped([out, retval] VARIANT_BOOL *pVal);
HRESULT DeleteSourceFileIfMoveSkipped([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If UploadParams.DeleteSourceFileIfMoveSkipped Then
  'Delete is specified
...
End If
```

Specifying:

```
'set "Do not delete source file if move is skipped"
DownloadParams.DeleteSourceFileIfMoveSkipped = False
```

EFT v7 and later

- Property **FTPDataConnectionMode** As [FTPDataConnectionMode](#)

Retrieves or specifies the FTP data connection mode:

FTPMode_PORT (0)—When Port mode is selected, EFT opens an additional port and tells the remote server to connect to <IP:PORT> to establish a data connection. This is useful when the server is behind a firewall that closes all unnecessary ports. If you select this mode, specify the port range from which the client will choose.

FTPMode_PASV (1)—When PASV mode is selected, EFT tells the remote server to provide <IP:PORT> to which EFT can connect to establish a data connection. This is useful when a client is behind a firewall that closes all unnecessary ports. Helps avoid conflicts with security systems.

FTPMode_Auto (2)—EFT initially makes connections in PASV mode. If the PASV connection fails, EFT attempts to connect in PORT mode automatically.

```
HRESULT FTPDataConnectionMode([out, retval]
FTPDataConnectionMode *pVal);
HRESULT FTPDataConnectionMode([in] FTPDataConnectionMode
newVal);
```

EFT v6.2.10 and later

- Property **FTPDataConnectionPortMax** As Long

Retrieves or specifies the upper end of the port range.

```
HRESULT FTPDataConnectionPortMax([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMax([in] LONG newVal);
```

EFT v6.2.10 and later

- Property **FTPDataConnectionPortMin** As Long

Retrieves or specifies the lower end of the port range.

```
HRESULT FTPDataConnectionPortMin([out, retval] LONG *pVal);
HRESULT FTPDataConnectionPortMin([in] LONG newVal);
```

EFT v6.2.10 and later

- Property **FTPSClearCommandChannel** As Boolean

Retrieves or specifies whether EFT sends FTP commands in clear text. Only available only when FTPS is specified.

```
HRESULT FTPSClearCommandChannel([out, retval] VARIANT_BOOL
    *pVal);
HRESULT FTPSClearCommandChannel([in] VARIANT_BOOL newVal);
```

EFT v6.2.10 and later

- Property **FTPSClearDataChannel** As Boolean

Retrieves or specifies whether EFT transfers files without encryption. Only available only when FTPS is specified.

```
HRESULT FTPSClearDataChannel([out, retval] VARIANT_BOOL
    *pVal);
HRESULT FTPSClearDataChannel([in] VARIANT_BOOL newVal);
```

EFT v6.2.10 and later

- Property **Host** As String

Retrieves or specifies the host address.

```
HRESULT Host([out, retval] BSTR *pVal);
HRESULT Host([in] BSTR newVal);
```

EFT v5.2 and later

- Property **LocalPath** As String

Retrieves or specifies the local path.

```
HRESULT LocalPath([out, retval] BSTR *pVal);
HRESULT LocalPath([in] BSTR newVal);
```

EFT v5.2 and later

- Property **MaxConcurrentThreads** As Long

Retrieves or specifies the maximum number of concurrent threads.

```
HRESULT MaxConcurrentThreads([out, retval] LONG *pVal);
HRESULT MaxConcurrentThreads([in] LONG newVal);
```

EFT v6.2.10 and later

- Property **OverwriteType** As [TransferOverwriteType](#)

Retrieves or specifies the transfer overwrite type.

```
HRESULT OverwriteType([out, retval] TransferOverwriteType
    *pVal);
HRESULT OverwriteType([in] TransferOverwriteType newVal);
```

Example (VBScript):

Retrieving:

```
if TransferOTOverwrite = oTA.OverwriteType then
MsgBox "OverwriteType: Always overwrite"
end if
```

Specifying:

```
oTA.OverwriteType = TransferOTSmart
```

EFT v6.3 and later

- Property **Password** As String

Retrieves or specifies the password.

```
HRESULT Password([out, retval] BSTR *pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v5.2 and later

- Property **Port** As Long

Retrieves or specifies the port.

```
HRESULT Port([out, retval] long *pVal);
HRESULT Port([in] long newVal);
```

EFT v5.2 and later

- Property **PortAsString** As String

```
HRESULT PortAsString([out, retval] BSTR *pVal);
HRESULT PortAsString([in] BSTR newVal);
```

EFT v7.4.6 and later

- Property **PostTransferCommandParam** As string

Specify post transfer command parameters.

```
HRESULT PostTransferCommandParam([out, retval] BSTR *pVal);
HRESULT PostTransferCommandParam([in] BSTR newVal);
```

EFT v7.1.1 and later

- Property **PostTransferCommandType** As string

Specify post transfer command type.

```
HRESULT PostTransferCommandType([out, retval]
PrePostTransferCommand *pVal);
HRESULT PostTransferCommandType([in] PrePostTransferCommand
newVal);
```

EFT v7.1.1 and later

- Property **PreserveFileTime** As Boolean

Retrieves or specifies whether to keep a file's time stamp that same on the local computer as on the remote server.

Use **ICIDownloadActionParams::PreserveTimeStamp** to keep a *downloaded* file's time stamp the same on the local computer as it is on the remote server.

Use **ICIUploadActionParams::PreserveTimeStamp** to keep an *uploaded* file's time stamp the same on EFT as it is on the local computer. (Not applicable to SFTP.)

```
HRESULT PreserveFileTime([out, retval] VARIANT_BOOL *pVal);
HRESULT PreserveFileTime([in] VARIANT_BOOL newVal);
```

EFT v6.2.10 and later

- Property **PreTransferCommandParam** As string

Specify pretransfer command parameters.

```
HRESULT PreTransferCommandParam([out, retval] BSTR *pVal);
HRESULT PreTransferCommandParam([in] BSTR newVal);
```

EFT v7.1.1 and later

- Property **PreTransferCommandType** As string

Specify pretransfer command type.

```
HRESULT PreTransferCommandType([out, retval]
PrePostTransferCommand *pVal);
HRESULT PreTransferCommandType([in] PrePostTransferCommand
newVal);
```

EFT v7.1.1 and later

- Property **PrivateKeyPassword** As String

Retrieves or specifies the private key password.

```
HRESULT PrivateKeyPassword([out, retval] BSTR *pVal);
HRESULT PrivateKeyPassword([in] BSTR newVal);
```

EFT v6.0 and later

- Property **PrivateKeyPath** As String

Retrieves or specifies the private key path.

```
HRESULT PrivateKeyPath([out, retval] BSTR *pVal);
HRESULT PrivateKeyPath([in] BSTR newVal);
```

EFT v6.0 and later

- Property **Protocol** As [NetworkProtocol](#)

Retrieves or specifies the network protocol.

```
HRESULT Protocol([out, retval] NetworkProtocol *pVal);
HRESULT Protocol([in] NetworkProtocol newVal);
```

EFT v5.2 and later

- Property **ProxyAdvCustomCmds** As String

(In v6.3.0, the "y" in "Proxy" was missing. Please update your scripts accordingly.)

Retrieves or specifies whether to use proxy commands.

```
HRESULT ProxyAdvCustomCmds([out, retval] BSTR *pVal);
HRESULT ProxyAdvCustomCmds([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Custom commands is: " & oTA.ProxyAdvCustomCmds
```

Specifying:

```
oTA.ProxyAdvCustomCmds = ""
```

EFT v6.3 and later

- Property **ProxyAdvSeparator** As String

(In v6.3.0, the "y" in "Proxy" was missing. Please update your scripts accordingly.)

Retrieves or/specifies separator for username and Site name in case of ProxyAuthUserAtSite authentication.

```
HRESULT ProxyAdvSeparator([out, retval] BSTR *pVal);
HRESULT ProxyAdvSeparator([in] BSTR newVal);
```

ProxyAuthenticationType

- ProxyAuthCustom = 5
- ProxyAuthOpenSite = 4
- ProxyAuthSite = 1
- ProxyAuthUserAtSite = 0
- ProxyAuthUserPassAcct = 3
- ProxyAuthUserWithLogon = 2

Example (VBScript):**Retrieving:**

```
if ProxyAuthSite = oTA.ProxyAdvAuthenticationType then
' some code
end if
```

Specifying:

```
oTA.ProxyAdvAuthenticationType = ProxyAuthUserWithLogon
```

EFT v6.3 and later

- Property **ProxyAuthenticationType** As [ProxyAuthenticationType](#)

(In v6.2.31 - 6.3.0, this property was **ProxyAdvAuthenticationType**. Please update your scripts accordingly.)

Retrieves or specifies the proxy authentication type.

```
HRESULT ProxyAuthenticationType([out, retval]
ProxyAuthenticationType *pVal);
HRESULT ProxyAuthenticationType([in] ProxyAuthenticationType
newVal);
```

Example (VBScript):

Retrieving:

```
if ProxyAuthSite = oTA.ProxyAuthenticationType then
' some code
end if
```

Specifying:

```
oTA.ProxyAdvAuthenticationType = ProxyAuthUserWithLogon
```

EFT v6.3 and later

- Property **ProxyHostName** As String

Retrieves or specifies the proxy host name or IP address.

```
HRESULT ProxyHostName([out, retval] BSTR *pVal);
HRESULT ProxyHostName([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Proxy host is: " & oTA.ProxyHostName
```

Specifying:

```
oTA.ProxyHostName = "192.168.0.5"
```

EFT v6.3 and later

- Property **ProxyPassword** As String

Retrieves or specifies the proxy password.

```
HRESULT ProxyPassword([out, retval] BSTR *pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy user name is: " & oTA.ProxyPassword
```

Specifying:

```
oTA.ProxyPassword = "swordfish"
```

EFT v6.3 and later

- Property **ProxyPort** As Long

Retrieves or specifies the proxy port.

```
HRESULT ProxyPort([out, retval] LONG *pVal);
HRESULT ProxyPort([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy port is: " & oTA.ProxyPort
```

Specifying:

```
oTA.ProxyPort = 8080
```

EFT v6.3 and later

- Property **ProxyType** As [ProxyType](#)

Retrieves or specifies the proxy type.

NOTE: You must set CIUploadActionParams.[UseProxy](#) = true before you set the CIUploadActionParams.proxyType property,

```
HRESULT ProxyType([out, retval] ProxyType *pVal);
HRESULT ProxyType([in] ProxyType newVal);
```

Example (VBScript):**Retrieving:**

```
if ProxyTypeDMZ = oTA.ProxyType then
  MsgBox "DMZ proxy is selected."
end if
```

Specifying:

```
oTA.ProxyType = ProxyTypeHTTP
```

EFT v6.3 and later

- Property **ProxyUserName** As String

Retrieves or specifies the proxy user name.

```
HRESULT ProxyUserName([out, retval] BSTR *pVal);
HRESULT ProxyUserName([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Proxy user name is: " & oTA.ProxyUserName
```

Specifying:

```
oTA.ProxyUserName = "User_1"
```

EFT v6.3 and later

- Property **PublicKeyPath** As String

Retrieves or specifies the public key path.

```
HRESULT PublicKeyPath([out, retval] BSTR *pVal);
HRESULT PublicKeyPath([in] BSTR newVal);
```

EFT v6.0 and later

- Property **RemotePath** As String

Retrieves or specifies the remote path.

```
HRESULT RemotePath([out, retval] BSTR *pVal);
HRESULT RemotePath([in] BSTR newVal);
```

EFT v5.2 and later

- Property **RenameAfterTransfer** As Boolean

Specify whether to rename a file after it is transferred.

```
HRESULT RenameAfterTransfer([out, retval] VARIANT_BOOL
*pVal);
HRESULT RenameAfterTransfer([in] VARIANT_BOOL newVal);
```

EFT v6.4 and later

- Property **RenameAfterTransferTo** As String

Specify a new name for a transferred file.

```
HRESULT RenameAfterTransferTo([out, retval] BSTR *pVal);
HRESULT RenameAfterTransferTo([in] BSTR newVal);
```

EFT v6.4 and later

- Property **RetryDelaySeconds** As Long

Specify the number of seconds to wait before retrying a failed transfer.

```
HRESULT RetryDelaySeconds([out, retval] LONG *pVal);
HRESULT RetryDelaySeconds([in] LONG newVal);
```

EFT v6.2.10 and later

- Property **SOCKSHostName** As String

Retrieves or specifies the SOCKS host name.

```
HRESULT SOCKSHostName([out, retval] BSTR *pVal);
HRESULT SOCKSHostName([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SOCKS server host name is: " & oTA.SOCKSHostName
```

Specifying:

```
oTA.SOCKSHostName = "192.168.80.16"
```

EFT v6.3 and later

- Property **SOCKSPassword** As String

Retrieves or specifies the SOCKS password.

```
HRESULT SOCKSPassword([out, retval] BSTR *pVal);
HRESULT SOCKSPassword([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Use SOCKS5 password: " & oTA.SOCKSPassword
```

Specifying:

```
oTA.SOCKSPassword = "swordfish"
```

EFT v6.3 and later

- Property **SOCKSPort** As Long

Retrieves or specifies the SOCKS port.

```
HRESULT SOCKSPort([out, retval] LONG *pVal);
HRESULT SOCKSPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SOCKS server port is: " & CStr(oTA.SOCKSPort)
```

Specifying:

```
oTA.SOCKSPort = 1080
```

EFT v6.3 and later

- Property **SOCKSType** As [SOCKSType](#)

Retrieves or specifies the type of SOCKS server.

```
HRESULT SOCKSType([out, retval] SOCKSType *pVal);
HRESULT SOCKSType([in] SOCKSType newVal);
```

Example (VBScript):

Retrieving:


```
if SOCKS5 = oTA.SOCKSType then
MsgBox "SOCKS 5"
end if
```

Specifying:

```
oTA.SOCKSType = SOCKS4
```

EFT v6.3 and later

- Property **SOCKSUseAuthentication** As Boolean

Retrieves or specifies whether to use authentication on a SOCKS server.

```
HRESULT SOCKSUseAuthentication([out, retval] VARIANT_BOOL
*pVal);
HRESULT SOCKSUseAuthentication([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Use SOCKS5 authentication: " & CStr
(oTA.SOCKSUseAuthentication)
```

Specifying:

```
oTA.SOCKSUseAuthentication = False
```

EFT v6.3 and later

- Property **SOCKSUserName** As String

Retrieves or specifies the SOCKS username when SOCKS

```
HRESULT SOCKSUserName([out, retval] BSTR *pVal);
HRESULT SOCKSUserName([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Use SOCKS5 user name: " & oTA.SOCKSUserName
```

Specifying:

```
oTA.SOCKSUserName = "User_1"
```

EFT v6.3 and later

- Property **TextFileTypes** As String

Retrieves or specifies the file types that will be transferred and text files (ASCII).

```
HRESULT TextFileTypes([out, retval] BSTR *pVal);
HRESULT TextFileTypes([in] BSTR newVal);
```

EFT v6.2.10 and later

- Property **TransferEncoding** As [TransferEncodingType](#)

Retrieves or specifies the transfer encoding mode for client transfer (upload/download) Actions.

```
HRESULT TransferEncoding([out, retval] TransferEncodingType
*pVal);
HRESULT TransferEncoding([in] TransferEncodingType newVal);
```

Example (VBScript):

Retrieving:

```
If UploadParams.TransferEncoding = 0 Then
  'Auto-detect
End If
```

Specifying:

```
'set Auto-detect mode for Download action
DownloadParams.TransferEncoding = 0
```

EFT v6.5 and later

- Property **TreatMissingSourceAsSuccess** As Boolean

Specify whether a missing source file is treated as a success (TRUE) or failure (FALSE).

```
HRESULT TreatMissingSourceAsSuccess([out, retval] VARIANT_
BOOL *pVal);
HRESULT TreatMissingSourceAsSuccess([in] VARIANT_BOOL
newVal);
```

EFT v6.4 and later

- Property **TrustedPublicKeyPath** As string

Specify path to trusted public key.

```
HRESULT TrustedPublicKeyPath([out, retval] BSTR *pVal);
HRESULT TrustedPublicKeyPath([in] BSTR newVal);
```

EFT v7.1.1 and later

- Property **UseLocalCodePage** As Boolean

```
HRESULT UseLocalCodePage([out, retval] VARIANT_BOOL *pVal);
HRESULT UseLocalCodePage([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property **UseProxy** As Boolean

Retrieves or specifies whether to use a proxy server for the transfer.

NOTE: You must set `CIUploadActionParams.UseProxy = true` before you set the `CIUploadActionParams.proxyType` property.

```
HRESULT UseProxy([out, retval] VARIANT_BOOL *pVal);
HRESULT UseProxy([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
if True = oTA.UserProxy then
  MsgBox "Proxy is using."
end if
```

Specifying:

```
oTA.UserProxy = False
```

EFT v6.3 and later

- Property **User** As String

Retrieves or specifies username.

```
HRESULT User([out, retval] BSTR *pVal);
HRESULT User([in] BSTR newVal);
```

EFT v5.2 and later

- Property **UseSOCKS** As Boolean

Retrieves or specifies whether to use SOCKS.

```
HRESULT UseSOCKS([out, retval] VARIANT_BOOL *pVal);
HRESULT UseSOCKS([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Using SOCKS: " & CStr(oTA.UseSOCKS)
```

Specifying:

```
oTA.UseSOCKS = False
```

EFT v6.3 and later

- Property **ValidateIntegrity** As Boolean

Specify that EFT should double check binary files to ensure the files downloaded completely and correctly. (Not applicable to SFTP.)

```
HRESULT ValidateIntegrity([out, retval] VARIANT_BOOL *pVal);
HRESULT ValidateIntegrity([in] VARIANT_BOOL newVal);
```

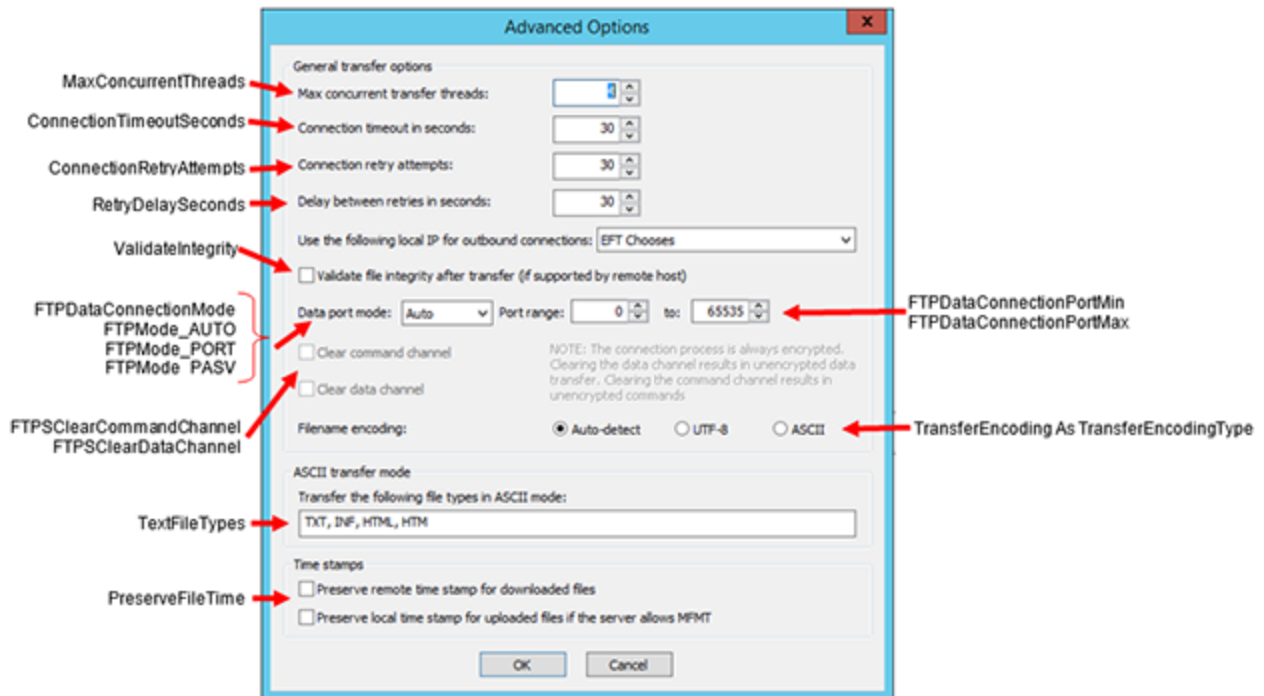
EFT v6.2.10 and later

Dialog Box Equivalents

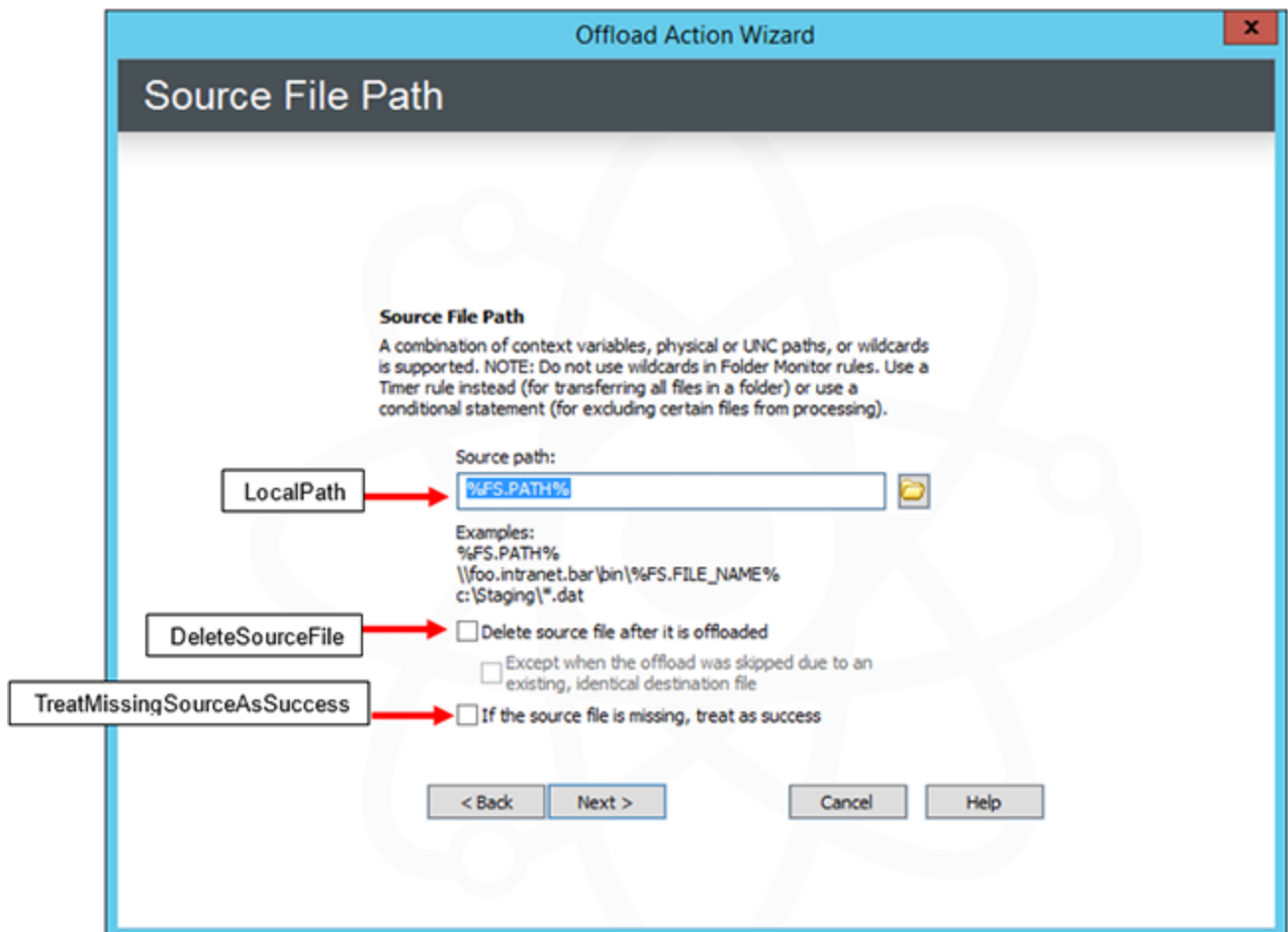
The ICITransferActionParams interface properties and methods correlate to the following fields and controls in the EFT administration interface.

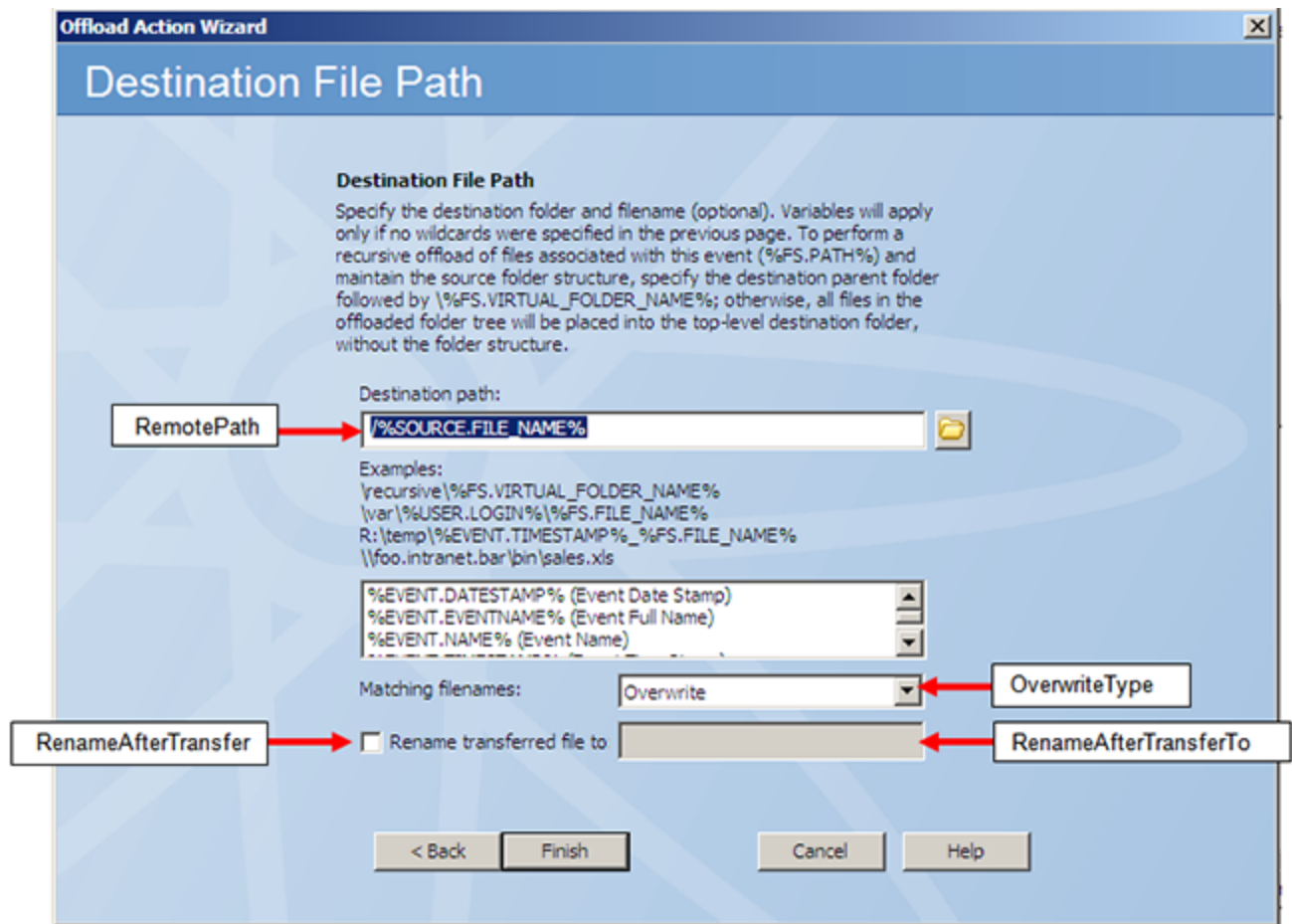
Advanced Options

Several API properties were introduced in EFT v6.2.10 with which you can specify the advanced options in a File Download or File Offload Event Rule Action. (To view the Advanced Options dialog box in the EFT administration interface, in the [File Download Configuration wizard](#) or the [File Offload Configuration wizard](#), click Advanced.)



Copy/Move (Offload) Action Wizard





ICIUploadActionParams

The ICIUploadActionParams interface (derived from [ICITransferActionParams](#) defining client operation parameters) allows you to make changes to the Event Rule settings.

```
interface ICIUploadActionParams : ICITransferActionParams
```

Refer to [ICITransferActionParams](#) for details of the methods and properties.

ICIUserAccountActionParams

The UserAccountActionParams interface is used to specify parameters for the User Account Action.

Members

- Property **Action As** [UserAccountActionType](#)

```
HRESULT Action([out, retval] UserAccountActionType *pVal);
HRESULT Action([in] UserAccountActionType newVal);
```

EFT v7.4.13

- Property **Parameter** As Variant

```
HRESULT Parameter([out, retval] VARIANT *pVal);
HRESULT Parameter([in] VARIANT newVal);
```

EFT v7.4.13

ICIWebServiceActionParams

The ICIWebServiceActionParams interface properties are used to define the Invoke Web Service Action.

- Property **AdvancedHomeIP** As String

```
HRESULT AdvancedHomeIP([out, retval] BSTR* pVal);
HRESULT AdvancedHomeIP([in] BSTR newVal);
```

EFT v7.2

- Property **AdvancedRetries** As Int

```
HRESULT AdvancedRetries([out, retval] int* pVal);
HRESULT AdvancedRetries([in] int newVal);
```

EFT v7.2

- Property **AdvancedRetryDelayInSec** As Int

```
HRESULT AdvancedRetryDelayInSec([out, retval] int* pVal);
HRESULT AdvancedRetryDelayInSec([in] int newVal);
```

EFT v7.2

- Property **AdvancedTimeoutInSec** As Int

```
HRESULT AdvancedTimeoutInSec([out, retval] int* pVal);
HRESULT AdvancedTimeoutInSec([in] int newVal);
```

EFT v7.2

- Property **BodyContent**

```
HRESULT BodyContent([out, retval] BSTR* pVal);
HRESULT BodyContent([in] BSTR newVal);
```

EFT v7.2

- Property **BodyFile** As String

```
HRESULT BodyFile([out, retval] BSTR* pVal);
HRESULT BodyFile([in] BSTR newVal);
```

EFT v7.2

- Property **Cookies** As String

```
HRESULT Cookies([out, retval] BSTR* pVal);
HRESULT Cookies([in] BSTR newVal);
```

EFT v7.2

- Property **EncodeBodyEditAsJson** As Boolean

```
HRESULT EncodeBodyEditAsJson([out, retval] VARIANT_BOOL* pVal);
HRESULT EncodeBodyEditAsJson([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **ForceBasicAuth**(val As [SFTPAdvBool](#))

```
HRESULT ForceBasicAuth([out, retval] SFTPAdvBool *pVal);
HRESULT ForceBasicAuth([in] SFTPAdvBool newVal);
```

EFT v7.2

- Property **Headers** As String

```
HRESULT Headers([out, retval] BSTR* pVal);
HRESULT Headers([in] BSTR newVal);
```

EFT v7.2

- Property **LoadBodyFromFile** As Boolean

```
HRESULT LoadBodyFromFile([out, retval] VARIANT_BOOL* pVal);
HRESULT LoadBodyFromFile([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **Password** As String

```
HRESULT Password([out, retval] BSTR* pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v7.2

- Property **ProxyEnabled** As Boolean

```
HRESULT ProxyEnabled([out, retval] VARIANT_BOOL* pVal);
HRESULT ProxyEnabled([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **ProxyHost** As String

```
HRESULT ProxyHost([out, retval] BSTR* pVal);
HRESULT ProxyHost([in] BSTR newVal);
```

EFT v7.2

- Property **ProxyPassword** As String

```
HRESULT ProxyPassword([out, retval] BSTR* pVal);
HRESULT ProxyPassword([in] BSTR newVal);
```

EFT v7.2

- Property **ProxyPort** As Int

```
HRESULT ProxyPort([out, retval] int* pVal);
HRESULT ProxyPort([in] int newVal);
```

EFT v7.2

- Property **ProxyUser** As String

```
HRESULT ProxyUser([out, retval] BSTR* pVal);
HRESULT ProxyUser([in] BSTR newVal);
```

EFT v7.2

- Property **ResponseFile** As String

```
HRESULT ResponseFile([out, retval] BSTR* pVal);
HRESULT ResponseFile([in] BSTR newVal);
```

EFT v7.2

- Property **ResponseVariable** As String

```
HRESULT ResponseVariable([out, retval] BSTR* pVal);
HRESULT ResponseVariable([in] BSTR newVal);
```

EFT v7.2

- Property **SaveResponseToFile** As Boolean

```
HRESULT SaveResponseToFile([out, retval] VARIANT_BOOL*
pVal);
HRESULT SaveResponseToFile([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **SaveResponseToVariable** As Boolean

```
HRESULT SaveResponseToVariable([out, retval] VARIANT_BOOL*
pVal);
HRESULT SaveResponseToVariable([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **SOCKSEnabled** As Boolean

```
HRESULT SOCKSEnabled([out, retval] VARIANT_BOOL* pVal);
HRESULT SOCKSEnabled([in] VARIANT_BOOL newVal);
```

EFT v7.2

- Property **SOCKSHost** As String

```
HRESULT SOCKSHost([out, retval] BSTR* pVal);
HRESULT SOCKSHost([in] BSTR newVal);
```

EFT v7.2

- Property **SOCKSPassword** As String

```
HRESULT SOCKSPassword([out, retval] BSTR* pVal);
HRESULT SOCKSPassword([in] BSTR newVal);
```

EFT v7.2

- Property **SOCKSPort** As Int

```
HRESULT SOCKSPort([out, retval] int* pVal);
HRESULT SOCKSPort([in] int newVal);
```

EFT v7.2

- Property **SOCKSType** As Int

```
HRESULT SOCKSType([out, retval] int* pVal);
HRESULT SOCKSType([in] int newVal);
```

EFT v7.2

- Property **SOCKSUseAuth** As Boolean

```
HRESULT SOCKSUseAuth([out, retval] VARIANT_BOOL* pVal);
HRESULT SOCKSUseAuth([in] VARIANT_BOOL newVal);
```

- Property **SOCKSUser** As String

```
HRESULT SOCKSUser([out, retval] BSTR* pVal);
HRESULT SOCKSUser([in] BSTR newVal);
```

EFT v7.2

- Property **SslClientCertificate** As String

```
HRESULT SslClientCertificate([out, retval] BSTR* pVal);
HRESULT SslClientCertificate([in] BSTR newVal);
```

EFT v7.2

- Property **SslPassphrase** As String

```
HRESULT SslPassphrase([out, retval] BSTR* pVal);
HRESULT SslPassphrase([in] BSTR newVal);
```

EFT v7.2

- Property **SslPrivateKey** As String

```
HRESULT SslPrivateKey([out, retval] BSTR* pVal);
HRESULT SslPrivateKey([in] BSTR newVal);
```

EFT v7.2

- Property **SslServerCertificate** As String

```
HRESULT SslServerCertificate([out, retval] BSTR* pVal);
HRESULT SslServerCertificate([in] BSTR newVal);
```

EFT v7.2

- Property **Url** As String

```
HRESULT Url([out, retval] BSTR
HRESULT Url([in] BSTR newVal);
```

EFT v7.2

- Property **UserName** As String

```
HRESULT UserName([out, retval] BSTR* pVal);
HRESULT UserName([in] BSTR newVal);
```

EFT v7.2

- Property **Verb** As [WebServiceVerbType](#)

```
HRESULT Verb([out, retval] WebServiceVerbType* pVal);
HRESULT Verb([in] WebServiceVerbType newVal);
```

EFT v7.2

ICIWindowsEventLogActionParams

The ICIWindowsEventLogActionParams interface properties are listed below.

```
interface ICIWindowsEventLogActionParams : IDispatch
```

Properties

- Property **Description** As String

Retrieves or specifies the description.

```
HRESULT Description([out, retval] BSTR* pVal);
HRESULT Description([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Event is: " & oWEL.Description
```

Specifying:

```
oWEL.Description = "Some event"
```

EFT v6.3 and later

- Property **EventID** As Long

Retrieves or specifies the event log ID.

```
HRESULT EventID([out, retval] LONG* pVal);
HRESULT EventID([in] LONG newVal);
```

EFT v6.4 and later

- Property **type** As [EventLogType](#)

Retrieves or specifies the event log type.

```
HRESULT Type([out, retval] EventLogType* pVal);  
HRESULT Type([in] EventLogType newVal);
```

Example (C#):

```
eventLogActionParams.type = EventLogType.EventError;
```

Example (VBScript):

Retrieving:

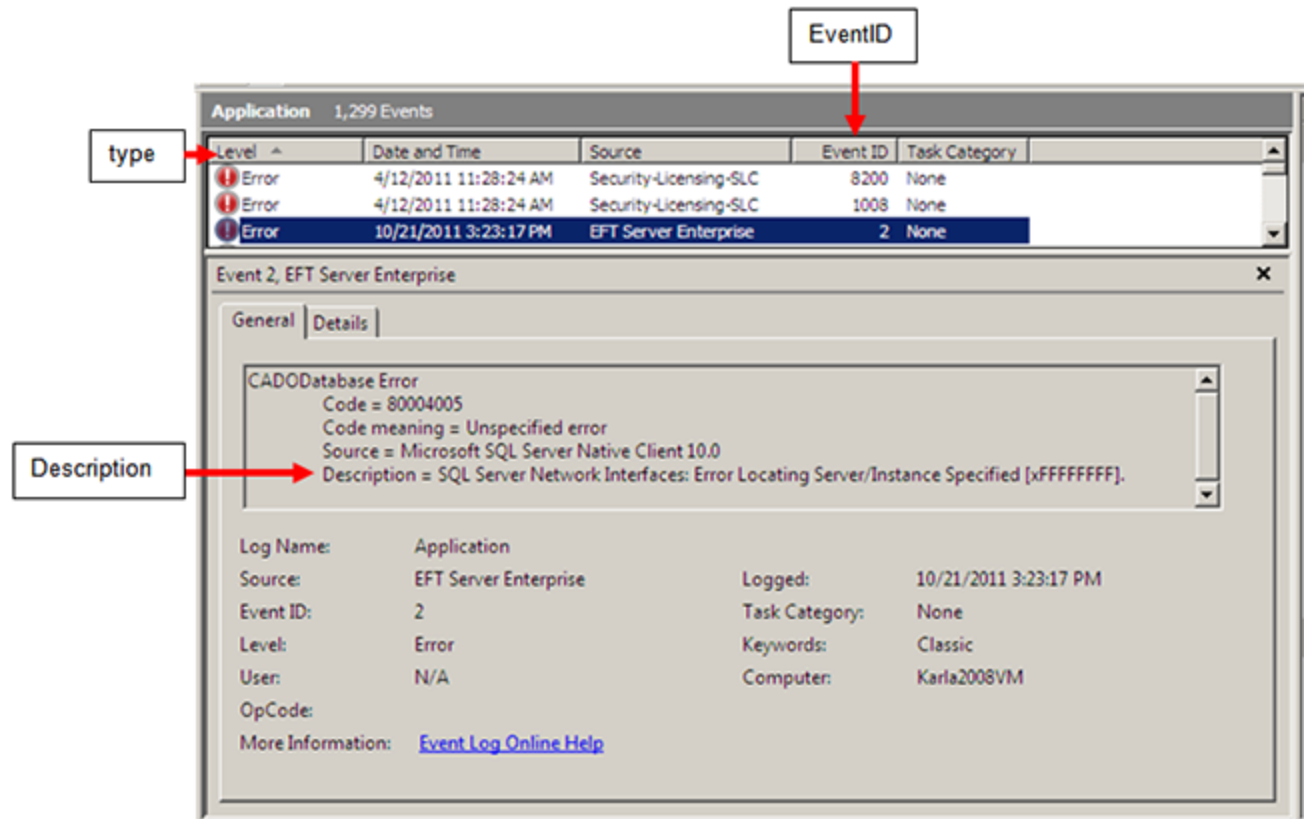
```
if EventInformation = oWEL.Type then  
'...  
end if
```

Specifying:

```
oWEL.Type = EventError
```

EFT v6.3 and later

Dialog Box Equivalents



High Availability-Related Objects

This chapter contains HA-related COM objects.

ICHighAvailabilityParams Interface

The ICHighAvailabilityParams interface contains parameters defined for a high availability deployment.

- Property **CoherenceQueue** As String

```
HRESULT CoherenceQueue([out, retval] BSTR *pVal);
HRESULT CoherenceQueue([in] BSTR newVal);
```

- Property **DefaultNodeList**

Allows you to specify one or more nodes that will execute Event Rules. This list can be overridden in the Event Rules.

```
HRESULT DefaultNodeList([out, retval] VARIANT *pVal);
HRESULT DefaultNodeList([in] VARIANT pVal);
```

- Property **Enabled** As Boolean

```
HRESULT Enabled([out, retval] VARIANT_BOOL *pVal);
HRESULT Enabled([in] VARIANT_BOOL newVal);
```

- Property **EventQueue** As String

```
HRESULT EventQueue([out, retval] BSTR *pVal);
HRESULT EventQueue([in] BSTR newVal);
```

- Property **MulticastAddress** As String

```
HRESULT MulticastAddress([out, retval] BSTR *pVal);
HRESULT MulticastAddress([in] BSTR newVal);
```

- Property **MulticastPortAs** Long

```
HRESULT MulticastPort([out, retval] long *pVal);
HRESULT MulticastPort([in] long newVal);
```

- Property **SharedConfigurationPath** As String

```
HRESULT SharedConfigurationPath([out, retval] BSTR* pVal);
HRESULT SharedConfigurationPath([in] BSTR newVal);
```

ICIRoundRobinEventRuleSchedulingParams Interface

The ICIRoundRobinEventRuleSchedulingParams interface provides the methods and properties used for scheduling the "round robin" process for Event Rules in a high availability, active-active cluster.

- Sub **Add**(newVal As String)

This function adds a new node entry to the run on list for the event rule.

```
HRESULT Add([in] BSTR newVal);
```

EFT v7.0 and later

- Function **Count()** As Long

This function returns the number of nodes in the event rule's run on list.

```
HRESULT Count([out, retval] long* plCount);
```

EFT v7.0 and later

- Sub **Delete**(lIndex As Long)

This function removes the node entry at the specified list index.

```
HRESULT Delete([in] long lIndex);
```

EFT v7.0 and later

- Function **Item**(lIndex As Long) As String

This function returns the node name at the specified list index.

```
HRESULT Item([in] long lIndex, [out, retval] BSTR *pVal);
```

EFT v7.0 and later

IP Access Rule-Related Objects

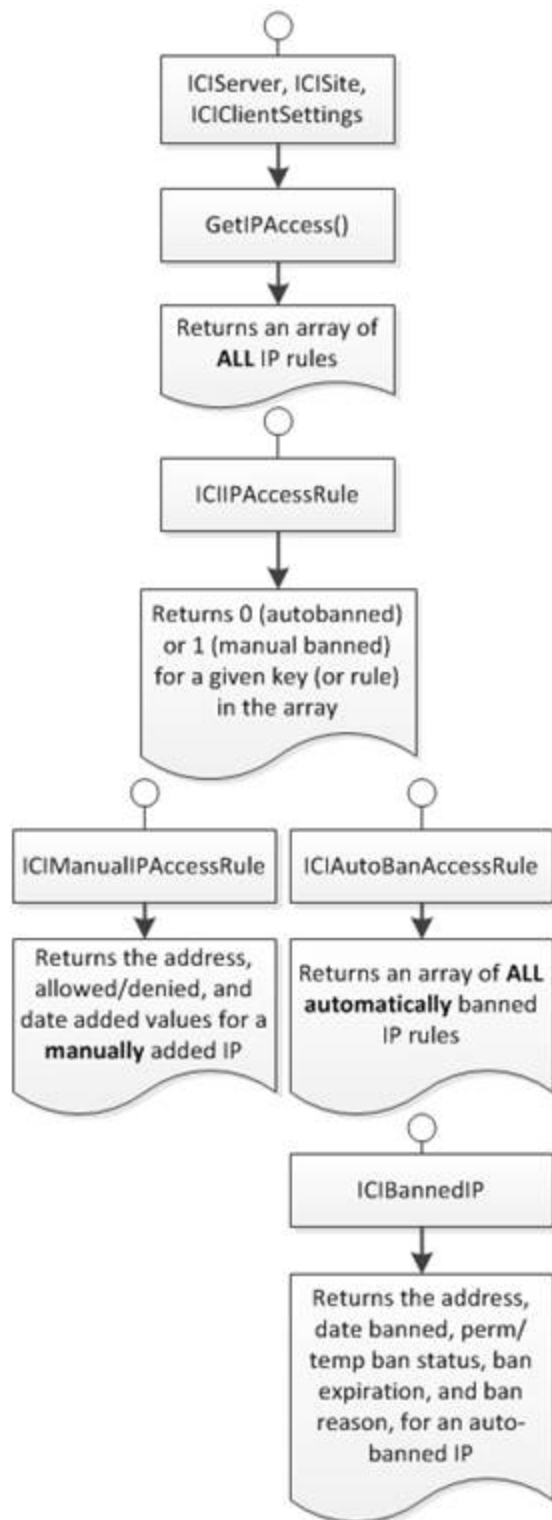
This chapter contains IP Access Rule-related COM objects. (Also see the IP access ban settings in the [ICIServer interface](#) (for remote administration), [ICISite Interface](#), and [ICIClientSettings Interface](#).)

How Do the IP Access/Ban Methods and Properties Work?

Several interfaces, methods, properties, and an enum can be used to retrieve and set IP addresses that are denied or allowed access. The diagram and description below describe how the interfaces, methods, properties, and enum all work together. Scroll down to **Related Topics** for links to the IP-related interfaces, methods, properties, and enum. For an example of an IP ban access script, refer to [Script Examples: UserIPBanEntry](#).

- The ICIServer, ICISite, or ICIClientSettings interfaces each provide a GetIPAccess() method to retrieve an array (list) of IP access rules, regardless of how those rules were created, whether automatically due to auto-ban, or manually with deny/allow assigned to an IP address. The array of IP rules returned by GetIPAccess() will be different depending on whether you invoke the method under the ICIServer, ICISite, or ICIClientSettings interface.
- ICIIIPAccessRule is the parent interface for managing IP access rules. It returns the IP access rule **type** (manual or auto) for a given rule in the array of IP rules obtained using GetIPAccess(). The available types are **0** for auto-banned IPs, **1** for manually added IPs (which could be banned or allowed).
- Two interfaces inherit from ICIIIPAccessRule: ICIManualIPAccessRule and ICIAutobanIPAccessRule.

- **ICIManualIPAccessRule** is used to obtain the properties of an IP rule that was **manually** added to the IP access rule set. Those properties include the IP address, whether the IP is allowed or denied, and the date the IP was added to the IP access rule set.
- **ICIAutobanIPAccessRule** is used to obtain an array of **automatically** banned IPs as a subset of the entire array of IP access rules.
- After you have the ICIAutobanIPAccessRule array, you use ICIBannedIP to obtain the properties for each of those **auto**-banned IP rules, including the IP address, the date banned, whether the ban is permanent or temporary ban, the ban expiration (if applicable), and ban reason.



- [ICIAutobanIPAccessRule](#) - Provides the Autoban IP access rule BannedIPs property
- [ICIBannedIP](#) - Represents an entry in an IP address autoban list
- [ICIClientSettings](#)

- [AddIPAccessRule](#) - Adds allowed (true) or denied (false) IP mask for the client
- [GetIPAccessRules](#) - Retrieves an array of client/template IP Access rules
- [GetLimitAccessByIP](#) - Determines if the IP Access Restriction List is enabled on the Settings Template or user account
- [IPAccessAllowedDefault](#) - Set or retrieve whether user or Settings Template IP access settings allow or deny access by default (if no rule match)
- [MoveIPAccessRule](#) - Change allowed IP access rule position
- [ICIManualIPAccessRule](#) - Represents an IP address access rule that was added manually
- [ICIServer](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for administration
 - [GetAdminIPs](#) - Retrieve an array of IP addresses for the computer on which the Server is running
 - [GetIPAccessRules](#) - Retrieve an array of administration IP address access rules
 - [GetLocalIP](#) - Given an index, retrieves a Local IP address
 - [IPAccessAllowedDefault](#) - Retrieve or specify the default IP Access restriction list
 - [ListenIP](#) - Retrieve or specify the server's IP address
 - [ListenIPs](#) - Retrieve or specify the IP addresses on which the server listens
 - [MoveIPAccessRule](#) - Change allowed administration IP access rule position.
 - [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for administration
 - [UnbanIP](#) - Remove denied IP mask from Autoban list for administration
- [ICISite](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for the Site
 - [AutoBanIPsPermanently](#) - Retrieve or specify whether to automatically ban IP addresses permanently
 - [BanIPForInvalidLoginAttempts](#) - Retrieve or specify whether to ban an IP address after excessive invalid login attempts with existing username and wrong password.
 - [BanIPForInvalidLoginAttemptsWithExistingUsername](#) - When banning an IP address for invalid login attempts, specify whether to count incorrect username and correct username + incorrect password or only incorrect username attempts.
 - [BanIPOnDOS](#) - Retrieve or specify whether to ban an IP address after excessive invalid commands
 - [GetIP](#) - Retrieves the index of the Site Listen IP

- [GetIPAccessRules](#) - Retrieves an array of allowed IP masks.
- [IPAccessAllowedDefault](#) - Retrieve or specify whether IP access is allowed by default
- [ListenIPs](#) - Retrieve or specify the IP addresses on which the Site listens
- [MoveIPAccessRule](#) - Change allowed or denied IP mask position for the Site
- [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for the Site
- [SetIP](#) - Changes the index of Listen IP
- [UnbanIP](#) - Removes denied IP mask from the Autoban list for the Site
- [IPAccessRuleType](#) enum

IP Access Rule Settings

In addition to the [ICIAutobanIPAccessRule Interface](#), [ICIBannedIP Interface Interface](#), and [ICIManualIPAccessRule Interface](#), there are also other settings in the [ICIServer Interface](#), [ICISite Interface](#), [ICIClientSettings Interface](#), and [Enum Reference](#).

In the ICIServer Interface:

- Sub **AddIPAccessRule** (bstrMask As String, bAllow As Boolean, nRulePos As Long)
Adds allowed or denied IP mask for administration.

```
HRESULT AddIPAccessRule(
    [in] BSTR bstrMask,
    [in] VARIANT_BOOL bAllow,
    [in] long nRulePos) //position in list (0 is first)
    [in] BSTR bstrComment (added in v7.5)
```

Example (VBScript):

```
oServer.AddIPAccessRule("1.2.3.4", false, 2, reason)
```

EFT v6.3 and later

- Function **GetIPAccessRules()** As Variant
Retrieve an array of administration IP address access rules.

```
HRESULT GetIPAccessRules([out, retval] VARIANT *aMasks);
```

Example (VBScript):

```
rules = oServer.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
```

```
WScript.echo "AutoBan"
End if
If key.type = 1 Then
WScript.echo "Manual " + key.address
End if
Next
```

EFT v6.3 and later

- Property **IPAccessAllowedDefault** As Boolean

Retrieves or specifies the default IP Access restriction list.

```
HRESULT IPAccessAllowedDefault([out, retval] VARIANT_BOOL
*pVal);
HRESULT IPAccessAllowedDefault([in] VARIANT_BOOL newVal);
```

True = All IP addresses are allowed except those specified

False = All IP addresses are denied except those specified

All versions

- Sub **MoveIPAccessRule**(nRulePosFrom As Long, nRulePosTo As Long)

Changes allowed administration IP access rule position.

```
HRESULT MoveIPAccessRule(
[in] long nRulePosFrom, //initial position
[in] long nRulePosTo); //new position
```

Example (VBScript):

```
oServer.MoveIPAccessRule(1,2)
```

EFT v6.3 and later

- Sub **RemoveIPAccessRule** (nRulePos As Long)

Removes an allowed or denied IP mask for administration.

```
HRESULT RemoveIPAccessRule([in] long nRulePos); //position
```

Example (VBScript):

```
oServer.RemoveIPAccessRule(0)
```

EFT v6.3 and later

- Sub **UnbanIP**(bstrMask As String)

Removes denied IP mask from Autoban list for administration.

(Does not remove manually banned IPs. Use RemoveIPAccessRule instead.)

```
HRESULT UnbanIP([in] BSTR bstrMask); //mask to remove
```

Example (VBScript):

```
oServer.UnbanIP("1.2.3.4")
```

EFT v6.3 and later

In the ICISite Interface:

- Sub **AddIPAccessRule**(bstrMask As String, bAllow As Boolean, nRulePos As Long)

Adds allowed or denied IP mask for the Site.

```
HRESULT AddIPAccessRule(  
[in] BSTR bstrMask,  
[in] VARIANT_BOOL bAllow,  
[in] long nRulePos);
```

Example (VBScript):

```
oSite.AddIPAccessRule("1.2.3.4", false, 2)
```

All versions

- Property **AutoBanIPsPermanently** As Boolean

Retrieves or specifies whether to automatically ban IP addresses permanently.

```
HRESULT AutoBanIPsPermanently([out, retval] VARIANT_BOOL  
*pVal);  
HRESULT AutoBanIPsPermanently([in] VARIANT_BOOL newVal);
```

True=Permanently; False=Temporarily

All versions

- Property **BanIPForInvalidLoginAttempts** As Boolean

When banning an IP address for invalid login attempts, specify whether to count incorrect username and correct username + incorrect password or only incorrect username attempts.

```
HRESULT BanIPForInvalidLoginAttempts([out, retval] VARIANT_
BOOL* pVal);
HRESULT BanIPForInvalidLoginAttempts([in] VARIANT_BOOL
newVal);
```

TRUE = Count both 'incorrect username' and 'correct username + incorrect password'

FALSE = Count only incorrect username attempts

EFT v6.4 and later

- Property **BanIPForInvalidLoginAttemptsWithExistingUsername** As Boolean

Retrieves or specifies whether to ban an IP address after excessive invalid login attempts with existing username and wrong password.

```
HRESULT BanIPForInvalidLoginAttemptsWithExistingUsername
([out, retval] VARIANT_BOOL* pVal);
HRESULT BanIPForInvalidLoginAttemptsWithExistingUsername
([in] VARIANT_BOOL newVal);
```

EFT v6.4 and later

- Property **BanIPOnDOS** As Boolean

Retrieves or specifies whether to ban an IP address after excessive invalid commands.

```
HRESULT BanIPOnDOS([out, retval] VARIANT_BOOL *pVal);
HRESULT BanIPOnDOS([in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

All versions

- Function **GetIPAccessRules()** As Variant

Retrieves an array of allowed IP masks.

```
HRESULT GetIPAccessRules([out, retval] VARIANT *aMasks);
```

Example (VBScript):

```
rules = oSite.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
WScript.echo "AutoBan"
End if
```

```
If key.type = 1 Then
WScript.echo "Manual " + key.address
End if
Next
```

EFT v6.3 and later

- Property **InvalidLoginAttemptsCountToBanIP** As Long

Retrieves or specifies the number of login attempts after which to ban the IP address.

```
HRESULT InvalidLoginAttemptsCountToBanIP([out, retval] long*
pVal);
HRESULT InvalidLoginAttemptsCountToBanIP([in] long newVal);
```

EFT v6.4 and later

- Property **IPAccessAllowedDefault** As Boolean

Retrieves or specifies whether IP access is allowed by default.

```
HRESULT IPAccessAllowedDefault([out, retval] VARIANT_BOOL
*pVal);
HRESULT IPAccessAllowedDefault([in] VARIANT_BOOL newVal);
```

True = Allowed; False = Denied

All versions

- Property **MaxInvalidLoginAttemptsPeriodToBanIP** As Long

Retrieves or specifies the period during which to count invalid login attempts.

```
HRESULT MaxInvalidLoginAttemptsPeriodToBanIP([out, retval]
long* pVal);
HRESULT MaxInvalidLoginAttemptsPeriodToBanIP([in] long
newVal);
```

EFT v6.4 and later

- Sub **MoveIPAccessRule**(bstrMask As String, bAllow As Boolean, nRulePos As Long)

Changes allowed or denied IP mask position for the Site.

```
HRESULT MoveIPAccessRule([in] long nRulePosFrom, [in] long
nRulePosTo);
```


Example (VBScript):

```
oSite.MoveIPAccessRule(1,2)
```

EFT v6.3 and later

- Sub **RemoveIPAccessRule**(nRulePos As Long)
Removes allowed or denied IP mask for the Site.

```
HRESULT RemoveIPAccessRule([in] long nRulePos);
```

Example (VBScript):

```
oSite.RemoveIPAccessRule(0)
```

All versions

- Sub **UnbanIP**(bstrMask As String) (read only)
Removes denied IP mask from the Autoban list for the Site.
(Does not remove manually banned IPs. UseRemoveIPAccessRule instead.)

```
HRESULT UnbanIP([in] BSTR bstrMask //mask to remove);
```

Example (VBScript):

```
oSite.UnbanIP("1.2.3.4")
```

In the ICIClientSettings Interface:

- Sub **AddIPAccessRule**(bstrMask As String, bAllow As Boolean, [nRulePos As Long])
Adds allowed (true) or denied (false) IP mask for the client.

```
HRESULT AddIPAccessRule([in] BSTR bstrMask,  
[in] VARIANT_BOOL bAllow,  
[in] long nRulePos);
```

VARIANT_BOOL equals TRUE(-1) when it is allowed and FALSE(0) when it is denied

Example (C#):

```
client.AddIPAccessRule("1.1.1.2", true, 0);
```

EFT v5.1.1 and later

- Function **GetIPAccessRules**() As Variant
Retrieves an array of client/template IP Access rules.

```
HRESULT GetIPAccessRules([out, retval] VARIANT *aMasks);
```

Example (VBScript):

The following script would retrieve a list of all IP addresses that have been banned automatically for a user or Settings template.

```
rules = oClientSettings.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
WScript.echo "AutoBan"
End if
If key.type = 1 Then
WScript.echo "Manual " + key.address
End if
Next
```

EFT v6.3 and later

- Function **GetLimitAccessByIP**([pInherited]) As Boolean

Determines if the IP Access Restriction List is enabled on the Settings Template or user account.

```
HRESULT GetLimitAccessByIP(
[out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
If client.GetLimitAccessByIP Then
MsgBox "client's Limit Access by IP is on"
End If
```

EFT v6.1 and later

- Property **IPAccessAllowedDefault** As Boolean

Specify or retrieve whether user or Settings Template IP access settings allow or deny access by default (if no rule match).

```
HRESULT IPAccessAllowedDefault([out, retval] VARIANT_BOOL
*pVal);
HRESULT IPAccessAllowedDefault([in] VARIANT_BOOL newVal);
```

VARIANT_BOOL equals True(-1) = Allow; False(0) = Deny

EFT v5.1.1 and later

- Sub **MoveIPAccessRule**(nRulePosFrom As Long nRulePosTo As Long)

Change allowed IP access rule position.

```
HRESULT MoveIPAccessRule([in] long nRulePosFrom,[in] long
nRulePosTo);
```

VARIANT_BOOL True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
oClientSettings.MoveIPAccessRule(1,2)
```

EFT v6.3 and later

- Sub **RemoveIPAccessRule**(nRulePos As Long)

Remove the IP access rule.

```
HRESULT RemoveIPAccessRule([in] long nRulePos);
```

Example (VBScript):

```
oClientSettings.RemoveIPAccessRule(0)
```

EFT v5.1.1 and later

- Sub **SetLimitAccessByIP**(val As SFTPAdvBool)

Specify whether Limit Access by IP is on for the Settings Template or user account.

```
HRESULT SetLimitAccessByIP([in] SFTPAdvBool val);
```

SFTPAdvBool

- abFalse = 0
- abInherited = -2
- abTrue = 1

Example (VBScript):

```
client.SetLimitAccessByIP abInherit
```

Example (VBScript); assigns IP address 1.1.1.1 to a client as an allowed IP address:

```
client.AddIPAccessRule "1.1.1.1", True 'Assign 'Allowed 'IP
```

EFT v6.1 and later

In the Enum Reference:

IPAccessRuleType (v6.3 and later)

[Server.GetIPAccessRules](#), [Site.GetIPAccessRules](#), and [Client.GetIPAccessRules](#) return the array of `ICIIPAccessRule` (which can be either Manual or Autoban);

`ICIAutobanIPAccessRule.BannedIPs` is the array of `ICIBannedIP`. All are read-only properties; to modify IP Access, use the corresponding methods of Server, Site, or Client (Move, Remove, Unban, etc).

- `AutobanIPRule = 0`
- `ManualIPRule = 1`

Example (VBScript):

```
rules = oServer.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
WScript.echo "AutoBan"
End if
If key.type = 1 Then
WScript.echo "Manual " + key.address
End if
Next
```

ICIAutobanIPAccessRule Interface

This interface provides the Autoban IP access rule `BannedIPs` property.

Example (VBScript):

```
rules = oSite.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
WScript.echo "AutoBan"
autoban = key.BannedIPs
For Each key2 In autoban
WScript.echo "Address " & CStr(key2.address)
WScript.echo "Banned " & CStr(key2.banned)
WScript.echo "Permanently " & CStr(key2.permanently)
```

```
WScript.echo "Expires " & CStr(key2.expires)
WScript.echo "Reason " & CStr(key2.reason)
Next
End if
Next
```

Properties

- Property **BannedIPs** As Variant

read-only

```
HRESULT BannedIPs([out, retval] VARIANT* pvarBannedIPs);
```

EFT v6.3.0 and later

- Property **type** As [IPAccessRuleType](#)

read-only

```
HRESULT type([out, retval] IPAccessRuleType* pVal);
```

EFT v6.3.0 and later

- [ICIBannedIP](#) - Represents an entry in an IP address autoban list
 - [ICIClientSettings](#)
 - [AddIPAccessRule](#) - Adds allowed (true) or denied (false) IP mask for the client
 - [GetIPAccessRules](#) - Retrieves an array of client/template IP Access rules
 - [GetLimitAccessByIP](#) - Determines if the IP Access Restriction List is enabled on the Settings Template or user account
 - [IPAccessAllowedDefault](#) - Set or retrieve whether user or Settings Template IP access settings allow or deny access by default (if no rule match)
 - [MoveIPAccessRule](#) - Change allowed IP access rule position
- [ICIManualIPAccessRule](#) - Represents an IP address access rule that was added manually
- [ICIServer](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for administration
 - [GetAdminIPs](#) - Retrieve an array of IP addresses for the computer on which the Server is running
 - [GetIPAccessRules](#) - Retrieve an array of administration IP address access rules
 - [GetLocalIP](#) - Given an index, retrieves a Local IP address

- [IPAccessAllowedDefault](#) - Retrieve or specify the default IP Access restriction list
- [ListenIP](#) - Retrieve or specify the server's IP address
- [ListenIPs](#) - Retrieve or specify the IP addresses on which the server listens
- [MoveIPAccessRule](#) - Change allowed administration IP access rule position.
- [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for administration
- [UnbanIP](#) - Remove denied IP mask from Autoban list for administration
- [ICISite](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for the Site
 - [AutoBanIPsPermanently](#) - Retrieve or specify whether to automatically ban IP addresses permanently
 - [BanIPForInvalidLoginAttempts](#) - Retrieve or specify whether to ban an IP address after excessive invalid login attempts with existing username and wrong password.
 - [BanIPForInvalidLoginAttemptsWithExistingUsername](#) - When banning an IP address for invalid login attempts, specify whether to count incorrect username and correct username + incorrect password or only incorrect username attempts.
 - [BanIPOnDOS](#) - Retrieve or specify whether to ban an IP address after excessive invalid commands
 - [GetIP](#) - Retrieves the index of the Site Listen IP
 - [GetIPAccessRules](#) - Retrieves an array of allowed IP masks.
 - [IPAccessAllowedDefault](#) - Retrieve or specify whether IP access is allowed by default
 - [ListenIPs](#) - Retrieve or specify the IP addresses on which the Site listens
 - [MoveIPAccessRule](#) - Change allowed or denied IP mask position for the Site
 - [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for the Site
 - [SetIP](#) - Changes the index of Listen IP
 - [UnbanIP](#) - Removes denied IP mask from the Autoban list for the Site
- [IPAccessRuleType](#) enum

ICIBannedIP Interface

The ICIBannedIP interface represents an entry in an IP address autoban list.

Example (VBScript):

```
rules = oSite.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
```

```

WScript.echo "AutoBan"
autoban = key.BannedIPs
For Each key2 In autoban
WScript.echo "Address " & CStr(key2.address)
WScript.echo "Banned " & CStr(key2.banned)
WScript.echo "Permanently " & CStr(key2.permanently)
WScript.echo "Expires " & CStr(key2.expires)
WScript.echo "Reason " & CStr(key2.reason)
Next
End if
Next

```

Members

- Property **Address** As String

(read-only)

Retrieves a string of banned IP addresses

```
HRESULT Address([out, retval] BSTR* pbzAddress);
```

EFT v6.3.0 and later

- Property **Banned** As Date

(read-only)

Retrieves the date an IP address was added to the ban list.

```
HRESULT Banned([out, retval] DATE* pdBanned);
```

EFT v6.3.0 and later

- Property **Expires** As Date

(read-only)

Retrieves the date the ban expires.

```
HRESULT Expires([out, retval] DATE* pdExpires);
```

EFT v6.3.0 and later

- Property **Permanently** As Boolean

(read-only)

Retrieves whether the ban is permanent (True or False).

```
HRESULT Permanently([out, retval] VARIANT_BOOL*
pbPermanently);
```

EFT v6.3.0 and later

- Property **Reason As String**
(read-only)

Retrieves the reason the IP address was banned.

```
HRESULT Reason([out, retval] BSTR* pbzReason);
```

EFT v6.3.0 and later

- [ICIAutobanIPAccessRule](#) - Provides the Autoban IP access rule BannedIPs property
- [ICIClientSettings](#)
 - [AddIPAccessRule](#) - Adds allowed (true) or denied (false) IP mask for the client
 - [GetIPAccessRules](#) - Retrieves an array of client/template IP Access rules
 - [GetLimitAccessByIP](#) - Determines if the IP Access Restriction List is enabled on the Settings Template or user account
 - [IPAccessAllowedDefault](#) - Set or retrieve whether user or Settings Template IP access settings allow or deny access by default (if no rule match)
 - [MoveIPAccessRule](#) - Change allowed IP access rule position
- [ICIManualIPAccessRule](#) - Represents an IP address access rule that was added manually
- [ICIServer](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for administration
 - [GetAdminIPs](#) - Retrieve an array of IP addresses for the computer on which the Server is running
 - [GetIPAccessRules](#) - Retrieve an array of administration IP address access rules
 - [GetLocalIP](#) - Given an index, retrieves a Local IP address
 - [IPAccessAllowedDefault](#) - Retrieve or specify the default IP Access restriction list
 - [ListenIP](#) - Retrieve or specify the server's IP address
 - [ListenIPs](#) - Retrieve or specify the IP addresses on which the server listens
 - [MoveIPAccessRule](#) - Change allowed administration IP access rule position.
 - [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for administration
 - [UnbanIP](#) - Remove denied IP mask from Autoban list for administration

- [ICISite](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for the Site
 - [AutoBanIPsPermanently](#) - Retrieve or specify whether to automatically ban IP addresses permanently
 - [BanIPForInvalidLoginAttempts](#) - Retrieve or specify whether to ban an IP address after excessive invalid login attempts with existing username and wrong password.
 - [BanIPForInvalidLoginAttemptsWithExistingUsername](#) - When banning an IP address for invalid login attempts, specify whether to count incorrect username and correct username + incorrect password or only incorrect username attempts.
 - [BanIPOnDOS](#) - Retrieve or specify whether to ban an IP address after excessive invalid commands
 - [GetIP](#) - Retrieves the index of the Site Listen IP
 - [GetIPAccessRules](#) - Retrieves an array of allowed IP masks.
 - [IPAccessAllowedDefault](#) - Retrieve or specify whether IP access is allowed by default
 - [ListenIPs](#) - Retrieve or specify the IP addresses on which the Site listens
 - [MoveIPAccessRule](#) - Change allowed or denied IP mask position for the Site
 - [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for the Site
 - [SetIP](#) - Changes the index of Listen IP
 - [UnbanIP](#) - Removes denied IP mask from the Autoban list for the Site
- [IPAccessRuleType](#) enum

ICIManualIPAccessRule Interface

The ICIManualIPAccessRule interface represents an IP address access rule that was added manually.

Example (VBScript):

```
rules = oServer.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then WScript.echo "AutoBan"
End if
If key.type = 1 Then
WScript.echo "Manual " + key.address
WScript.echo "Added " + key.added
WScript.echo "Allow " + key.allow
End if
Next
```

Members

- Property **Address** As String (read-only)

Retrieves the IP address.

```
HRESULT Address([out, retval] BSTR* pbzAddress);
```

EFT v6.3.0 and later

- Property **Added** As Date (read-only)

Retrieves the date when the IP access rule was added.

```
HRESULT Added([out, retval] DATE* pdAdded);
```

EFT v6.3.0 and later

- Property **Allow** As Boolean (read-only)

Retrieves whether the IP access rule is a permission or ban.

```
HRESULT Allow([out, retval] VARIANT_BOOL* pbAllow);
```

EFT v6.3.0 and later

- Property **Reason** As String (read-only)

Retrieves the reason for the IP access rule.

```
HRESULT Reason([out,
                retval] BSTR* pbzReason);
```

EFT v8 and later

- Property **type** As [IPAccessRuleType](#) (read-only)

Retrieves the type of the rule.

```
HRESULT type([out, retval] IPAccessRuleType* pVal);
```

EFT v6.3.0 and later

- [ICIAutobanIPAccessRule](#) - Provides the Autoban IP access rule BannedIPs property
- [ICIBannedIP](#) - Represents an entry in an IP address autoban list
- [ICIClientSettings](#)

- [AddIPAccessRule](#) - Adds allowed (true) or denied (false) IP mask for the client
- [GetIPAccessRules](#) - Retrieves an array of client/template IP Access rules
- [GetLimitAccessByIP](#) - Determines if the IP Access Restriction List is enabled on the Settings Template or user account
- [IPAccessAllowedDefault](#) - Set or retrieve whether user or Settings Template IP access settings allow or deny access by default (if no rule match)
- [MoveIPAccessRule](#) - Change allowed IP access rule position
- [ICIServer](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for administration
 - [GetAdminIPs](#) - Retrieve an array of IP addresses for the computer on which the Server is running
 - [GetIPAccessRules](#) - Retrieve an array of administration IP address access rules
 - [GetLocalIP](#) - Given an index, retrieves a Local IP address
 - [IPAccessAllowedDefault](#) - Retrieve or specify the default IP Access restriction list
 - [ListenIP](#) - Retrieve or specify the server's IP address
 - [ListenIPs](#) - Retrieve or specify the IP addresses on which the server listens
 - [MoveIPAccessRule](#) - Change allowed administration IP access rule position.
 - [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for administration
 - [UnbanIP](#) - Remove denied IP mask from Autoban list for administration
- [ICISite](#)
 - [AddIPAccessRule](#) - Adds allowed or denied IP mask for the Site
 - [AutoBanIPsPermanently](#) - Retrieve or specify whether to automatically ban IP addresses permanently
 - [BanIPForInvalidLoginAttempts](#) - Retrieve or specify whether to ban an IP address after excessive invalid login attempts with existing username and wrong password.
 - [BanIPForInvalidLoginAttemptsWithExistingUsername](#) - When banning an IP address for invalid login attempts, specify whether to count incorrect username and correct username + incorrect password or only incorrect username attempts.
 - [BanIPOnDOS](#) - Retrieve or specify whether to ban an IP address after excessive invalid commands
 - [GetIP](#) - Retrieves the index of the Site Listen IP
 - [GetIPAccessRules](#) - Retrieves an array of allowed IP masks.

- [IPAccessAllowedDefault](#) - Retrieve or specify whether IP access is allowed by default
- [ListenIPs](#) - Retrieve or specify the IP addresses on which the Site listens
- [MoveIPAccessRule](#) - Change allowed or denied IP mask position for the Site
- [RemoveIPAccessRule](#) - Remove allowed or denied IP mask for the Site
- [SetIP](#) - Changes the index of Listen IP
- [UnbanIP](#) - Removes denied IP mask from the Autoban list for the Site
- [IPAccessRuleType](#) enum

PGP-Related Objects

This chapter contains PGP-related COM objects.

In addition to the [ICIPgpActionParams Interface](#) and the [ICIPGPKeyInfo Interface](#), PGP-related settings can also be found in the [ICIServer Interface](#), [ICISite Interface](#), and [Enum Reference](#).

In the ICIServer Interface:

- Property **AvailablePGPKeys** As Variant read-only

Retrieve a list of EFT's PGP Keys. Its value is a variant containing the array of [ICIPGPKeyInfo](#)

```
HRESULT AvailablePGPKeys([out, retval]VARIANT* pVal);
```

Example (VBScript):

```
For Each key In Server.AvailablePGPKeys
  MsgBox key.Name & " key is available"
Next
```

EFT v6.1 and later

- Function **CreatePGPKey**(bstrFullName As String, bstrEmailAddress As String, dtExpirationDate As Date, nKeyType As PGPKeyType, nKeyCipher As PGPKeyCipher, ulKeySize As DWord, bstrPassphrase As String) As String

Create a PGPKey on EFT:

```
HRESULT CreatePGPKey(
  [in] BSTR bstrFullName,
  [in] BSTR bstrEmailAddress,
  [in] DATE dtExpirationDate,
```

```
[in] PGPPKeyType nKeyType,
[in] PGPPKeyCipher nKeyCipher,
[in] ULONG ulKeySize,
[in] BSTR bstrPassphrase,
[out,retval] BSTR* pVal);
```

PGPPKeyType

- PGPPKeyType_DSS = 0
- PGPPKeyType_RSA = 1
- PGPPKeyType_RSALegacy = 2

PGPPKeyCipher

- PGPPKeyCipher_IDEA = 0
- PGPPKeyCipher_3DES = 1
- PGPPKeyCipher_CAST5 = 2
- PGPPKeyCipher_AES128 = 3
- PGPPKeyCipher_AES192 = 4
- PGPPKeyCipher_AES256 = 5
- PGPPKeyCipher_TWOFISH = 6

Example (VBScript):

```
strKeyId = oServer.CreatePGPPKey("Ivan Ivanuch",
"ipetrov@mail.com",
expDate, PGPPKeyType_RSA, PGPPKeyCipher_IDEA, 1024,
"password")
MsgBox "New key id is " & strKeyId
```

EFT v6.3 and later

- Sub **ExportPGPPKey**(bstrKeyId As String, vbIncludePrivate As Boolean, bstrFileName As String)

Export an Open PGP Key:

```
HRESULT ExportPGPPKey(
[in] BSTR bstrKeyId,
[in] VARIANT_BOOL vbIncludePrivate,
[in] BSTR bstrFileName);
```

Example (VBScript):

```
oServer.ExportPGPKey "0x12345678", False,
"C:\PGP\pgpkey.asc"
```

EFT v6.3 and later

- Sub **GetPGPKeyringSettings**(pPublicPath As String, pPrivatePath As String)

For an example using this method, refer to the PGP script in [Script Examples](#).

Retrieve the OpenPGP Keyring settings.

```
HRESULT GetPGPKeyringSettings (
[out] VARIANT* pPublicPath,
[out] VARIANT* pPrivatePath);
```

Example (VBScript):

```
oServer.GetPGPKeyringSettings strPublicKeyring,
strPrivateKeyring
```

Example (C#):

```
m_server.ImportSSHKey(@"C:\SSHKey.pub");
object publicPath = new object();
object privatePath = new object();
m_server.GetPGPKeyringSettings(out publicPath, out
privatePath);
m_server.SetPGPKeyringSettings
(@"C:\filecryptdata\pubring.pgp",
@"C:\filecryptdata\sekring.pgp");
```

EFT v6.3 and later

- Sub **ImportPGPKey**(bstrKey As String, bstrKeyFile As String)

```
HRESULT ImportPGPKey([in] BSTR bstrKeyPath,[out, retval]
BSTR* pNewKeyID);
```

Example (VBScript):

```
KeyId = oServer.ImportPGPKey("C:\PGP\pgpkey.asc")
```

EFT v6.3 and later

- Sub **RemovePGPKey**(bstrKeyId As String)

Removes OpenPGP Key from EFT.

```
HRESULT RemovePGPKey([in] BSTR bstrKeyId);
```

Example (VBScript):

```
oServer.RemovePGPKey "0x01234567"
```

EFT v6.3 and later

- Sub **SetPGPKeyringSettings**(newPublicKeyring As String, newPrivateKeyring As String)

For an example using this method, refer to the PGP script in [Script Examples](#).

Changes OpenPGP Keyring settings.

```
HRESULT SetPGPKeyringSettings([in] BSTR newPublicKeyring,
[in] BSTR newPrivateKeyring);
```

Retrieve OpenPGP Keyring settings with GetPGPKeyringSettings.

Example (VBScript):

```
oServer.SetPGPKeyringSettings "C:\PGP\pubrign.pgp",
"C:\PGP\sekring.pgp"
```

Example (C#):

```
m_server.ImportSSHKey(@"C:\SSHKey.pub");
object publicPath = new object();
object privatePath = new object();
m_server.GetPGPKeyringSettings(out publicPath, out
privatePath);
m_server.SetPGPKeyringSettings
(@"C:\filecryptdata\pubring.pgp",
@"C:\filecryptdata\sekring.pgp");
```

EFT v6.3 and later

In the ICISite Interface:

- Property **DefaultPGPKeyID** As String (read-only)

See also **SetDefaultPGPKey**

Retrieves the default PGP Key ID or empty string if Site has not default key.

```
HRESULT DefaultPGPKeyID([out, retval] BSTR* pVal);
```

EFT v6.4 and later

- Property **DefaultPGPKeyPassphrase** As String (read-only)

See also **SetDefaultPGPKey**

Retrieves the default PGP Key passphrase.

```
HRESULT DefaultPGPKeyPassphrase([out, retval] BSTR* pVal);
```

EFT. v6.4 and later

- Property **EnablePGPLogging** As Boolean

Retrieves or specifies whether OpenPGP logging is enabled.

```
HRESULT EnablePGPLogging([out, retval] VARIANT_BOOL* pVal);
HRESULT EnablePGPLogging([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; FALSE=Disabled

Example (VBScript):

Retrieving:

```
MsgBox "PGP logging: " & CStr(oSite.EnablePGPLogging)
```

Specifying:

```
oSite.EnablePGPLogging = False
```

EFT v6.3 and later

- Property **PGPLogFilePath** As String

Retrieves or specifies the log file path.

```
HRESULT PGPLogFilePath([out, retval] BSTR* pVal);
HRESULT PGPLogFilePath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "PGP log file path: " & oSite.PGPLogFilePath
```

Specifying:

```
oSite.PGPLogFilePath = "C:\PGP\logfile.txt"
```


EFT v6.3 and later

- Property **PGPLogLevel** As PGPLogLevel

Retrieves or specifies the OpenPGP logging level.

```
HRESULT PGPLogLevel([out, retval] PGPLogLevel* pVal);
HRESULT PGPLogLevel([in] PGPLogLevel newVal);
```

PGPLogLevel

- PGPLogLevelStandard = 0
- PGPLogLevelVerbose = 1

Example (VBScript):

Retrieving:

```
if PGPLogLevelStandard = oSite.PGPLogLevel
MsgBox "PGP log level is standard."
End If
```

Specifying:

```
oSite.PGPLogLevel = PGPLogLevelVerbose
```

EFT v6.3 and later

- Sub **SetDefaultPGPKey**(bzKeyID As String, bzPassPhrase As String)

See also **DefaultPGPKeyID** and **DefaultPGPKeyPassphrase**.

Specifies the default PGP Key and passphrase. Specify an empty string for the key to assign "no key."

```
HRESULT SetDefaultPGPKey([in] BSTR bzKeyID, [in] BSTR
bzPassphrase);
```

EFT v6.4 and later

- Property **UseDynamicPGPLogFile** As Boolean

Retrieves or specifies whether to use dynamic log file.

```
HRESULT UseDynamicPGPLogFile([out, retval] VARIANT_BOOL*
pVal);
HRESULT UseDynamicPGPLogFile([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "PGP uses dynamic log file:" & CStr
(oSite.UseDynamicPGPLogFile)
```

Specifying:

```
oSite.UseDynamicPGPLogFile = true
```

EFT v6.3 and later

In the Enum Reference:**EventActionType (v6.0 and later)**

Used in the ICIEventAction Interface [type](#) property.

- PGPACTION = 32

PGPCompressionLevel (v6.1 and later)

Used to specify the compression level for a PGP encryption operation in the [ICIPgpActionParams InterfaceCompressionLevel](#) property.

- NoCompression = 0
- CompressionLevel1 = 1
- CompressionLevel2 = 2
- CompressionLevel3 = 3
- CompressionLevel4 = 4
- CompressionLevel5 = 5
- CompressionLevel6 = 6
- CompressionLevel7 = 7
- CompressionLevel8 = 8
- CompressionLevel9 = 9

PGPKeyCipher (v6.3 and later)

Used in the ICIServer Interface [CreatePGPKey](#) method to specify the PGP key cipher.

- PGPKeyCipher_IDEA = 0
- PGPKeyCipher_3DES = 1
- PGPKeyCipher_CAST5 = 2

- PGPCipher_AES128 = 3
- PGPCipher_AES192 = 4
- PGPCipher_AES256 = 5
- PGPCipher_TWOFISH = 6

PGPKeyType (v6.3 and later)

Used in the ICIServer Interface [CreatePGPKey](#) method to specify the PGP key type.

- PGPKeyType_DSS = 0
- PGPKeyType_RSA = 1
- PGPKeyType_RSALegacy = 2

PGPLogLevel (v6.3 and later)

Used in the ICISite Interface [PGPLogLevel](#) property to specify the PGP module log level.

- PGPLogLevelStandard = 0
- PGPLogLevelVerbose = 1

PGPOperation (v6.0 and later)

Used to specify the PGP operation to perform on a file in the ICIPgpActionParams Interface [Operation](#) property.

- Encrypt = 0
- EncryptAndSign (added in v6.1) = 1
- Sign (added in v6.1) = 2
- SDA (added in v6.1) = 3
- Decrypt (added in v6.1) = 4
- DecryptAndVerify (added in v6.3) = 5
- VerifyOnly (added in v6.3) = 6

PGPSigningHash (v6.1 and later)

Used to specify the hash algorithm for a PGP signing operation in the ICIPgpActionParams Interface [SigningHash](#) property.

- MD5Hash = 0
- SHA1Hash = 1
- RIPEMD160Hash = 2
- SHA256Hash = 3

- SHA384Hash = 4
- SHA512Hash = 5

ServerModule (v6.1 and later)

Used in the ICIServer Interface [ModuleRegistrationState](#) method to retrieve an EFT module state. Used with the RegistrationState enumerator.

- PGPMModule = 5

ICIPgpActionParams Interface

The ICIPgpActionParams interface allows you to make changes to the OpenPGP Event Action settings. Access the ICIPgpActionParams interface with the [ICIServerEventActionParams](#) property (when its Type property returns PGPACTION).

```
interface ICIPgpActionParams : IDispatch
```

Example:

```
If action.Type = 32 then 'PGPACTION
Set pgpParams = action.Params
EndIf
```

Properties

- Property **ASCIIArmoredOutput** As Boolean

Retrieves or specifies whether ASCII Armored Output is specified.

```
HRESULT ASCIIArmoredOutput([out, retval] VARIANT_BOOL*
pVal);
HRESULT ASCIIArmoredOutput([in] VARIANT_BOOL newVal);
```

EFT v6.1 and later

- Property **CompressionLevel** As [PGPCompressionLevel](#)

Retrieves or specifies the PGP compression level.

```
HRESULT CompressionLevel([out, retval] PGPCompressionLevel*
pVal);
HRESULT CompressionLevel([in] PGPCompressionLevel newVal);
```

EFT v6.1 and later

- Property **FilePath** As String

Retrieves or specifies the file path.

```
HRESULT FilePath([out, retval] BSTR *pVal);
HRESULT FilePath([in] BSTR newVal);
```

EFT v5.2 and later

- Property **KeyIDs** As Variant

Retrieves or specifies an array of key IDs.

```
HRESULT KeyIDs([out, retval] VARIANT *pVal);
HRESULT KeyIDs([in] VARIANT newVal);
```

EFT v5.2 and later

- Property **Operation** As [PGPOperation](#)

Retrieves or specifies the PGP operation.

```
HRESULT Operation([out, retval] PGPOperation *pVal);
HRESULT Operation([in] PGPOperation newVal);
```

EFT v5.2 and later

- Property **PassPhrase** As String

Retrieves or specifies the passphrase.

```
HRESULT Passphrase([out, retval] BSTR *pVal);
HRESULT Passphrase([in] BSTR newVal);
```

EFT v5.2 and later

- Property **SDAPassword** As String

Retrieves or specifies the SDA password.

```
HRESULT SDAPassword([out, retval] BSTR *pVal);
HRESULT SDAPassword([in] BSTR newVal);
```

EFT v6.1 and later

- Property **SignatureInSeparateFile** As Boolean

Retrieves or specifies whether a signature in a separate file is specified.

```
HRESULT SignatureInSeparateFile([out, retval] VARIANT_BOOL*
pVal);
HRESULT SignatureInSeparateFile([in] VARIANT_BOOL newVal);
```

EFT v6.1 and later

- Property **SigningHash** As [PGPSigningHash](#)

Retrieves or specifies the PGP signing hash.

```
HRESULT SigningHash([out, retval] PGPSigningHash* pVal);
HRESULT SigningHash([in] PGPSigningHash newVal);
```

EFT v6.1 and later

- Property **SignKeyID** As String

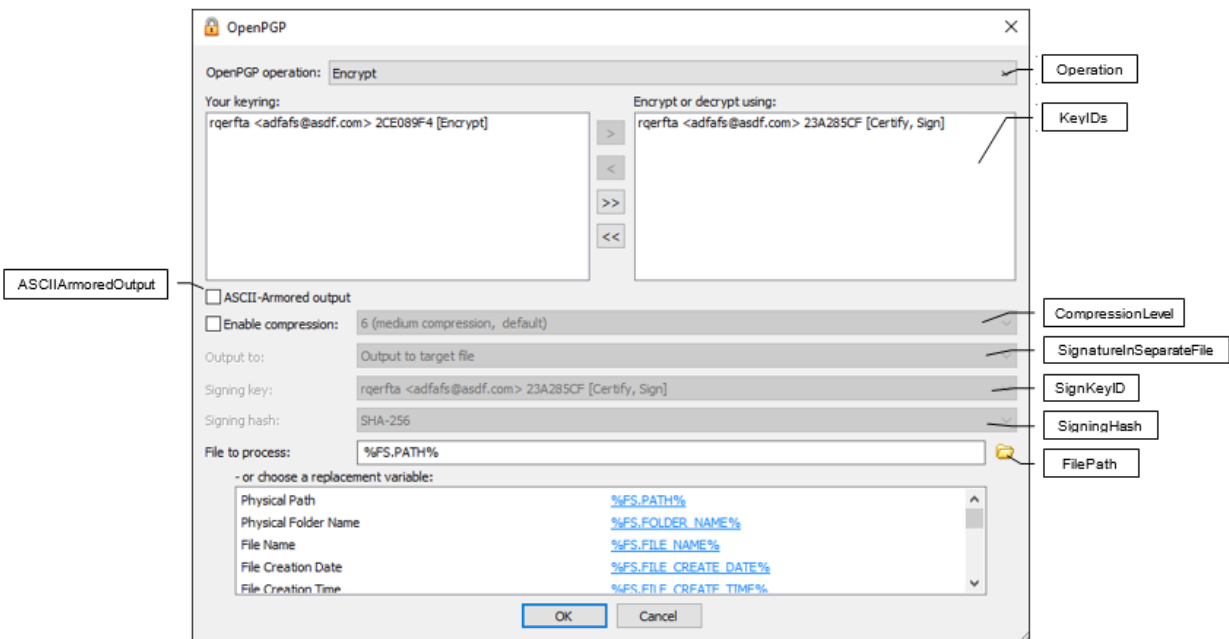
Retrieves or specifies the signing key ID.

```
HRESULT SignKeyID([out, retval] BSTR *pVal);
HRESULT SignKeyID([in] BSTR newVal);
```

EFT v5.2 and later

Dialog Box Equivalents

The dialog box below is labeled with the parameters that are available in the interface.



ICIPGPKeyInfo Interface

Use the ICIPGPKeyInfo interface to retrieve information about EFT's PGP keys. You cannot create or configure PGP keys with this interface.

```
interface ICIPGPKeyInfo : IDispatch
```

(To make changes to the OpenPGP Event Action settings, refer to [ICIPgpActionParams - OpenPGP Action Interface](#). See also the ICIServer::[AvailablePGPKeys](#) property).

Properties

- (read-only) Property **CreationDate** As Date

Retrieves the date the key was created.

```
HRESULT CreationDate([out, retval] DATE* pVal);
```

EFT v6.1 and later

- Property **Description** As String

(read-only) Retrieves the description of the key.

```
HRESULT Description([out, retval] BSTR* pVal);
```

EFT v6.1 and later

- Property **ExpirationDate** As Date

(read-only) Retrieves the expiration date of the key.

```
HRESULT ExpirationDate([out, retval] DATE* pVal);
```

EFT v6.1 and later

- Property **ID** As String

(read-only) Retrieves the ID of the key.

```
HRESULT ID([out, retval] BSTR* pVal);
```

EFT v6.1 and later

- Property **Name** As String

(read-only) Retrieves the name of the key.

```
HRESULT Name([out, retval] BSTR* pVal);
```

EFT v6.1 and later

- Property **NeverExpires** As Boolean

(read-only) Retrieves whether the key never expires.

```
HRESULT NeverExpires([out, retval] VARIANT_BOOL* pVal);
```

EFT v6.1 and later

- Property **Private** As Boolean

(read-only) Retrieves whether the key is a private key.

```
HRESULT Private([out, retval] VARIANT_BOOL* pVal);
```

EFT v6.1 and later

- Property **Size** As Long

(read-only) Retrieves the size of the key.

```
HRESULT Size([out, retval] long* pVal);
```

EFT v6.1 and later

- Property **UsageFlags** As Long

(read-only) Retrieves the intended use for the key:

- 0x01 This key may be used to certify other keys.
- 0x02 This key may be used to sign data.
- 0x0C This key may be used to encrypt communications and encrypt storage.
- 0x10 The private component of this key may have been split by a secret-sharing mechanism.
- 0x20 This key may be used for authentication.
- 0x80 The private component of this key may be in the possession of more than one person.

```
HRESULT UsageFlags([out, retval] long* pVal);
```

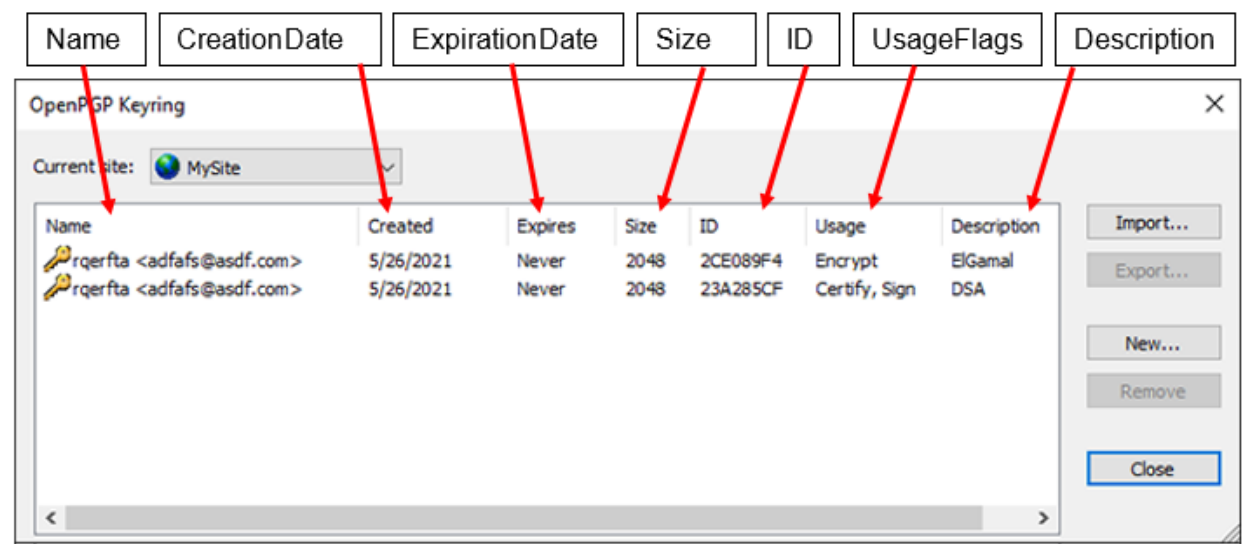
Example:


```
If key.UsageFlags = &HC Then
    ...
End If
```

EFT v8.0.6 and later

Dialog Box Equivalents

The dialog box below is labeled with the parameters that are available in the interface.



Privacy-Related Objects

This chapter contains Privacy-related COM objects.

GDPR and Privacy Settings

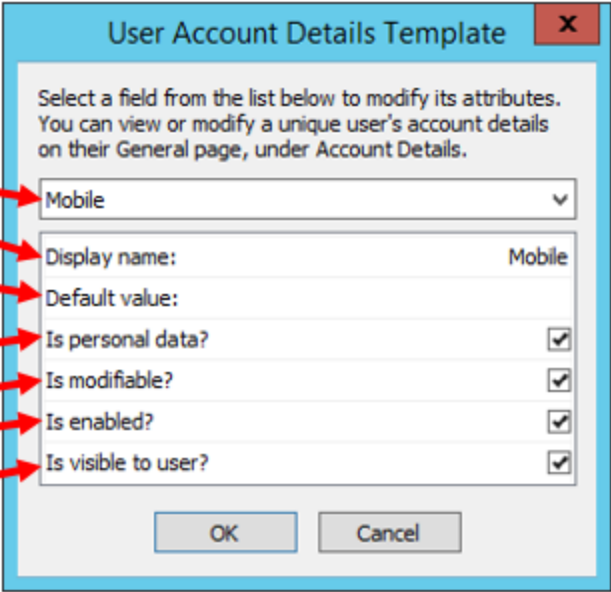
To accommodate the EU's General Data Protection Regulations (GDPR), many user interfaces and COM API objects were updated/added in EFT v8. Please refer to the links below to read details of these objects.

GDPR-related interfaces:

- [ICIClientDataField Interface](#)
- [ICIPrivacyPolicy Interface](#)
- [ICITermsOfService Interface](#)
- [ICIGDPRSettings Interface](#)

Added to the ICISite interface:

- [GetClientDataField](#) (using [ClientDataField](#) enum), [SetClientDataField](#)



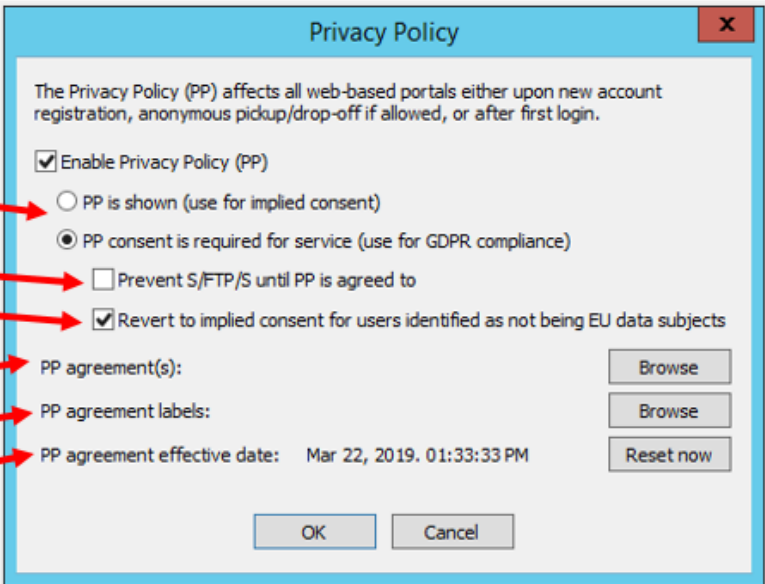
User Account Details Template

Select a field from the list below to modify its attributes. You can view or modify a unique user's account details on their General page, under Account Details.

ID → Mobile
 DisplayName → Display name: Mobile
 DefaultValue → Default value:
 IsPersonalData → Is personal data? ☒
 IsModifiable → Is modifiable? ☒
 IsEnabled → Is enabled? ☒
 IsVisible → Is visible to user? ☒

OK Cancel

- [GetPrivacyPolicy](#), [SetPrivacyPolicy](#), [ICIPrivacyPolicy](#) interface



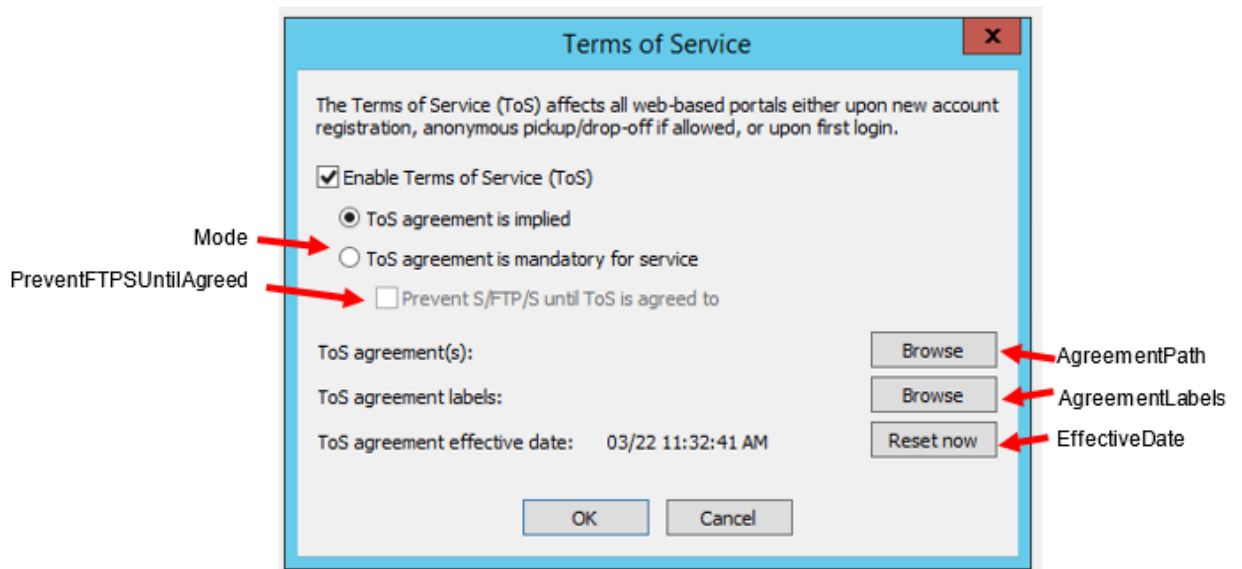
Privacy Policy

The Privacy Policy (PP) affects all web-based portals either upon new account registration, anonymous pickup/drop-off if allowed, or after first login.

☒ Enable Privacy Policy (PP)
 Mode → ☐ PP is shown (use for implied consent)
 ☒ PP consent is required for service (use for GDPR compliance)
 PreventFTPSUntilConsentGranted → ☐ Prevent S/FTP/S until PP is agreed to
 RevertToImpliedConsentForNotEUDDataSubjects → ☒ Revert to implied consent for users identified as not being EU data subjects
 AgreementPath → PP agreement(s): Browse
 AgreementLabels → PP agreement labels: Browse
 EffectiveDate → PP agreement effective date: Mar 22, 2019. 01:33:33 PM Reset now

OK Cancel

- [GetTermsOfService](#), [SetTermsOfService](#), [ICITermsOfService](#) interface



- [WorkspacesDomainsToForbidInviteeAddresses](#)
- [WorkspacesEnableDomainsToForbidInviteeAddresses](#)
- [WorkspacesOAIDomainsToForbidInviteeAddresses](#)
- [WorkspacesOAIEnableDomainsToForbidInviteeAddresses](#)
- [WorkspacesOAIEnableDomainsToLimitInviteeAddresses](#)

Added to the ICIClientSettings interface:

- [GetGDPRPrivacyPolicyConsentStatus](#), [SetGDPRPrivacyPolicyConsentStatus](#)
- [GetGDPRSubjectStatus](#), [SetGDPRSubjectStatus](#)
- [GetGDPRTermsOfServiceConsentStatus](#), [SetGDPRTermsOfServiceConsentStatus](#)
- [ResetAllFieldAttributeDefaults](#)
- [CompanyName](#)
- [Duns](#)
- [Mobile](#)
- [PartnerId](#)

Privacy-related enums:

- In the AdminPermissionsPolicy enum, added [ManagePersonalData](#)
- In the [EventProperty](#) enum, added:
 - ClientGDPREUSubjectStatus
 - ClientGDPRPPConsentStatus

- ClientGDPRConsentStatus
- ClientGDPRReasonGiven
- ClientGDPRRightExercised
- ClientGDPRRightExercisedArticleID
- In the EventType enum, added [OnGDPRRightExercised](#)
- [AgeRestrictionsType](#)
- [ClientDataField](#)
- [ConditionsForConsent](#)
- [DirectCollectionType](#)
- [DPOAssignedType](#)
- [EncryptedPDType](#)
- [GDPRPrivacyPolicyConsentStatus](#)
- [GDPRPrivacyPolicyMode](#)
- [GDPRSubjectStatus](#)
- [GDPRTermsOfServiceConsentStatus](#)
- [GDPRTermsofServiceMode](#)
- [IndirectCollectionType](#)
- [LawfulnessOfProcessingType](#)
- [LimitedAccessType](#)
- [MaterialScopeType](#)
- [NecessityType](#)
- [ProcessingPrinciplesType](#)
- [PurposeType](#)
- [RightStateType](#)
- [RightToBeNotifiedStateType](#)
- [RiskAssessmentType](#)
- [RiskMitigationType](#)
- [TerritorialScopeType](#)
- [TransferSafeguardsType](#)
- [TransparentInformationType](#)
- [UserAccountActionType](#) enum, added:
 - PersonalDataDeleteFields
 - PersonalDataResetToDefaults

- `PersonalDataSetEnabled`
- `PersonalDataSetModifiable`
- `PersonalDataSetVisible`
- `SetEUDataSubjectStatus`
- `SetPrivacyPolicyConsent`
- `SetTermsOfServiceConsent`

ICIClientDataField Interface

The `ICIClientDataField` interface is used in the User Account Details Template on the Site > Security tab, which is used to specify, for privacy purposes, whether a field is enabled, visible, or is considered personal data, in addition to the field's display name and default value.

Members

- Property **DefaultValue** As String

Retrieves or modifies a specified data field's default value,

```
HRESULT DefaultValue([out, retval] BSTR *pVal);
HRESULT DefaultValue([in] BSTR val);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.DefaultValue = "Default"
```

EFT v8 and later

- Property **DisplayName** As String

Retrieves or modifies a specified data field's display name.

```
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([in] BSTR val);
```

Example (VBScript):

```
set email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.DisplayName = "Name"
```

EFT v8 and later

- Property ID As [ClientDataField](#) read-only

Retrieves the specified data field's ID.

```
HRESULT ID([out, retval] ClientDataField *pVal);
```

Example (VBScript):

```
set email = oSite.GetClientDataField(ClientDataFieldId_Email)
wscript.echo email.ID
```

EFT v8 and later

- Property IsEnabled As Boolean

Retrieves or modifies the specified data field's "is enabled" attribute.

```
HRESULT IsEnabled([out, retval] VARIANT_BOOL *pVal);
HRESULT IsEnabled([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_Email)
email.IsEnabled = True
```

EFT v8 and later

- Property IsModifiable As Boolean

Retrieves or modifies data field's "is modifiable" attribute.

```
HRESULT IsModifiable([out, retval] VARIANT_BOOL *pVal);
HRESULT IsModifiable([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_Email)
email.IsModifiable = True
```

EFT v8 and later

- Property IsPersonalData As Boolean

Retrieves or modifies a specified data field's personal data attribute.

```
HRESULT IsPersonalData([out, retval] VARIANT_BOOL *pVal);
HRESULT IsPersonalData([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsPersonalData = True
```

EFT v8 and later

- Property **IsVisible** As Boolean

Retrieves modifies data field's "is visible" attribute.

```
HRESULT IsVisible([out, retval] VARIANT_BOOL *pVal);
HRESULT IsVisible([in] VARIANT_BOOL newVal);
```

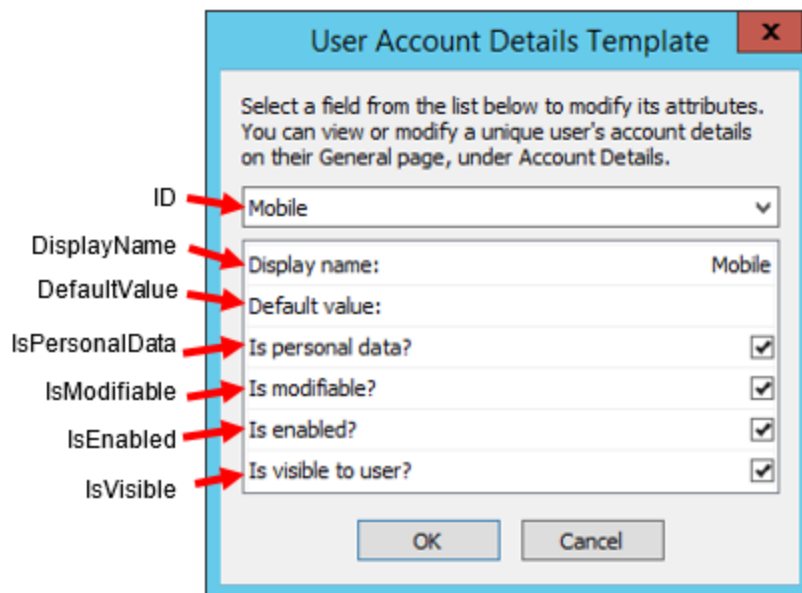
Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsEnabled = True
```

EFT v8 and later

Dialog Box Equivalents

Site > User Account Details Template



ICIGDPRSettings Interface

The ICIGDPR Settings interface is used to define GDPR Settings for a Site.

Properties

- Property **AgeRestrictions** As [AgeRestrictionsType](#)
Retrieve/modify GDPR age of restrictions mode (AgeRestrictionType)
EFT v8 and later
- Property **ConditionsForConsent** As [ConditionsForConsentType](#)
Retrieve/modify GDPR conditions for consent mode (ConditionsForConsentMode)
EFT v8 and later
- Property **DirectCollection** As [DirectCollectionType](#)
Retrieve/modify GDPR direct collection (DirectCollectionType)
EFT v8 and later
- Property **DPOAssigned** As [DPOAssignedType](#)
Retrieve/modify GDPR data protection officer assigned mode (DPOAssignedType)
EFT v8 and later
- Property **DPOAssignedEmail** As String
Retrieve/modify GDPR data protection officer email (email)
EFT v8 and later
- Property **EncryptedPD** As [EncryptedPDType](#)
Retrieve/modify GDPR encrypted PD mode (EncryptedPDType)
EFT v8 and later
- Property **EncryptionPDKey** As String As String

```
HRESULT EncryptionPDKey([out, retval] BSTR* pVal);
HRESULT EncryptionPDKey([in] BSTR newVal);
```

EFT v8.0.5 and later
- Property **IndirectCollection** As [IndirectCollectionType](#)
Retrieve/modify GDPR indirect collection mode (IndirectCollectionType)

EFT v8 and later

- Property **LawfulnessOfProcessing** As [LawfulnessOfProcessingType](#)
Retrieve/modify GDPR lawfulness or processing mode (LawfulnessOfProcessType)
EFT v8 and later
- Property **LimitedAccess** As [LimitedAccessType](#)
Retrieve/modify GDPR limited access mode (LimitedAccessType)
EFT v8 and later
- Property **MaterialScope** As [MaterialScopeType](#)
Retrieve/modify GDPR material scope (MaterialScopeType)
EFT v8 and later
- Property **Necessity** As [NecessityType](#)
Retrieve/modify GDPR necessity mode (NecessityType)
EFT v8 and later
- Property **NecessityReason** As String
Retrieve/modify GDPR necessity reason (text)
EFT v8 and later
- Property **ProcessingPrinciples** As [ProcessingPrinciplesType](#)
Retrieve/modify GDPR processing principles mode (ProcessingPrinciplesType)
EFT v8 and later
- Property **Purpose** As [PurposeType](#)
Retrieve/modify GDPR purpose mode (PurposeMode)
EFT v8 and later
- Property **PurposeReason** As String
Retrieve/modify GDPR purpose reason (text)
EFT v8 and later
- Property **RightOfAccess** As [RightStateType](#)
Retrieve/modify GDPR right to access mode (RightStateType)

EFT v8 and later

- Property **RightToBeForgotten** As [RightStateType](#)

Retrieve/modify GDPR right to be forgotten mode (RightStateType)

EFT v8 and later

- Property **RightToBeNotified** As [RightToBeNotifiedStateType](#)

Retrieve/modify GDPR right to be notified mode (RightToBeNotifiedStateType)

EFT v8 and later

- Property **RightToExport** As [RightStateType](#)

Retrieve/modify GDPR right to export mode (RightStateType)

EFT v8 and later

- Property **RightToObject** As [RightStateType](#)

Retrieve/modify GDPR right to object mode (RightStateType)

EFT v8 and later

- Property **RightToRectify** As [RightStateType](#)

Retrieve/modify GDPR right to rectify mode (RightStateType)

EFT v8 and later

- Property **RightToRestrict** As [RightStateType](#)

Retrieve/modify GDPR right to restrict mode (RightStateType)

EFT v8 and later

- Property **RiskAssessment** As [RiskAssessmentType](#)

Retrieve/modify GDPR risk assessment mode (RiskAssessmentType)

EFT v8 and later

- Property **RiskMitigation** As [RiskMitigationType](#)

Retrieve/modify GDPR risk mitigation mode (RiskMitigationType)

EFT v8 and later

- Property **RiskMitigationReason** As String

Retrieve/modify GDPR risk mitigation reason (text)

EFT v8 and later

- Property **TerritorialScope** As [TerritorialScopeType](#)

Retrieve/modify GDPR territorial scope (TerritorialScopeType)

EFT v8 and later

- Property **TransferSafeguards** As [TransferSafeguardsType](#)

Retrieve/modify GDPR transfer safeguards mode (TransferSafeguardsType)

EFT v8 and later

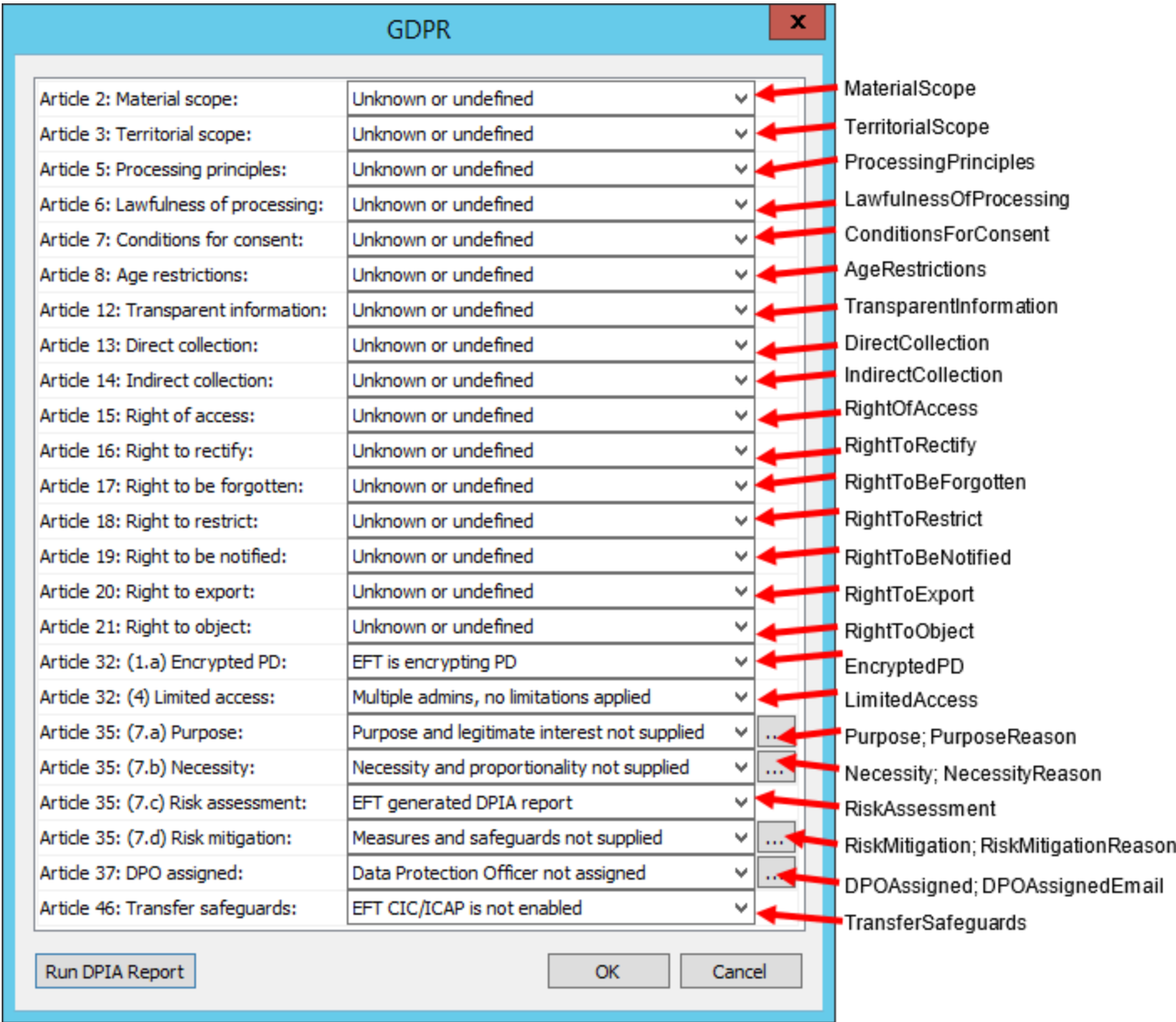
- Property **TransparentInformation** As [TransparentInformationType](#)

Retrieve/modify GDPR transparent information mode (TransparentInformationType)

EFT v8 and later

[Dialog Equivalents](#)

The ICIGDPRSettings interface properties correlate to the GDPR dialog box on the Site > Web tab.



ICIPrivacyPolicy Interface

The ICIPrivacyPolicy interface is used to specify the Privacy Policy on the Site for all web portals. The Privacy Policy applies to new account registration, anonymous pickup/dropoff, and after first login.

Members

- Property **AgreementLabels** As String
Retrieves or modifies

```
HRESULT AgreementLabels([out, retval] BSTR *pVal);
HRESULT AgreementLabels([in] BSTR val);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.AgreementLabels = "C:\pplabels.json"
```

EFT v8 and later

- Property **AgreementPath** As String

Retrieves or modifies privacy policy agreement path.

```
HRESULT AgreementPath([out, retval] BSTR *pVal);
HRESULT AgreementPath([in] BSTR val);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.AgreementPath = "C:\pp.json"
```

EFT v8 and later

- Property **EffectiveDate** As Variant

Retrieves or modifies privacy policy effective date.

```
HRESULT EffectiveDate([out, retval] VARIANT *pVal);
HRESULT EffectiveDate([in] VARIANT newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.WScript.echo pp.EffectiveDate
```

EFT v8 and later

- Property **Mode** As [GDPRPrivacyPolicyMode](#)

Retrieves or modifies Privacy Policy mode.

```
HRESULT Mode([out, retval] GDPRPrivacyPolicyMode *pVal);
HRESULT Mode([in] GDPRPrivacyPolicyMode newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.Mode = GDPRPrivacyPolicyMode_Shown
```

EFT v8 and later

- Property **PreventFTPSUntilConsentGranted** As Boolean

Retrieves or modifies "Prevent S/FTP/S until PP is agreed to" option state.

```
HRESULT PreventFTPSUntilConsentGranted([out, retval]
VARIANT_BOOL *pVal);
HRESULT PreventFTPSUntilConsentGranted([in] VARIANT_BOOL
newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.PreventFTPSUntilConsentGranted
```

EFT v8 and later

- Property **RevertToImpliedConsentForNotEUDataSubjects** As Boolean

Retrieves or modifies "Revert to implied consent for users identified as not being EU data subjects" option state.

```
HRESULT RevertToImpliedConsentForNotEUDataSubjects([out,
retval] VARIANT_BOOL *pVal);
HRESULT RevertToImpliedConsentForNotEUDataSubjects([in]
VARIANT_BOOL newVal);
```

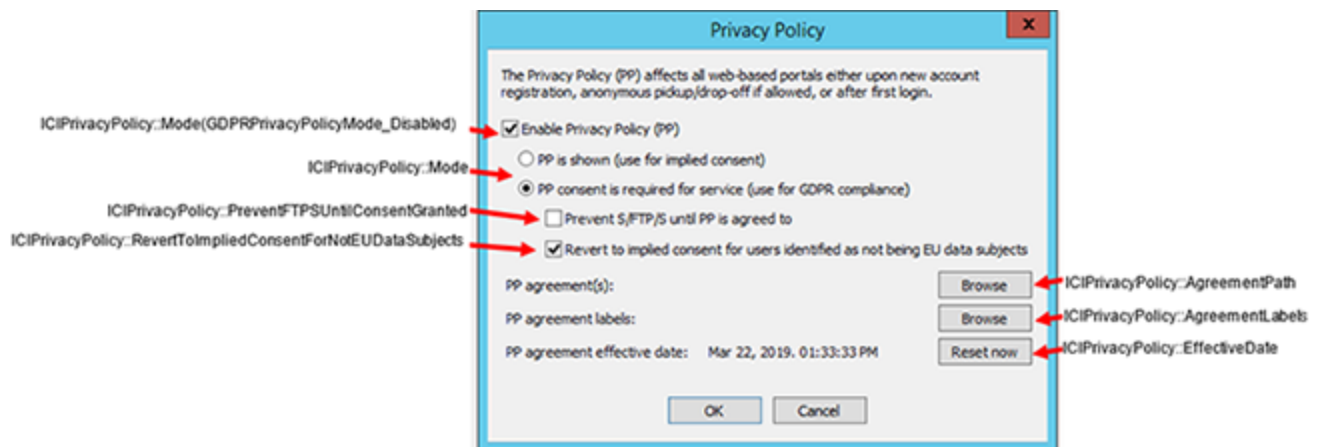
Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.RevertToImpliedConsentForNotEUDataSubjects =False
```

EFT v8 and later

Dialog Box Equivalents

The Privacy Policy dialog box is available from the Site > Web tab.



ICITermsOfService Interface

The ICITermsOfService interface provides the properties of the Terms of Service.

Members

- Property **AgreementLabels** As String

Retrieves or modifies Terms of Service labels.

```
HRESULT AgreementLabels([out, retval] BSTR *pVal);
HRESULT AgreementLabels([in] BSTR val);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.AgreementLabels = "C:\toslabels.json"
```

EFT v8 and later

- Property **AgreementPath** As String

Retrieves or modifies Terms of Service agreement path.

```
HRESULT AgreementPath([out, retval] BSTR *pVal);
HRESULT AgreementPath([in] BSTR val);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.AgreementPath = "C:\tos.json"
```

EFT v8 and later

- Property **EffectiveDate** As Variant

Retrieves or modifies Terms of Service effective date.

```
HRESULT EffectiveDate([out, retval] VARIANT *pVal);
HRESULT EffectiveDate([in] VARIANT newVal);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
WScript.echo tos.EffectiveDate
```

EFT v8 and later

- Property **Mode** As [GDPRTermsOfServiceMode](#)

Retrieves or modifies Terms of Service mode.

```
HRESULT Mode([out, retval] GDPRTermsOfServiceMode *pVal);
HRESULT Mode([in] GDPRTermsOfServiceMode newVal);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.Mode = GDPRTermsOfServiceMode_Implied
```

EFT v8 and later

- Property **PreventFTPSUntilAgreed** As Boolean

Retrieves or modifies “Prevent FTPS connection until agreed” option state.

```
HRESULT PreventFTPSUntilAgreed([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreventFTPSUntilAgreed([in] VARIANT_BOOL newVal);
```

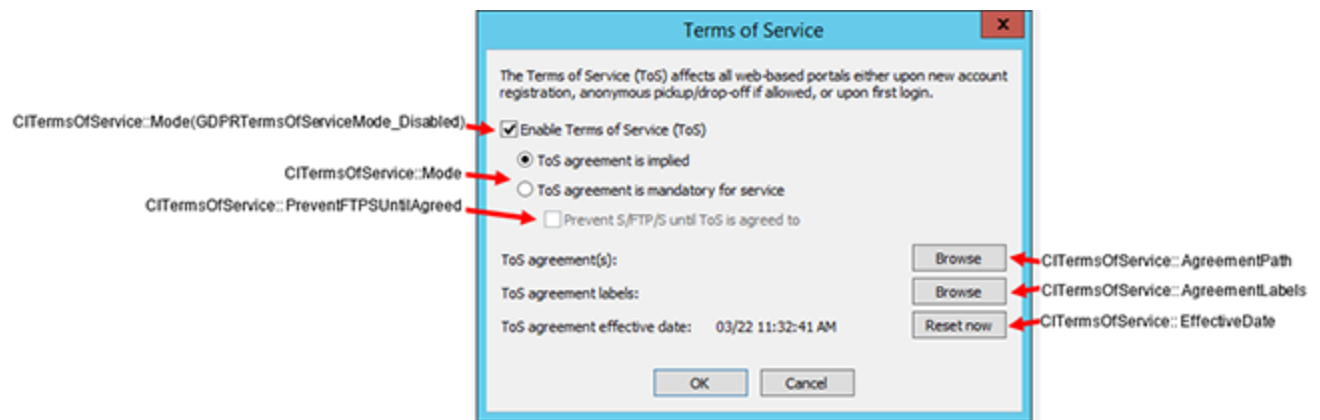
Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.PreventFTPSUntilAgreed =True
```

EFT v8 and later

Dialog Box Equivalents

The Terms of Service dialog box is available from the Site > Web tab.



Server-Level Objects

This chapter contains Server-related COM objects.

ICIServerVersion Interface

Use the ICIServerVersion interface to get the version of EFT you are using.

- Property **GetServerVersion**(Host As String, nPort As Long) As DWord (read-only)

```
HRESULT GetServerVersion([in] BSTR Host, [in] long nPort,
[out, retval] DWORD *serverVersion);
```

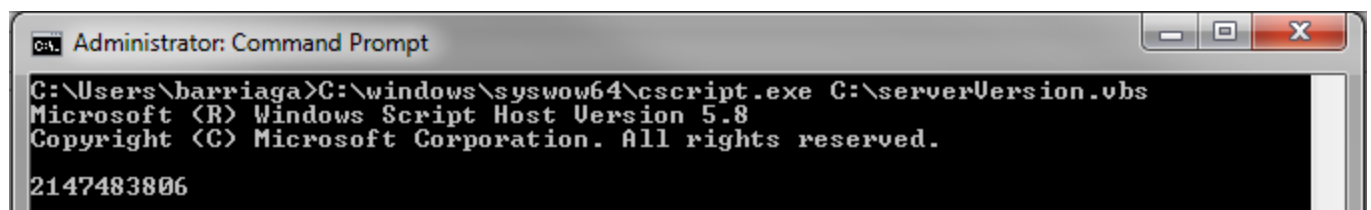
EFT v7.4 and later

Example

This script:

```
Set SFTPServer1 = WScript.CreateObject
("SFTPCOMInterface.CIServerVersion")
txtServer = "localhost"
txtPort = "1100"
WScript.Echo SFTPServer1.GetServerVersion(txtServer,txtPort)
```

Outputs the following:



Here are the version numbers as documented in source code (Enterprise version begins with "0x8000", SMB with "0x0000"):

```
#define EFT_SERVER_VER_7_1_1 0x00000098
```

```
#define EFT_SERVER_VER_7_2_0 0x00000099
```

```
#define EFT_SERVER_VER_7_2_1 0x0000009A
```

```
#define EFT_SERVER_VER_7_3_2 0x0000009B
```

```
#define EFT_SERVER_VER_7_4_0 0x0000009C
```

Running the output of the above COM script through the converter, we get the following:

The screenshot shows a web browser window with the address bar displaying `www.binaryhexconverter.com/decimal-to-hex-converter`. The page title is "BinaryHex Converter". Below the title, the heading "Decimal to Hexadecimal Converter" is displayed. A text block explains the tool's usage: "To use this **decimal to hex converter** tool, you have to type a decimal value like 79 into the left field below, and then hit the Convert button. Therefore, you can convert up to 19 decimal characters (max. value of 9223372036854775807) to hex." Below this text are three social media buttons: Facebook, Google+, and Twitter. The main conversion area has two input fields: "Decimal Value (max: 9223372036854775807)" containing the value "2147483806" and "Hexadecimal Value" containing the value "8000009E". A green "Convert" button is positioned below the decimal input field. To the right of the hexadecimal input field, it says "swap conversion: Hex to Decimal".

ICIServer Interface

Use the ICIServer interface to manage the Server and access the Server's attributes.

```
interface ICIServer : IDispatch
```

- [GetAdminIPs](#) is a list of all IP addresses assigned to the EFT computer.
- [ListenIP](#) is a deprecated property from previous versions in which EFT's administrative subsystem listened on only one IP address; instead use [ListenIPs](#).
- [CreateAutomationAdmin](#) is almost the same as [CreateAdmin](#) in that it creates an EFT-managed administrator account. The difference is that an automated administrator's original password is not reset upon first connection regardless of administrator password policy. An automated administrator account is created only via the API, operates via the API, and does not require an initial password change.

Jump to: [ARM](#) [B](#) [C](#) [D](#) [E](#) [G](#) [H](#) [I](#) [K](#) [L](#) [LOG](#) [M](#) [P](#) [R](#) [S](#) [MTP](#) [T](#) [U](#) [W](#)

Dialog Box Equivalents

Members (listed alphabetically)

- Property [ActiveDownloads](#) As Long read-only

Retrieves the number of active downloads.

```
HRESULT ActiveDownloads([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Active downloads " & CStr(oServer.ActiveDownloads)
```

EFT v6.3 and later

- Property [ActiveUploads](#) As Long read-only

Retrieves the number of active uploads.

```
HRESULT ActiveUploads([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Active uploads " & CStr(oServer.ActiveUploads)
```

EFT v6.3 and later

- Sub **AddContentIntegrityControlProfile**(pProfile As Object)

Adds a new profile to use with an ICAP (antivirus and DLP) server.

```
HRESULT AddContentIntegrityControlProfile([in] IDispatch* pProfile);
```

EFT 7.0.3 and later

- Sub [AddIPAccessRule](#)(bstrMask As String, bAllow As Boolean, nRulePos As Long, [bstrReason As String = ""])

Adds allowed or denied IP mask for Remote Administration. (default value and Reason added in EFT v8)

```
HRESULT AddIPAccessRule(
    [in] BSTR bstrMask,
    [in] VARIANT_BOOL bAllow,
    [in] long nRulePos,
    [in, defaultvalue("")] BSTR bstrReason); //position in list (0 is first)
```

Example (VBScript):

```
oServer.AddIPAccessRule("1.2.3.4", false, 2)
```

See also [GetIPAccessRules](#) and [RemoveIPAccessRule](#), and the following topics:

- [ICIAutobanIPAccessRule Interface](#)
- [ICIBannedIP Interface](#)
- [ICIIPAccessRule Interface](#)
- [ICIManualIPAccessRule Interface](#)
- [How Do the IP Access/Ban Methods and Properties Work?](#)

EFT v6.3 and later

- Property **AdminAccounts** As Variant read-only

For an example using this property, refer to the RetrieveAdminUser script in [Script Examples](#).

See also [ICIAdminPermission - Administrator Permission Interface](#) and [ICIAdminAccount - Administrator Account Interface](#).

Retrieves an array of administrator accounts.

```
HRESULT AdminAccounts([out,retval] VARIANT* pVal);
```

Example (VBScript):

```
For Each admin In oServer.AdminAccounts
MsgBox admin.Login
Next
```

Example (C#):

```
object[] accounts = (object[])m_server.AdminAccounts;
```

EFT v6.3 and later

- Property [AdminPort](#) As Long

Retrieves or specifies the port used for remote administration connections to administer the Server. You must restart the server to apply any change to the server administrator port.

```
HRESULT AdminPort([out, retval] long *pVal);
HRESULT AdminPort([in] long newVal);
```

All versions

- Property [AllowClearCommandChannel](#) As Boolean

Allows or prohibits a clear command channel on the server.

```
HRESULT AllowClearCommandChannel([out, retval] VARIANT_BOOL
*pVal);
HRESULT AllowClearCommandChannel([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
oServer.AllowClearCommandChannel = true
```

EFT v5.0.1 and later

- Property [AllowRemoteAdministration](#) As Boolean

Allows or prohibits remote administration of the server.

```
HRESULT AllowRemoteAdministration([out, retval] VARIANT_BOOL
*pVal);
HRESULT AllowRemoteAdministration([in] VARIANT_BOOL newVal);
```

All versions

- Property [AllowUnprotectedDataChannel](#) As Boolean

Allows or prohibits an unprotected data channel on the server for the FTP Protocol.

```
HRESULT AllowUnprotectedDataChannel([out, retval] VARIANT_
BOOL *pVal);
HRESULT AllowUnprotectedDataChannel([in] VARIANT_BOOL
newVal);
```

Example (VBScript):

```
oServer.AllowUnprotectedDataChannel = true
```

EFT v5.0.1 and later

- Sub **ApplyChanges()**

Applies changes to confirm and set changes you have made. (Any "Set" or "Write" method applies the change made immediately.)

```
HRESULT ApplyChanges();
```

All versions; does not apply to HA nodes

- Property [ARMAuthenticationType](#) As [ARMAuthenticationType](#)

For an example using this property, refer to the BackupEx script in [Script Examples](#).

Retrieves or specifies the type of database authentication, SQL or Windows.

```
HRESULT ARMAuthenticationType([out, retval]
ARMAuthenticationType*
    pVal);
HRESULT ARMAuthenticationType([in] ARMAuthenticationType
newVal)
```

Example (VBScript):

Retrieving:

```
If WindowsAuthentication = oServer.ARMAuthenticationType
Then
    ' some code
End If
```

Specifying:

```
oServer.ARMAuthenticationType = SQLServerAuthentication
```

EFT v6.3 and later

- Property **ARMConnectionStatus** As Boolean

Retrieves the database connection status.

```
HRESULT ARMConnectionStatus([out, retval] VARIANT_BOOL*
pVal);
```

EFT v7.4 and later

- Property **ARMCommitEventRuleTransferLog** As [DiagLoggingER](#)

```
HRESULT ARMCommitEventRuleTransferLog([out, retval]
DiagLoggingER
    *pVal);
HRESULT ARMCommitEventRuleTransferLog([in] DiagLoggingER
newVal)
```

EFT v7.4.2 and later

- Property [ARMDatabaseName](#) As String

For an example using this method, refer to the BackupEx script in [Script Examples](#).

Retrieves or specifies the Auditing and Reporting database name.

```
HRESULT ARMDatabaseName([out, retval] BSTR *pVal);
HRESULT ARMDatabaseName([in] BSTR newVal);
```

Example (VBScript):

```
Dim strDBName: strDBName = oServer.ARMDatabaseName
WScript.Echo "ARM Database: " & strDBName
```

All versions

- Property [ARMDatabaseType](#) As [ARMDatabaseType](#)

Retrieves or specifies the database type.

```
HRESULT ARMDatabaseType([out, retval] ARMDatabaseType*
pVal);
HRESULT ARMDatabaseType([in] ARMDatabaseType newVal);
```

Example (VBScript):

Retrieving:

```
If Oracle = oServer.ARMDatabaseType Then
' some code
End If
```

Specifying:

```
oServer.ARMDatabaseType = SQLServer
```

EFT v6.3 and later

- Property [ARMFailureAuditFolder](#) As String

Retrieves or specifies the folder used for logging ARM audit failures.

```
HRESULT ARMFailureAuditFolder([out, retval] BSTR* pVal);
HRESULT ARMFailureAuditFolder([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
Auditfolder = oServer.ARMFailureAuditFolder
```

Specifying:

```
oServer.ARMFailureAuditFolder = "C:\some_folder"
```

EFT v6.3 and later

- Property [ARMFailureEmailList](#) As String

Retrieves or specifies a list of emails that are notified of ARM failure.

```
HRESULT ARMFailureEmailList([out, retval] BSTR* pVal);
HRESULT ARMFailureEmailList([in] BSTR newVal);
```

Example (VBScript):

```
strEmailList = oServer.ARMFailureEmailList If strEmailList
""
Then strEmailList = strEmailList + "; " End If strEmailList
=
strEmailList + "NewRecipientName
SomeAddratSomeMailServer.com"
oServer.ARMFailureEmailList = strEmailList
```

EFT v6.3 and later

- Property **ARMHealConnection** As Boolean

Enable or disable ARM healing connection mechanism.

```
HRESULT ARMHealConnection([out,retval] VARIANT_BOOL* pVal);
HRESULT ARMHealConnection([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If MyServer.ARMHealConnection Then 'Healing connection
mechanism
    is enabled
...
End If
```

Specifying:


```
MyServer.ARMHealConnection = False
'Healing connection mechanism have disabled
```

EFT v7.0 and later

- Property **ARMNotifyOnDisconnect** As Boolean

Enable or disable notify on ARM database disconnect by sending an e-mail to the recipients defined on [ARMFailureEmailList](#).

```
HRESULT ARMNotifyOnDisconnect([out,retval] VARIANT_BOOL*
pVal);
HRESULT ARMNotifyOnDisconnect([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If MyServer.ARMNotifyOnDisconnect Then
'Notify on ARM database disconnect is enabled
...
End If
```

Specifying:

```
MyServer.ARMNotifyOnDisconnect = False
'Notify on ARM database disconnect is disabled
```

EFT v7.0 and later

- Property **ARMNotifyOnReconnect** As Boolean

Enable or disable notify on ARM database reconnect by sending an e-mail to the recipients defined on [ARMFailureEmailList](#).

```
HRESULT ARMNotifyOnReconnect([out,retval] VARIANT_BOOL*
pVal);
```

```
HRESULT ARMNotifyOnReconnect([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If MyServer.ARMNotifyOnReconnect Then
'Notify on ARM database Reconnect is enabled
```

```
...
End If
```

Specifying:

```
MyServer.ARMNotifyOnReconnect = False
'Notify on ARM database reconnect have disabled
```

EFT v7.0 and later

- Property [ARMPassword](#) As String

For an example using this method, refer to the BackupEx script in [Script Examples](#).

Retrieves or specifies the password for logging in to the ARM Database.

```
HRESULT ARMPassword([out, retval] BSTR *pVal);
HRESULT ARMPassword([in] BSTR newVal);
```

Example:

```
Dim strNewPassword: strNewPassword = "newpassword"
oServer.ARMPassword = strNewPassword
```

All versions

- Function [ARMReconnect\(\)](#) As Boolean

Specify whether ARM should reconnect after disconnection.

```
HRESULT ARMReconnect([out,retval] VARIANT_BOOL* pVal);
```

Example (VBScript):

```
If Not oServer.ARMReconnect() Then
MsgBox "Reconnection failed."
End If
```

EFT v6.3 and later

- Property **ARMReconnectPeriod** As DWord

Retrieves or specifies the period to reconnect to the ARM database. If a database error occurs and the ARM healing mechanism is enabled, then EFT tries to reconnect to the database every [ARMReconnectPeriod](#) seconds. The ARMReconnectPeriod can be set from 7 to 86400 seconds (24 hours). The default is 7 seconds.

```
HRESULT ARMReconnectPeriod([out,retval] ULONG* pVal);
HRESULT ARMReconnectPeriod([in] ULONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Reconnect period: " & CStr
(MyServer.ARMReconnectPeriod)
```

Specifying:

```
MyServer.ARMReconnectPeriod = 60 'every one minute
```

EFT v7.0 and later

- Property [ARMServerName](#) As String

For an example using this method, refer to the BackupEx script in [Script Examples](#).

Retrieves or specifies the name of the server on which the Auditing and Reporting database is located.

```
HRESULT ARMServerName([out, retval] BSTR *pVal);
HRESULT ARMServerName([in] BSTR newVal);
```

Example (VBScript):

```
Dim strServerName: strServerName = oServer.ARMServerName
oServer.ARMServerName = "localhost\GlobalSCAPE"
```

All versions

- Property [ARMStopAuditingOnError](#) As Boolean

Retrieves or specifies whether ARM should stop auditing when it encounters an error.

```
HRESULT ARMStopAuditingOnError([out, retval] VARIANT_BOOL*
pVal);
HRESULT ARMStopAuditingOnError([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If oServer.ARMStopAuditingOnError Then
    ' some code
End If
```

Specifying:

```
oServer.ARMStopAuditingOnError = True
```

EFT v6.3 and later

- Function [ARMTestConnection\(\)](#) As Boolean

For an example using this method, refer to the ARM script in [Script Examples](#).

Retrieves or specifies whether to test the ARM connection.

```
HRESULT ARMTestConnection([out,retval] VARIANT_BOOL* pVal);
```

Example (VBScript):

```
If Not oServer.ARMTestConnection() Then
    MsgBox "Database is unavailable!"
End If
```

EFT v6.3 and later

- Property [ARMUserName](#) As String

For an example using this property, refer to the BackupEx script in [Script Examples](#).

Retrieves or specifies the username used to connect to the Auditing and Reporting database.

```
HRESULT ARMServerName([out, retval] BSTR *pVal);
HRESULT ARMServerName([in] BSTR newVal);
```

Example (VBScript):

```
Dim strUserName : strUserName = oServer.ARMUserName
oServer.ARMUserName = "admin"
```

All versions

- Function **Auditor()** As [CIAudit](#)

Retrieves or specifies log audit information in the Auditing and Reporting Module (ARM) related to the Secure Ad Hoc Transfer module.

```
HRESULT Auditor([out,retval] ICIAudit** ppAuditor);
```

EFT v6.0 and later

- Property **AutoSave** As Boolean

Used for better performance in special cases and available only for COM admin sessions.

When ICIServer::AutoSave is FALSE settings changes do not save in the Transfer Engine (TE) and you have to call ICIServer::ApplyChanges manually.

When ICIServer::AutoSave is TRUE every settings change is saved (usual behavior).

```
HRESULT AutoSave([out, retval] VARIANT_BOOL *pVal);
HRESULT AutoSave([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Auto save mode: " & CStr(oServer.AutoSave)
```

Specifying:

```
oSFTPServer.AutoSave = FALSE
Dim strName
strName = "User_ "
kBase = 1
kAdd = 100000
For j = kBase To kAdd
oSite.CreateUserEx strName+Cstr(j),Cstr(j),0,Cstr(j),Cstr
(j),True,False,
    "Default settings"
Next
oSFTPServer.AutoSave = TRUE
Wscript.echo "Saving Settings..."
oSFTPServer.ApplyChanges
Wscript.echo "Saved "
```

EFT v6.2.18 and later; does not apply to HA nodes

- Property **AvailableEvents** As Variant read-only

Retrieve a list of the Events defined on EFT. Its value is a variant containing the array of [ICIEventInfo](#).

```
HRESULT AvailableEvents([out, retval] VARIANT* pVal);
```

Example (VBScript):

```
For Each event In Server.AvailableEvents
  MsgBox event.Name & " event is available"
Next
```

EFT v6.1 and later

- Property [AvailablePGPKeys](#) As Variant read-only

Retrieve a list of EFT's PGP Keys. Its value is a variant containing the array of [ICIPGPKeyInfo](#)

```
HRESULT AvailablePGPKeys([out, retval]VARIANT* pVal);
```

Example (VBScript):

```
For Each key In Server.AvailablePGPKeys
  MsgBox key.Name & " key is available" Next
```

EFT v6.1 and later

- Property **AvailableReports** As Variant read-only

See also [ICIReportInfo - Report Information Interface](#).

Retrieve a list of the reports available on EFT. Its value is a variant containing the array of the [ICIReportInfo](#).

```
HRESULT AvailableReports([out, retval] VARIANT* pVal);
```

Example (VBScript):

```
For Each report In Server.AvailableReports
  MsgBox report.Name & " report is available"
Next
```

EFT v6.1 and later

- Property [AvailableSSHKeys](#) As Variant read-only

For an example using this property, refer to the SSH script in [Script Examples](#).

Retrieve a list of SSH public keys available for EFT. Its value is a variant containing the array of [ICISSHKeyInfo](#).

```
HRESULT AvailableSSHKeys([out, retval] VARIANT* pVal);
```

Example (VBScript):

```
For Each key In Server.AvailableSSHKeys MsgBox "Key " &
key.Name & "("
    & key.Fingerprint & ") is available"
Next
```

Example (C#):

```
keyIdList = (object[])m_server.AvailableSSHKeys;
```

EFT v6.1 and later

- Function **AvailableSSLVersions()** As [CISSLVersions](#)

The AvailableSSLVersions property of ICIServer returns an [CISSLVersions](#) object, which is an array of [ICISSSLVersion](#) objects. The BitFlag property of each ICISSSLVersion is the value you would need to include when setting the [SSLVersionMask](#) property of ICIServer to enable the use of a particular SSL protocol version.

For example, if you wanted to enable TLS v1, v1.1, and v1.2, the value would be 25:

Protocol Version	Bitwise value
TLSv1	1
TLS v1.1	8
TLS v1.2	16

$$1 + 8 + 16 = 25$$

EFT v7.1.1 and later

- Property [AverageSpeed](#) As Long read-only

Retrieve the average transfer speed of the server.

```
HRESULT AverageSpeed([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Average speed is " & CStr(oServer.AverageSpeed) &
"KB/s"
```

EFT v6.3 and later

- Sub **BackupServerConfiguration(bstrFilePath As String)**

For an example using this method, refer to the Backup script in [Script Examples](#).

Retrieves or specifies the path at which to save server backup files.

```
HRESULT BackupServerConfiguration([in] BSTR bstrFilePath);
```

Example (VBScript):

```
oServer.BackupServerConfiguration "C:\lastbackup.bak"
```

EFT v6.3 and later

- Property [CertificateFilePath](#) As String

Retrieves or specifies the path to the server's certificate files.

```
HRESULT CertificateFilePath([out, retval] BSTR *pVal);
HRESULT CertificateFilePath([in] BSTR newVal);
```

All versions

- Sub [ChangeAdminPassword](#)(bstrAdminUser As String, bstrNewPassword As String)

Change the password for an administrator account on the server.

```
HRESULT ChangeAdminPassword([in] BSTR bstrAdminUser, [in]
BSTR bstrNewPassword);
```

Example (VBScript):

```
oServer.ChangeAdminPassword( "subadmin", "newpassword")
```

EFT 4.3.4 and later

- Property [CipherList](#) As String

Refer to "Using Ciphers for Inbound SSL Connections" in the EFT documentation for a complete list of allowed ciphers and how to properly format the cipher string.

Retrieves or specifies the cipher list.

```
HRESULT CipherList([out, retval] BSTR *pVal);
HRESULT CipherList([in] BSTR newVal);
```

Examples (VBScript):

```
Dim strCipherList: strCipherList = oServer.CipherList
Dim strNewList
```



```
strNewList = "ALL:!EXPORT:!LOW"
SFTPServer.CipherList = strNewList
Dim strCipherList: strCipherList = SFTPServer.CipherList
WScript.Echo strCipherList
```

CipherList must be delimited by colon (:). For example,

```
"AES256-SHA:CAMELLIA256-SHA:DES-CBC3-SHA:AES128-SHA:IDEA-
CBC-SHA:RC4-MD5"
```

EFT v5.0.1 and later

- **Sub Close()**

For an example using this method, refer to the Backup script or GetCertInfo script in [Script Examples](#).

Close the COM administrative connection to the server.

```
HRESULT Close();
```

All versions

- **Sub CloseServerService()**

Method used to stop the server service

```
HRESULT CloseServerService(void);
```

EFT v8.0.6 and later

- Property [ConfigurationFilePath](#) As String

Retrieves or specifies the path to the configuration files.

```
HRESULT ConfigurationFilePath([out, retval] BSTR* pVal);
HRESULT ConfigurationFilePath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Server setting folder is " & server.
ConfigurationFilePath
```

Specifying:

```
oServer.ConfigurationFilePath = "C:\some_folder"
oServer.StopServerService("127.0.0.1")
oServer.StartServerService("127.0.0.1")
```

EFT v6.3 and later

- Sub **Connect**(Host As String, nPort As Long, Login As String, Password As String)

For an example using this method, refer to the Backup script or GetCertInfo script in [Script Examples](#).

Connect to the local or remote server as the administrator. (In EFT v6.2 and later, use [ConnectEx](#) to connect using an IWA or Windows administrator login.)

```
HRESULT Connect([in] BSTR Host,[in] long nPort,[in] BSTR
Login,[in] BSTR Password);
```

Example (VBScript):

```
Dim Server As CIServer 'assumes the SFTPCOMInterface Type
Library is referenced by the VB IDE
Dim Site As CISite 'instantiates the ISite interface
Dim Sites As CISites 'instantiates the ISites interface
txtServer = "127.0.0.1" 'local IP address of server
txtPort = "1000" 'admin port
txtUserName = "Admin" 'admin username
txtPassword = "mypass" 'admin password
On Error Resume Next 'verify it could connect ok
Server.Connect txtServer, txtPort, txtUserName, txtPassword
'now connect using supplied information If Err.Number 0 Then
MsgBox "Error connecting to '" & txtServer & ":" & txtPort &
"' -- " & Err.Description
& " [" & CStr(Err.Number) & "]", vbInformation, "Error" 'all
this on one line!
End If
Set Sites = Server.Sites 'now retrieve the list of sites in
the server
Set Site = Sites.Item(0) 'Instantiate the first and only,
site
(note, this script assumes only one site present)
'continue rest of code...
```

All versions

- Property [ConnectedUsersNumber](#) As Long read-only

Retrieve the number of users connected to the Server.

```
HRESULT ConnectedUsersNumber([out, retval] BSTR *pVal);
```

Example (VBScript):

```
Dim iConnectedUses: iConnectedUses =  
oServer.ConnectedUsersNumber
```

EFT 3.5.1 and later

- Sub **ConnectEx**(Host As String, nPort As Long, nLoginType As AdminLoginType, Login As String, Password As String)

Log in to EFT using Integrated Windows Authentication (IWA) or Windows administrator login credentials.

```
HRESULT ConnectEx(/*  
[in]*/ BSTR Host, /*  
[in]*/ long nPort, /*  
[in]*/ AdminLoginType nLoginType, /*  
[in]*/ BSTR Login, /*  
[in]*/ BSTR Password);
```

AdminLoginType

```
EFTLogin = 0, // With EFT-managed admin login and password  
(legacy style)  
IWALogin = 1, // With currently logged user credentials  
// (Integrated Windows Authentication, IWA;  
// for AD and local computer accounts)NetLogon = 2  
// With Windows login and password  
// (for AD and local computer accounts)
```

Examples (VBScript):

IWA:

```
SFTPServer.ConnectEx "EFTServer", 1100, IWALogin, "", ""
```

NetLogin:

```
SFTPServer.ConnectEx "localhost", 1100, 2, txtUserName,  
txtPassword
```

Logging in using local computer login or domain login:

```
//...
server.ConnectEx "192.168.0.1", 1100, IWALogin, "", ""
//...
server.ConnectEx "192.168.0.1", 1100, NetLogon, "PC1\user",
"User'sLocalPassword"
//...
server.ConnectEx "192.168.0.1", 1100, NetLogon, "DM1\user",
"User'sDomainPassword"
```

EFT v6.2 and later

- Property **ContentIntegrityControlProfileList** As Variant

Retrieves a list of Content Integrity Control profiles. (read only)

```
HRESULT ContentIntegrityControlProfileList([out, retval]
VARIANT *pVal);
```

EFT v7.0.3 and later

- Function [CreateAdmin](#)(bstrNewAdmin As String, bstrPassword As String, atype As [AdminAccountType](#), vbGroup As Boolean) As Object

For an example using this method, refer to the CreateAdminUser script in [Script Examples](#).

Create an administrator account on EFT.

```
HRESULT CreateAdmin(
[in] BSTR bstrNewAdmin,
[in] BSTR bstrPassword,
[in] AdminAccountType atype,
[in] VARIANT_BOOL vbGroup,
[out,retval] IDispatch** pAdminAccount);
```

Example (VBScript):

```
Set AdminAccount = server.CreateAdmin("admin", "swordfish",
EFTAdmin, False)
```

Example (C#):

```
ICIAdminAccount adminUser = (ICIAdminAccount)m_
server.CreateAdmin("Test_Administrator",
    "Tester!1", AdminAccountType.EFTAccount, false);
```

EFT v6.3 and later

- Function **CreateAdminID**(bstrLogin As String, type As [AdminAccountType](#), bGroup As Boolean) As [CIAdminID](#)

ID assigned to the CreateAdmin account.

```
HRESULT CreateAdminID(
    [in] BSTR bstrLogin,
    [in] AdminAccountType type,
    [in] VARIANT_BOOL bGroup,
    [out, retval] ICIAdminID** ppVal);
```

EFT v7.2 and later

- Function **CreateAutomationAdmin**(bstrNewAdmin As String, bstrPassword As String) As Object

Create an automation administrator account. This account does not require a password change, regardless of password security settings; the account is intended for use in an automated system that does not login via the administration interface. (No UI equivalent; for API only.)

```
HRESULT CreateAutomationAdmin([in] BSTR bstrNewAdmin,
    // admin name[in] BSTR bstrPassword,
    //admin password[out,retval] IDispatch** pAdminAccount);
//ICIAdmin Account
```

Example (VBScript):

```
Set admin = oServer.CreateAutomationAdmin("admin", "pass")
```

EFT v6.3.1 and later

- Function **CreateComplexPassword**(bstrUserName As String) As String

Create a complex password.

```
HRESULT CreateComplexPassword(
    [in] BSTR bstrUserName,
    [out, retval] BSTR *bstrPassword);
```

Example (VBScript):

```
pass = oServer.CreateComplexPassword ("user")
```

EFT v6.3.1 and later

- Function [CreatePGPKey](#)(bstrFullName As String, bstrEmailAddress As String, dtExpirationDate As Date, nKeyType As PGPKeyType, nKeyCipher As PGPKeyCipher, ulKeySize As DWord, bstrPassphrase As String) As String

Create a PGPKey on EFT.

```
HRESULT CreatePGPKey(
[in] BSTR bstrFullName,
[in] BSTR bstrEmailAddress,
[in] DATE dtExpirationDate,
[in] PGPKeyType nKeyType,
[in] PGPKeyCipher nKeyCipher,
[in] ULONG ulKeySize,
[in] BSTR bstrPassphrase,
[out,retval] BSTR* pVal);
```

PGPKeyType

- PGPKeyType_DSS = 0
- PGPKeyType_RSA = 1
- PGPKeyType_RSALegacy = 2

PGPKeyCipher

- PGPKeyCipher_IDEA = 0
- PGPKeyCipher_3DES = 1
- PGPKeyCipher_CAST5 = 2
- PGPKeyCipher_AES128 = 3
- PGPKeyCipher_AES192 = 4
- PGPKeyCipher_AES256 = 5
- PGPKeyCipher_TWOFISH = 6

Example (VBScript):

```
strKeyId = oServer.CreatePGPKey("Ivan Ivanuch",
"ipetrov@mail.com",
expDate, PGPKeyType_RSA, PGPKeyCipher_IDEA, 1024,
"password")
MsgBox "New key id is " & strKeyId
```

EFT v6.3 and later

- Sub [CreateSSHKey](#)(nKeyBits As Long, bstrPassword As String, bstrPrivateKeyFile As String, bstrPublicKeyFile As String, vbComFormat As Boolean, nSiteId As Long, bstrCopyTo As String)

Create an SSH Key on EFT. (In v8.0.4, nKeyBits is replaced with nKeyParams.)

```
HRESULT CreateSSHKey(
    [in] LONG nKeyBits,
    [in] BSTR bstrPassword,
    [in] BSTR bstrPrivateKeyFile,
    [in] BSTR bstrPublicKeyFile,
    [in] VARIANT_BOOL vbComFormat,
    [in] LONG nSiteId,
    [in] BSTR bstrCopyTo);
```

Example (VBScript):

```
oServer.CreateSSHKey 1024, "passphrase",
    "C:\Private", "C:\Public.pub", False, 0, ""
```

EFT v6.3 and later

- Sub **CreateSSHKeyRemotely**(nKeyBits As Long, bstrPassword As String, bstrPrivateKeyFile As String, bstrPublicKeyFile As String, nSiteId As Long, bstrCopyTo As String)

Create an SSK Key on EFT remotely. (In v8.0.4, nKeyBits is replaced with nKeyParams.)

```
HRESULT CreateSSHKeyRemotely(
    [in] LONG nKeyBits,
    [in] BSTR bstrPassword,
    [in] BSTR bstrPrivateKeyFile,
    [in] BSTR bstrPublicKeyFile,
    [in] LONG nSiteId,
    [in] BSTR bstrCopyTo);
```

EFT v7.411 and later

- Sub [CreateSSLCertificate](#)(bstrKeyFile As String, bstrReqfile As String, bstrcertfile As String, bstrPassword As String, bstrCountryName As String, bstrStateOrProvinceName As String, bstrLocalityName As String, bstrOrganizationName As String, bstrOrganizationalUnitName As String, bstrCommonName As String, bstrEmailAddress As String, nBits As Long, nDays As Long, CertFormat As [CertificateFormat](#), PrivateKeyFormat As CertificateFormat, nSiteId As Long,

vbUseCert As Boolean, vbMakeTrusted As Boolean)

NOTE: This method does not create a new path. That is, if you use "c:\test\cert.key" as the path and c:\test doesn't exist, the function appears to complete without error and no file/directory is created. Also, if executed from a non-local system, there is no feedback that execution of the function has been blocked for security reasons; it simply (silently) fails.

Create an SSL certificate on EFT.

```
HRESULT CreateSSLCertificate(
[in] BSTR bstrKeyfile,
[in] BSTR bstrReqfile,
[in] BSTR bstrcertfile,
[in] BSTR bstrpassword,
[in] BSTR bstrCountryName,
[in] BSTR bstrStateOrProvinceName,
[in] BSTR bstrLocalityName,
[in] BSTR bstrOrganizationName,
[in] BSTR bstrOrganizationalUnitName,
[in] BSTR bstrCommonName,
[in] BSTR bstrEmailAddress,
[in] LONG nBits,
[in] LONG nDays,
[in] CertificateFormat CertFormat,
[in] CertificateFormat PrivateKeyFormat,
[in] LONG nSiteId,
[in] VARIANT_BOOL vbUseCert,
[in] VARIANT_BOOL vbMakeTrusted);
```

Example (VBScript):

```
'This script is an example of using the COM API to call the
createSSLCertificate() function 'First check to see if the
path
C:\Certificates\ exists on the system (if not create one)
'This path is used in the files created below. (You can
change this value to whatever
location you want just update entire script.)
Set fileSystemObject = CreateObject
("Scripting.FileSystemObject")
Dim exists
If Not( exists ) Then
```



```

WScript.Echo "Please create a 'Certificates' folder on your
C:\ drive."
End If
' Instantiate COM object
Set SFTPServer = WScript.CreateObject
("SFTPCOMInterface.CIServer")
' Set connection information
'txtServer = IP Address = x.x.x.x = example :
"192.168.1.100"
"[IP ADDRESS HERE]"
"1100"
"Administrator"
"password"
' Connect to server
If Not Connect(txtServer, txtPort, txtUserName, txtPassword)
Then
End If
'Initialize all needed variables
Dim keyfile, reqfile, certfile, passphrase, country,
province,
city, organization, organizationUnit, commonname, email
Dim keylength, days, certFormat, privKeyFormat,
siteId, useCert, makeTrust
'Name and location of where you want the private key file
(.key),
request file (.csr), and certificate file (.crt) stored
'In this example, createSSLCertificate() will create
these 3 files at the location C:\Certificates\
"C:\Certificates\Cert.key"
"C:\Certificates\Cert.csr"
"C:\Certificates\Cert.crt"
'Passphrase for certificate
"passphrase"
'Location information for certificate
"US"
'State/Province
"TX"
'City/Town
"San Antonio"
'Organization Name/Unit
"GS"
"QA"

```

```

'Common Name (typically name or domain name associated with
the site)
"www.globalscape.com"
"globalscape@example.com"
'Key Length(in bits)
1024
21
'Certificate/PrivateKey Format PEM = (ASCII)
privKeyFormat = PEM 0
'Use Certification/Make Trusted
False
False
'With all these parameters ready, create SSL Certificate
'Close of COM Object
'Dispose of COM Object
Set SFTPServer = Nothing
Function Connect (serverOrIpAddress, port,
                username, password)
Error Resume Next
    SFTPServer.Connect serverOrIpAddress,
                port, username, password
If Err.Number 0 Then
CStr(err.Number) & "]", vbInformation,
    "Error"

False
Exit Function
End If
True
End Function

```

EFT v6.3 and later

- Function **CreateSSLCertificateRemotely** As String
and

Function **SignSSLCertificateRemotely** As Long

Used to create and sign an SSL certificate

```

HRESULT CreateSSLCertificateRemotely(
[in] BSTR bstrKeyfile,
[in] BSTR bstrReqfile,
[in] BSTR bstrcertfile,

```

```
[in] BSTR bstrpassword,
[in] BSTR bstrCountryName,
[in] BSTR bstrStateOrProvinceName,
[in] BSTR bstrLocalityName,
[in] BSTR bstrOrganizationName,
[in] BSTR
    bstrOrganizationalUnitName,
[in] BSTR bstrCommonName,
[in] BSTR bstrEmailAddress,
[in] LONG nBits,
[in] LONG nDays,
[in] CertificateFormat CertFormat,
[in] CertificateFormat PrivateKeyFormat,
[in] LONG nSiteId,
[in] VARIANT_BOOL vbUseCert,
[in] VARIANT_BOOL vbMakeTrusted);
```

```
HRESULT SignSSLCertificateRemotely(
[in] BSTR bstrRequest,
[in] BSTR bstrServCert,
[in] BSTR bstrPrivateKey,
[in] BSTR bstrPass,
[in] BSTR bstrOutCert,
[in] LONG nDays);
```

EFT v7.3.6 and later

- Property [DefaultPollTimeout](#) As Long

Retrieves or specifies the default value for the user database refresh interval (set to "never" by default).

```
HRESULT DefaultPollTimeout([out, retval] LONG* pVal);
HRESULT DefaultPollTimeout([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
Interval = oServer.DefaultPollTimeout
```

Specifying:

```
oServer.DefaultPollTimeout = -1
```

EFT v6.3 and later

- Sub **DeleteContentIntegrityControlProfile**(pProfileName As String)

```
HRESULT DeleteContentIntegrityControlProfile(
[in] BSTR pProfileName);
```

Removes the specified Content Integrity Control profile.

(See also the [ICContentIntegrityControlActionParams](#) ProfileName property.)

EFT v7.0.3 and later

- Property [DisconnectDueToTimeout](#) As Boolean

(On a PCI DSS-enabled account, disabling the timeout is not allowed.)

Enables/disables the timeout to disconnect the administration interface from EFT after administrator inactivity.

```
HRESULT DisconnectDueToTimeout([out, retval] VARIANT_BOOL*
pVal);
HRESULT DisconnectDueToTimeout(VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If Server.DisconnectDueToTimeout = VARIANT_TRUE Then
    'Admin timeout enabled
End If
```

Specifying:

```
'disable admin timeout
Server.DisconnectDueToTimeout = VARIANT_FALSE
```

EFT v6.5 and later

- Property [DisconnectDueToTimeoutPeriod](#) As Long

(On a PCI DSS-enabled account, setting a timeout longer than 15 minutes is not allowed.)

Set the timeout (in minutes) after administrator inactivity to disconnect the administration interface from EFT (and log out the administrator account).

Remark: Only values from 1 to 9999 are accepted

```
HRESULT DisconnectDueToTimeoutPeriod([out,retval] LONG*
pVal);
HRESULT DisconnectDueToTimeoutPeriod([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
If Server.DisconnectDueToTimeoutPeriod = 15 Then
    'There is 15 minutes admin timeout period
End If
```

Specifying:

```
`set admin timeout period to 5 minutes
Server.DisconnectDueToTimeoutPeriod = 5
```

EFT v6.5 and later

- Sub **DrainServer**(bRestartServer As Boolean)

Drain server allows in-progress transfers to continue, and prevents new ones from starting to gracefully drain connections and event rule activity.

```
HRESULT DrainServer([in] VARIANT_BOOL bRestartServer);
```

EFT v7.4 and later

- Property **EnableARM** As Boolean

For an example using this property, refer to the BackupEx script in [Script Examples](#).

Enable or disable Auditing and Reporting, or evaluate status of Auditing and Reporting.

```
HRESULT EnableARM([out, retval] VARIANT_BOOL *pVal);
HRESULT EnableARM([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
Dim bARMStatus: bARMStatus = oServer.EnableARMo
```

To set the state:

```
oServer.EnableARM = false
```

All versions

- Property **EventRuleChangeLog** As [ERChangeLog](#)

Retrieves or specifies whether adding comments to the Event Rule Change log is Off, Mandatory, or Optional.

```
HRESULT EventRuleChangeLog([out, retval] ERChangeLog *pVal);
HRESULT EventRuleChangeLog([in] ERChangeLog newVal);
```

EFT v7.2 and later

- Sub [ExportPGPKey](#)(bstrKeyId As String, vbIncludePrivate As Boolean, bstrFileName As String)

Export an Open PGP Key.

```
HRESULT ExportPGPKey(
[in] BSTR bstrKeyId,
[in] VARIANT_BOOL vbIncludePrivate,
[in] BSTR bstrFileName);
```

Example (VBScript):

```
oServer.ExportPGPKey "0x12345678", False,
"C:\PGP\pgpkey.asc"
```

EFT v6.3 and later

- Sub (lKeyID As Long, bzKeyFilePath As String)

Export an SSH key.

```
HRESULT ExportSSHKey([in] LONG lKeyID,[in] BSTR
bzKeyFilePath);
```

Example (VBScript):

```
oServer.ExportSSHKey nKeyId, "C:\SSHKeys\key.pub"
```

Example (C#):

```
m_server.ExportSSHKey(key.ID, keyPath);
```

For an example using this method, refer to the SSH script in [Script Examples](#).

EFT v6.3 and later

- Property **ForceAdminResetPasswordOnInitialLogin** As Boolean

```
HRESULT ForceAdminResetPasswordOnInitialLogin([out, retval]
VARIANT_BOOL *pVal);
HRESULT ForceAdminResetPasswordOnInitialLogin([in] VARIANT_
BOOL newVal);
```

EFT v7.4.6 and later

- Sub **GenerateReport**(pdispReportParams As Object, bzReportPath As String)

Generate an EFT report.

```
HRESULT GenerateReport(
[in] IDispatch* pdispReportParams,
[in] BSTR bzReportPath);
```

Example (VBScript):

```
Set ReportParams = CreateObject
("SFTPCOMInterface.CIReportActionParams")
'Setup report name, filters, period
' ...
ReportParams.ReportFileFormat = PDFFormat
Server.GenerateReport(ReportParams, "C:\Report\Report.pdf")
```

See also [ICIRReportActionParams Interface](#)

EFT v6.1 and later

- Function [GetAdminIPs\(\)](#) As Variant

Retrieve an array of IP addresses for the computer on which EFT is running. The IP addresses can be used as LISTENING IP addresses for administration or protocol services.

```
HRESULT GetAdminIPs([out,retval] VARIANT* pVar);
```

Example (VBScript):

```
Dim arIPS : arIPS = oServer.GetAdminIPs()
for i = LBound(arIPS) to UBound(arIPS)
WScript.Echo CStr(i) & " - " & arIPS(i)
next
```

EFT v4.3.4 and later

- Function **GetBackupArchiveInfo**(bstrFilePath As String, nLoginType As [AdminLoginType](#), bstrLogin As String, bstrPassword As String) As [CIBackupArchiveInfo](#)

Retrieves or specifies the backup archive information.

```
HRESULT GetBackupArchiveInfo(
    [in] BSTR bstrFilePath,
    [in] AdminLoginType nLoginType,
    [in] BSTR bstrLogin,
    [in] BSTR bstrPassword,
    [out,retval] IDispatch** pVal);
```

Example (VBScript):

```
Set oBackupSettings = oServer.GetBackupArchiveInfo
("C:\backup.bak",
    EFTLogin, "admin", "swordfish")
```

EFT v6.3 and later

- Function **GetComplexPasswordSettings**() As [CIComplexPasswordSettings](#)

Retrieve complex password settings.

```
HRESULT GetComplexPasswordSettings([out, retval]
    IDispatch** ppICComplexPasswordSettings);
```

Example (VBScript):

```
Set oPwdSettings = oServer.GetComplexPasswordSettings()
```

EFT v6.3 and later

- Function **GetContentIntegrityControlProfile**(profileName As String) As Long

Retrieves the name of the Content Integrity Control profile.

```
HRESULT GetContentIntegrityControlProfile([in] BSTR
    profileName, [out, retval] long *profileIndex);
```

EFT v7.0.3 and later

- Function **GetHighAvailabilityParams**() As Object and Sub **SetHighAvailabilityParams**(pdispParams As Object)

Retrieve/specify HA parameters.


```
HRESULT GetHighAvailabilityParams([out, retval] IDispatch**
    ppDispParams);
HRESULT SetHighAvailabilityParams([in] IDispatch*
    pDispParams);
```

EFT v7.0 and later

- Function [GetIPAccessRules\(\)](#) As Variant

Retrieve an array of remote administration IP address access rules.

```
HRESULT GetIPAccessRules([ out, retval] VARIANT *aMasks);
```

See also [AddIPAccessRule](#), [RemoveIPAccessRule](#), [MoveIPAccessRule](#) methods, and the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIManualIPAccessRule Interface](#)

ICIClientSettings : [GetIPAccessRules](#)

ICISite : [GetIPAccessRules](#)

Example (VBScript):

```
rules = oServer.GetIPAccessRules()
For Each key In rules
  If key.type = 0 Then
    WScript.echo "AutoBan"
  End if
  If key.type = 1 Then
    WScript.echo "Manual " + key.address
  End if
Next
```

EFT v6.3 and later

- Function [GetLocalIP](#)(nIP As Long) As String

Given an index, retrieves a Local IP address.

```
HRESULT GetLocalIP([in] long nIP, [out, retval] BSTR
    *prop);
```

All versions

- Function **GetLocalTime()** As String

Retrieve EFT's local time.

```
HRESULT GetLocalTime (
    [out, retval] BSTR *pTime);
```

All versions

- Sub [GetPGPKeyringSettings](#)(pPublicPath As String, pPrivatePath As String)

For an example using this method, refer to the PGP script in [Script Examples](#).

Retrieve the OpenPGP Keyring settings.

```
HRESULT GetPGPKeyringSettings (
    [out] VARIANT* pPublicPath,
    [out] VARIANT* pPrivatePath);
```

Example (VBScript):

```
oServer.GetPGPKeyringSettings strPublicKeyring,
strPrivateKeyring
```

Example (C#):

```
m_server.ImportSSHKey(@"C:\SSHKey.pub");
object publicPath = new object();
object privatePath = new object();
m_server.GetPGPKeyringSettings(out publicPath, out
    privatePath);
m_server.SetPGPKeyringSettings
(@"C:\filecryptdata\pubring.pgp",
    @"C:\filecryptdata\secring.pgp");
```

EFT v6.3 and later

- Function [GetServerAdminCount\(\)](#) As Long

Retrieve a count of administrator accounts defined on EFT.

```
HRESULT GetServerAdminCount ([out, retval] long* lCount);
```

Example (VBScript):

```
Dim i : i = oServer.GetServerAdminCount()
```

All versions

- Function **GetSiteChangeLog**(nSite As Int, [iDBrecsLimit As Int = 500000]) As Variant
Retrieve the change log on a Site.

```
HRESULT GetSiteChangeLog(
    [in] int nSite,
    [in, defaultvalue(500000)] int iDBrecsLimit,
    [out, retval] VARIANT* pVal);
```

EFT v7.2 and later

- Property **HACoherenceQueue** As String
Specify the HA coherence queue. (read only)

```
HRESULT HACoherenceQueue([out, retval] BSTR *pVal);
```

EFT v7.0 and later

- Property **HAConfigurationBackupNodeList** As Variant
Allows you to specify one or more nodes as the backup to execute Event Rules. This list can be overridden in the Event Rules.

```
HRESULT HAConfigurationBackupNodeList([out, retval] VARIANT
    *pVal);
HRESULT HAConfigurationBackupNodeList([in] VARIANT pVal);
```

EFT v7.2 and later

- Property **HAConfigurationDefaultNodeList** As Variant
Allows you to specify one or more nodes as the default to execute Event Rules. This list can be overridden in the Event Rules.

```
HRESULT HAConfigurationDefaultNodeList([out, retval] VARIANT
    *pVal);
HRESULT HAConfigurationDefaultNodeList([in] VARIANT pVal);
```

EFT v7.0.2 and later

- Property **HAConfigurationFilePath** As String
Retrieves the path to the HA configuration file. (read only)

```
HRESULT HAConfigurationFilePath([out, retval] BSTR* pVal);
```

EFT v7.0 and later

- Property **HAEnabled** As Boolean

Retrieves whether HA is enabled. (read only)

```
HRESULT HAEnabled([out, retval] VARIANT_BOOL *pVal);
```

EFT v7.0 and later

- Property **HAEventQueue** As String

Retrieves the event queue. (read only)

```
HRESULT HAEventQueue([out, retval] BSTR *pVal);
```

EFT v7.0 and later

- Property **HAFailureNotificationEmail** As String

EFT v7.0 and later

- Property **HAMulticastAddress** As String

Retrieves multicast address. (read only)

```
HRESULT HAMulticastAddress([out, retval] BSTR *pVal);
```

EFT v7.0 and later

- Property **HAMulticastPort** As Long

Retrieves multicast port. (read only)

```
HRESULT HAMulticastPort([out, retval] long *pVal);
```

EFT v7.0 and later

- Sub [ImportPGPKey](#)(bstrKey As String, bstrKeyFile As String)

Imports specified PGP key.

```
HRESULT ImportPGPKey(  
[in] BSTR bstrKeyPath,  
[out, retval] BSTR* pNewKeyID);
```

Example (VBScript):

```
KeyId = oServer.ImportPGPKey("C:\PGP\pgpkey.asc")
```

EFT v6.3 and later

- Function [ImportSSHKey](#)(bstrKeyPath As String) As Long

Import an SSH public key. Unlike other SSL certificate-related methods dealing with files, the key to import must be located on the computer running COM, not on the computer running EFT. This difference reflects EFT administration interface behavior.

```
HRESULT ImportSSHKey(
    [in] BSTR bstrKeyPath,
    // Public key path (on the computer running COM)
    [out, retval] long *plID);
// ID of the key imported (can be used to assign the key to
client)
```

Example (VBScript):

```
Client.SetSftpAuthenticationType "key"
keyID = Server.ImportSSHKey("C:\client.pub")
Client.SetSSHKeyID keyID
```

Example (C#):

```
m_server.ImportSSHKey(@"C:\SSHKey.pub");
object publicPath = new object();
object privatePath = new object();
m_server.GetPGPKeyringSettings(out publicPath, out
    privatePath);
m_server.SetPGPKeyringSettings
    (@@"C:\filecryptdata\pubring.pgp",
    @"C:\filecryptdata\sekring.pgp");
```

Example (C#):

```
int keyId = m_server.ImportSSHKey(keyPath);
```

For an example using this method, refer to the PGP script or SSH script in [Script Examples](#).

EFT v6.1 and later

- Property [InactiveAccountsRemoving](#) As Boolean

Enable or disable remove inactive administrator accounts feature.

```
HRESULT InactiveAccountsRemoving([out, retval] VARIANT_BOOL
*pVal);
HRESULT InactiveAccountsRemoving([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If MyServer.InactiveAccountsRemoving Then
  Inactive accounts will be removed
...
End If
```

Specifying:

```
MyServer.InactiveAccountsRemoving = False
Inactive admin account won't be removed.
```

EFT v7.0 and later

- Property [InvalidAttemptsPeriod](#) As Long

Retrieves or specifies the time period during which will be calculating invalid login attempts. Measured in minutes. Limitation: 1 = InvalidAttemptsPeriod = 99

```
HRESULT InvalidAttemptsPeriod([out, retval] LONG *pVal);
HRESULT InvalidAttemptsPeriod([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Invalid attempts period: " & CStr
(MyServer.InvalidAttemptsPeriod)
```

Specifying:

```
MyServer.InvalidAttemptsPeriod = 18
```

EFT v7.0 and later

- Property [IPAccessAllowedDefault](#) As Boolean

Retrieves or specifies the default IP Access restriction list for remote administration.

```
HRESULT IPAccessAllowedDefault([out, retval] VARIANT_BOOL
*pVal);
HRESULT IPAccessAllowedDefault([in] VARIANT_BOOL newVal);
```

True = All IP addresses are allowed except those specified

False = All IP addresses are denied except those specified

All versions

- Function **IsServerServiceStarted**(bstrHost As String) As Boolean

Determine if the EFT Server service is started.

```
HRESULT IsServerServiceStarted(
[in] BSTR bstrHost,
[out,retval] VARIANT_BOOL* pvResult);
```

Example (VBScript):

```
If Not oServer.IsServerServiceStarted("192.168.1.1")
Then
oServer.StartServerService("192.168.1.1")
End If
```

EFT v6.3 and later

- Property **IsStarted** As Boolean read-only

Determine if the EFT Server service is started.

```
HRESULT IsStarted(
[out, retval] VARIANT_BOOL* pVal);
```

Example (VBScript):

```
If Not oServer.IsStarted Then
oServer.Start()
End If
```

EFT v6.3 and later

- Property [KeyFilePath](#) As String

Retrieves or specifies the file path for EFT's private key files.

```
HRESULT KeyFilePath([out, retval] BSTR *pVal);
HRESULT KeyFilePath([in] BSTR newVal);
```

All versions

- Property [LastModifiedBy](#) As String read-only

Retrieve the name of the user who last modified EFT.

```
HRESULT LastModifiedBy([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Last modified by " & oServer.LastModifiedBy
```

Example (C#):

```
string lastModifiedBy = m_server.LastModifiedBy;
string lastModifiedTime = m_server.LastModifiedTime.ToString
("MM/dd/yyyy HH:mm:ss");
Console.WriteLine(string.Format("EFT properties Last
Modified: {0} by {1}", lastModifiedTime, lastModifiedBy));
```

See also **LastModifiedTime**, below.

For an example using this property, refer to the ServerStats script in [Script Examples](#).

EFT v6.3 and later

- Property [LastModifiedTime](#) As Date read-only

See also **LastModifiedBy**, above.

For an example using this property, refer to the ServerStats script in [Script Examples](#).

Retrieve the date and time when EFT was last modified.

```
HRESULT LastModifiedTime([out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Modification time is " & CStr
(oServer.LastModifiedTime)
```

Example (C#):

```
string lastModifiedBy = m_server.LastModifiedBy;
string lastModifiedTime = m_server.LastModifiedTime.ToString
```



```
("MM/dd/yyyy HH:mm:ss");
Console.WriteLine(string.Format("EFT properties Last
Modified: {0} by {1}", lastModifiedTime, lastModifiedBy));
```

EFT v6.3 and later

- Property [LimitLoginAttempts](#) As Boolean

Enable or disable lockout administrator accounts after incorrect login attempts feature.

```
HRESULT LimitLoginAttempts([out, retval] VARIANT_BOOL
*pVal);
HRESULT LimitLoginAttempts([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If MyServer.LimitLoginAttempts Then
'feature is enabled
End If
```

Specifying:

```
MyServer.LimitLoginAttempts = False
'feature is disabled
```

EFT v7.0 and later

- Property [ListenIP](#) As Long

In version 6.4 and later, use ListenIPs, below, instead.

Retrieves or specifies EFT's IP address.

```
HRESULT ListenIP([out, retval] long *pVal);
HRESULT ListenIP([in] long newVal);
```

All versions

- Property **ListenIPs** As Variant

Replaces **ListenIP**, above.

Retrieves or specifies the IP addresses on which EFT listens.

```
HRESULT ListenIPs([out, retval] VARIANT* pVal);
HRESULT ListenIPs([in] VARIANT newVal);
```

EFT v6.4 and later

- Property [LockoutPeriod](#) As Long

Retrieves or specifies the lockout period for administrator accounts after incorrect login attempts. Measured in minutes. Limitation: 30, 60, or 90

```
HRESULT LockoutPeriod([out, retval] LONG *pVal);
HRESULT LockoutPeriod([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Lockout Period: " & CStr(MyServer.LockoutPeriod)
```

Specifying:

```
MyServer.LockoutPeriod = 60
```

EFT v7.0 and later

- Property [LogPath](#) As String

Retrieves or specifies the file path to EFT's log. You must restart the EFT Server service to complete any changes to the log path.

```
HRESULT LogPath([out, retval] BSTR *pVal);
HRESULT LogPath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
logPath = oServer.LogPath
```

Specifying:

```
oServer.LogPath = "C:\Logs\server.log"
```

All versions

- Property [LogRotation](#) As Int

Retrieves or specifies the amount of time before a new log is started. You can choose from no rotation or three different lengths of time. You must restart the EFT Server service to complete any change to the log rotation period.

```
HRESULT LogRotation([out, retval] int *pVal);
HRESULT LogRotation([in] int newVal);
```

Values

- **0** = Never
- **1** = Daily
- **2** = Weekly
- **3** = Monthly

All versions

- Property [LogMode](#) As Int

Retrieves or specifies EFT's log format. You can choose no log, or you can choose to generate a log in one of three formats. You must restart the EFT Server service for any changes to the log format to take effect.

```
HRESULT LogMode([out, retval] LogMode* pVal);
HRESULT LogMode([in] LogMode newVal);
```

Values

2 - W3C Extended Log File Format

3 - Microsoft IIS Log File Format

4 - NCSA Common Log File Format

5 - No Log

All versions

- Property [MaxInactiveAccountPeriod](#) As Long

Retrieves or specifies the period of time after which inactive administrator account will be removed. Measured in days. Limitation: 1 = invalidattemptperiod = 365

```
HRESULT MaxInactiveAccountPeriod([out, retval] LONG *pVal);
HRESULT MaxInactiveAccountPeriod([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Inactive account period" & CStr
(MyServer.MaxInactiveAccountPeriod)
```

Specifying:

```
MyServer.MaxInactiveAccountPeriod = 14 'two weeks
```

EFT v7.0 and later

- Property [MaxInvalidLoginAttempts](#) As Long

Retrieves or specifies the max login attempts for administrator accounts before it will be locked. Limitation: 1 = MaxInvalidLoginAttempts = 99

```
HRESULT MaxInvalidLoginAttempts([out, retval] LONG *pVal);
HRESULT MaxInvalidLoginAttempts([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Max attempts: " & CStr
(MyServer.MaxInvalidLoginAttempts)
```

Specifying:

```
MyServer.MaxInvalidLoginAttempts = 10
```

EFT v7.0 and later

- Function **ModuleRegistrationState**(m As [ServerModule](#)) As [RegistrationState](#)

Retrieves the registration status of EFT modules.

```
HRESULT ModuleRegistrationState(
[in] ServerModule m,
[out, retval] RegistrationState* pRS);
```

Example (VBScript):

The example code illustrates how to correctly use the function and related objects. Using ServerModule with the ModuleRegistrationState function provides the RegistrationState enum value.

```
If Server.ModuleRegistrationState(0)
    0 Then
```

```
MsgBox "EFT is not registered"
End If
```

or

```
If Server.ModuleRegistrationState(0)= 0 Then
    MsgBox "EFT is registered"
End If
```

EFT v6.1 and later

- Sub [MoveIPAccessRule](#)(nRulePosFrom As Long, nRulePosTo As Long)

See also [GetIPAccessRules](#), [RemoveIPAccessRule](#), and [AddIPAccessRule](#)

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Changes allowed remote administration IP access rule position.

```
HRESULT MoveIPAccessRule(
[in] long nRulePosFrom, //initial position
[in] long nRulePosTo); //new position
```

Example (VBScript):

```
oServer.MoveIPAccessRule(1,2)
```

EFT v6.3 and later

- Property [PassPhrase](#) As String

Retrieves or specifies an SSL certificate private key passphrase.

```
HRESULT PassPhrase([out, retval] BSTR *pVal);
HRESULT PassPhrase([in] BSTR newVal);
```

All versions

- Sub [RefreshSettings](#)()

Updates the COM object with the latest set of data from EFT. For example, when you use [Site.ForceSynchronizeUserDatabase](#), you must use the RefreshSettings method after waiting a suitable amount of time for the database synchronization to complete.

If a call to any COM method throws an "MX Error: 52 (0x00000034)", this means that the COM object needs to invoke RefreshSettings.

```
HRESULT RefreshSettings();
```

All versions

- Sub [RemoveIPAccessRule](#) (nRulePos As Long)

See also [GetIPAccessRules](#), [AddIPAccessRule](#), and [MoveIPAccessRule](#), and the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Removes an allowed or denied IP mask for remote administration.

```
HRESULT RemoveIPAccessRule([in] long nRulePos); //position
```

Example (VBScript):

```
oServer.RemoveIPAccessRule(0)
```

EFT v6.3 and later

- Sub [RemovePGPKey](#) (bstrKeyId As String)

Removes OpenPGP Key from EFT.

```
HRESULT RemovePGPKey([in] BSTR bstrKeyId);
```

Example (VBScript):

```
oServer.RemovePGPKey "0x01234567"
```

EFT v6.3 and later

- Sub [RemoveServerAdminAccount](#) (bstrAdminUser As String, [accountType As [AdminAccountType](#) = 0])

Removes an administrator account from EFT.

```
HRESULT RemoveServerAdminAccount([in] BSTR bstrAdminUser,
[in,
        defaultvalue(EFTAccount)] AdminAccountType accountType);
```

Example (VBScript):

Version 6.1:

```
oServer.RemoveServerAdminAccount("subadmin");
```

Version 6.2 and later:

An optional parameter was added to ICIServer::RemoveServerAdminAccount method to allow specifying the type of administrator account to remove (EFT-managed by default)

```
Server.RemoveServerAdminAccount "Admin1", ADAccount
```

EFT 4.3.4 and later

- Sub [RemoveSSHKey](#)(IID As Long)

For an example using this method, refer to the SSH script in [Script Examples](#).

Removes an SSH public key from EFT.

```
HRESULT RemoveSSHKey([in] long lID); // key ID
```

Example (VBScript):

```
For Each key In Server.AvailableSSHKeys 'remove all keys
    Server.RemoveSSHKey key.ID
Next
```

Example (C#):

```
m_server.RemoveSSHKey(key.ID);
```

EFT v6.1 and later

- Sub [RenameSSHKey](#)(IID As Long, bstrNewName As String)

For an example using this method, refer to the SSH script in [Script Examples](#).

Renames an SSH Key.

```
HRESULT RenameSSHKey([in] LONG lID,[in] BSTR bstrNewName);
```

Example (VBScript):

```
For Each key In oServer.AvailableSSHKeys
    If key.Name = "Masha" Then
        oServer.RenameSSHKey key.ID, "Maria">
    End If
Next
```

Example (C#):

```
m_server.RenameSSHKey(keyId, string.Format("{0}_new",
key.Name));
```

EFT v6.3 and later

- Property **RequireStrongPasswords** As Boolean

Retrieves or specifies whether strong, complex passwords are required.

```
HRESULT RequireStrongPasswords([out, retval] VARIANT_BOOL
*pVal);
        HRESULT RequireStrongPasswords([in] VARIANT_BOOL newVal);
```

Example (VBScript)

Retrieving:

```
MsgBox "Complex passwords for admins: " & CStr
(oServer.RequireStrongPasswords)
```

Specifying:

```
oServer.RequireStrongPasswords = False
```

EFT v6.3 and later

- Sub **RestoreServerConfiguration**(bstrFilePath As String, nLoginType As [AdminLoginType](#), bstrLogin As String, bstrPassword As String)

Restores a backed up EFT configuration.

```
HRESULT RestoreServerConfiguration(
[in] BSTR bstrFilePath,
[in] AdminLoginType nLoginType,
[in] BSTR bstrLogin,
[in] BSTR bstrPassword);
```


Retrieve backup information with [GetBackupArchiveInfo](#).

Example (VBScript):

```
oServer.RestoreServerConfiguration "C:\lastbackup.bak",  
    EFTLogin, "admin", "password"
```

EFT v6.3 and later

- Sub **RestoreServerConfigurationEx**(pVal As [CIBackupArchiveInfo](#))

Restores EFT configuration.

```
HRESULT RestoreServerConfigurationEx([in] IDispatch* pVal);
```

Retrieve backup information with [GetBackupArchiveInfo](#).

Example (VBScript):

```
oServer.RestoreServerConfigurationEx oBackupSettings
```

For an example using this method, refer to the Backup script in [Script Examples](#).

EFT v6.3 and later

- Function **SendTestEmail**(bstrRecipientAddress As String) As Boolean

Used to send an email to test SMTP server configuration.

```
HRESULT SendTestEmail(  
    [in] BSTR bstrRecipientAddress,  
    [out, retval] VARIANT_BOOL* pVal);
```

EFT v7.3.6 and later

- Property **ServerInfo** As Object (read only)

This property is an object of type ICIServerInfo for the connected server.

```
HRESULT ServerInfo [out, retval] IDispatch** ppVal);
```

EFT v7.2.1 and later

- Sub **SetComplexPasswordSettings**(plComplexPasswordSettings As [CIComplexPasswordSettings](#))

Changes complex password settings.

```
HRESULT SetComplexPasswordSettings ([in] IDispatch**
                                     pIComplexPasswordSettings);
```

Retrieves complex password settings with [GetComplexPasswordSettings](#).

Example (VBScript):

```
Set oPwdSettings = oServer.GetComplexPasswordSettings()
oPwdSettings.MinPasswordLength = 8
oServer.SetComplexPasswordSettings(oPwdSettings)
```

EFT v6.3 and later

- Sub [SetPGPKeyringSettings](#)(newPublicKeyring As String, newPrivateKeyring As String)

For an example using this method, refer to the PGP script in [Script Examples](#).

Changes OpenPGP Keyring settings.

```
HRESULT SetPGPKeyringSettings(
    [in] BSTR newPublicKeyring,
    [in] BSTR newPrivateKeyring);
```

Retrieve OpenPGP Keyring settings with [GetPGPKeyringSettings](#).

Example (VBScript):

```
oServer.SetPGPKeyringSettings "C:\PGP\pubrign.pgp",
                              "C:\PGP\secreing.pgp"
```

Example (C#):

```
m_server.ImportSSHKey(@"C:\SSHKey.pub");
object publicPath = new object();
object privatePath = new object();
m_server.GetPGPKeyringSettings(out publicPath, out
privatePath);
m_server.SetPGPKeyringSettings
(@"C:\filecryptdata\pubring.pgp",
 @"C:\filecryptdata\secreing.pgp");
```

EFT v6.3 and later

- Sub [SetSSLCertificate](#)(bzCertFilePath As String, bzKeyFilePath As String, bzPassPhrase As String)

Specifies the SSL certificate for serving secure administration connections

```
HRESULT SetSSLCertificate(
    [in] BSTR bzCertFilePath, // Certificate path
    [in] BSTR bzKeyFilePath, // Certificate private key path
    [in] BSTR bzPassPhrase); // Certificate private key password
```

Example (VBScript):

```
Server.SetSSLCertificate "C:\admin.crt", "C:\admin.key",
    "admin_passwd"
```

EFT v6.1 and later

- Property **SFTPEncodingAlgorithms** As String

```
HRESULT SFTPEncodingAlgorithms([out, retval] BSTR *pVal);
HRESULT SFTPEncodingAlgorithms([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SFTPIdentificationComment** As String

```
HRESULT SFTPIdentificationComment([out, retval] BSTR *pVal);
HRESULT SFTPIdentificationComment([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SFTPKexAlgorithms** As String

```
HRESULT SFTPKexAlgorithms([out, retval] BSTR *pVal);
HRESULT SFTPKexAlgorithms([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SFTPMacAlgorithms** As String

```
HRESULT SFTPMacAlgorithms([out, retval] BSTR *pVal);
HRESULT SFTPMacAlgorithms([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SFTPSoftwareVersion** As String

```
HRESULT SFTPSoftwareVersion([out, retval] BSTR *pVal);
HRESULT SFTPSoftwareVersion([in] BSTR newVal);
```

EFT v8.0.5 and later

- Sub **SignSSLCertificate**(bstrRequest As String, bstrServCert As String, bstrPrivateKey As String, bstrPass As String, bstrOutCert As String, nDays As Long)

Signs an SSL certificate (Only for PEM format)

```
HRESULT SignSSLCertificate(
[in] BSTR bstrRequest,
[in] BSTR bstrServCert,
[in] BSTR bstrPrivateKey,
[in] BSTR bstrPass,
[in] BSTR bstrOutCert,
[in] LONG nDays);
```

EFT v6.3 and later

- Function **SignSSLCertificateRemotely** As Long

```
HRESULT SignSSLCertificateRemotely(
[in] BSTR bstrRequest,
[in] BSTR bstrServCert,
[in] BSTR bstrPrivateKey,
[in] BSTR bstrPass,
[in] BSTR bstrOutCert,
[in] LONG nDays);
```

EFT v7.3.6 and later

- Function **Sites()** As [CISites](#)

Retrieves a list of Sites and open the ICISites interface. You can subsequently use the ICISites [Item](#) method to retrieve a pointer to a particular Site. This method returns a pointer to an ICISites interface.

```
HRESULT Sites([out, retval] ICISites** prop);
```

Example (VBScript):

```
Dim Server As CIServer 'assumes the SFTPCOMInterface
Type Library is referenced by the VB IDE
Dim Site As CISite 'instantiates the ISite interface
Dim Sites As CISites 'instantiates the ISites interface
'insert connection routines here
'now retrieve the list of sites and then set a pointer to
the first one
Set Sites = Server.Sites 'now retrieve the list of sites in
the server
```

```
Set Site = Sites.Item(0)
'continue rest of code...
```

Example (C#):

```
CISites sites = m_server.Sites();
```

For an example using this method, refer to the *ServerIPBanEntry* or *RemoveSite* script in [Script Examples](#).

All versions

- Property [SMTPAddressBook](#) As String

Retrieves or specifies a list of email addresses in the SMTP address book.

```
HRESULT SMTPAddressBook([out, retval] BSTR* pVal);
HRESULT SMTPAddressBook([in] BSTR newVal);
```

Example (VBScript):

```
strEmailList = oServer.SMTPAddressBook
If strEmailList "" Then
strEmailList = strEmailList + "; "
End If
strEmailList = strEmailList + "NewRecipientName(SomeAddr
(at)SomeServer.com) "
oServer.SMTPAddressBook = strEmailList
```

EFT v6.3 and later

- Property [SMTPLogin](#) As String

Retrieves or specifies the username for the email server used by EFT for event notifications, if authentication is required.

```
HRESULT SMTPLogin([out, retval] BSTR *pVal);
HRESULT SMTPLogin([in] BSTR newVal);
```

All versions

- Property [SMTPPassword](#) As String

Retrieves or specifies the password for the email server used by EFT for event notifications, if authentication is required.

```
HRESULT SMTPPassword([out, retval] BSTR *pVal);
HRESULT SMTPPassword([in] BSTR newVal);
```

All versions

- Property [SMTPPort](#) As Long

Retrieves or specifies the port number for connection to the SMTP server used by EFT for event notifications.

```
HRESULT SMTPPort([out, retval] long *pVal);
HRESULT SMTPPort([in] long newVal);
```

All versions

- Property [SMTPRecipientAddr](#) As String

Retrieves or specifies the email address of the recipient notifications sent by EFT.

```
HRESULT SMTPRecipientAddr([out, retval] BSTR *pVal);
HRESULT SMTPRecipientAddr([in] BSTR newVal);
```

All versions

- Property [SMTPRecipientName](#) As String

Retrieves or specifies the administrator email name for email notifications sent by EFT.

```
HRESULT SMTPRecipientName([out, retval] BSTR *pVal);
HRESULT SMTPRecipientName([in] BSTR newVal);
```

All versions

- Property [SMTPSenderAddr](#) As String

Retrieves or specifies the administrator email address for email notifications sent by EFT.

```
HRESULT SMTPSenderAddr([out, retval] BSTR *pVal);
HRESULT SMTPSenderAddr([in] BSTR newVal);
```

All versions

- Property [SMTPSenderName](#) As String

Retrieves or specifies the From name used in email notifications sent by EFT.

```
HRESULT SMTPSenderName([out, retval] BSTR *pVal);
HRESULT SMTPSenderName([in] BSTR newVal);
```

All versions

- Property [SMTPServer](#) As String

Retrieves or specifies the address for the email server used by EFT for sending event notifications.

```
HRESULT SMTPServer([out, retval] BSTR *pVal);
HRESULT SMTPServer([in] BSTR newVal);
```

All versions

- Property [SMTPUseAuthentication](#) As Boolean

Retrieves or specifies whether the SMTP server requires authentication.

```
HRESULT SMTPUseAuthentication([out, retval] VARIANT_BOOL*
pVal);
HRESULT SMTPUseAuthentication([in] VARIANT_BOOL newVal);
```

Examples (VBScript):

Retrieving:

```
If oServer.SMTPUseAuthentication Then
' some code
End If
```

Specifying:

```
oServer.SMTPUseAuthentication = False
```

EFT v6.3 and later

- Property [SMTPUseImplicitTLS](#) As Boolean

Retrieves or specifies whether Implicit TLS is specified for the SMTP server.

```
HRESULT SMTPUseImplicitTLS([out, retval] VARIANT_BOOL*
pVal);HRESULT SMTPUseImplicitTLS([in] VARIANT_BOOL newVal);
```

EFT 7.3.6 and later

- Property [SSHFIPEEnabled](#) As Boolean

Retrieves or specifies whether FIPS SSH is enabled. Requires EFT restart.

```
HRESULT SSHFIPSEnabled([out, retval] VARIANT_BOOL* pVal);
HRESULT SSHFIPSEnabled([in] VARIANT_BOOL newVal);
```

EFT v6.3 and later

- Property [SSLFIPSEnabled](#) As Boolean

Retrieves or specifies whether FIPS SSL is enabled. Requires EFT restart.

```
HRESULT SSLFIPSEnabled([out, retval] VARIANT_BOOL* pVal);
HRESULT SSLFIPSEnabled([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
If oServer.SSLFIPSEnabled Then
MsgBox "SSL FIPS enabled"
End If
```

Specifying:

```
oServer.SSLFIPSEnabled = False
```

EFT v6.3 and later

- Property **SSLVersionMask** As Int

Retrieves or specifies the SSL version mask.

```
HRESULT SSLVersionMask([out, retval] int *pVal);
HRESULT SSLVersionMask([in] int newVal);
```

Example (VBScript):

```
Dim iVersionMask: iVersionMask = oServer.SSLVersionMask
WScript.Echo "Version Mask: " & iVersionMask
```

EFT v5.0.1 and later

- Sub **Start()**

Starts the server.

```
HRESULT Start(void);
```


Example (VBScript):

```
oServer.Start()
```

EFT v6.3 and later

- Function [StartServerService](#)(bstrHost As String) As Boolean

Starts EFT Server service.

```
HRESULT StartServerService(
[in] BSTR bstrHost, //empty string means localhost
[out,retval] VARIANT_BOOL* pvResult); // True=OK;
False=error
```

Example (VBScript):

```
If oServer.StartServerService("192.168.0.1")
    Then
MsgBox "EFT Service has started. "
End If
```

Example (C#):

```
m_server.StartServerService("localhost")
```

EFT v6.3 and later

- Property [StartTime](#) As Date

Retrieves the date and time when EFT was started. (read-only)

```
HRESULT StartTime([out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Start date is " & oServer.StartTime
```

For an example using this property, refer to the Backup script in [Script Examples](#).

EFT v6.3 and later

- Sub Stop()

Stops the server.

```
HRESULT Stop(void);
```

Example (VBScript):

```
oServer.Stop()
```

EFT v6.3 and later

- Function [StopServerService](#)(bstrHost As String) As Boolean

Stops EFT Server service. (Applies to EFT Enterprise only.)

```
HRESULT StopServerService(
[in] BSTR bstrHost, //empty string means localhost
[out,retval] VARIANT_BOOL* pvResult); // True=OK;
False=error
```

Example (VBScript):

```
If oServer.StopServerService("192.168.0.1")
Then
MsgBox "EFT Service has stopped."
End If
```

You can also use NET STOP EFT Server Enterprise or NET STOP EFT Server. To restart, you can use:

net stop service name && net start service name

EFT v6.3 and later

- Function **TestContentIntegrityControlProfile**(pProfile As Object) As Boolean

```
HRESULT TestContentIntegrityControlProfile([in] IDispatch*
pProfile, [out, retval] VARIANT_BOOL* pVal);
```

Test the specified Content Integrity Control profile.

EFT v7.0.3 and later

- Sub [UnbanIP](#)(bstrMask As String)

Removes denied IP mask from Autoban list for remote administration.

(Does not remove manually banned IPs. Use [RemoveIPAccessRule](#) instead.)

```
HRESULT UnbanIP([in] BSTR bstrMask); //mask to remove
```

Example (VBScript):

```
oServer.UnbanIP("1.2.3.4")
```

EFT v6.3 and later

- Sub **UpdateContentIntegrityControlProfile(pProfile As Object)**

Update the specified Content Integrity Control profile.

```
HRESULT UpdateContentIntegrityControlProfile([in]
    IDispatch* pProfile);
```

EFT v7.0.3 and later

- Property [Uptime](#) As ULONG

Retrieves uptime of EFT, in seconds. (read-only)

```
HRESULT Uptime([out, retval] ULONG* pVal);
```

Example (VBScript):

```
MsgBox "oServer up time is " & CStr(oServer.Uptime) & "
seconds "
```

Example (C#):

```
seconds = m_server.Uptime;
minutes = seconds / 60; //Calculate minutes
seconds = seconds % 60; //leftover seconds after taking
minutes out
hours = minutes / 60; //Calculate hours
minutes = minutes % 60; //leftover minutes after taking
hours out
days = hours / 24; //Calculate days
hours = hours % 24; //leftover hours after taking days out
Console.WriteLine(string.Format("EFT Uptime:
{0} days {1} hours {2} minutes {3} seconds", days, hours,
minutes, seconds));
```

For an example using this property, refer to the ServerStats script in [Script Examples](#).

EFT v6.3 and later

- Property [UseSSLForAdministration](#) As Boolean

Enables or disables SSL for the administration socket connection. You must restart EFT to apply changes.

```
HRESULT UseSSLForAdministration([out, retval] VARIANT_BOOL
*pVal);HRESULT UseSSLForAdministration([in] VARIANT_BOOL
newVal);
```

All versions

- Property [UseUTCInListings](#) As Boolean

Retrieves or specifies whether to use UTC in directory listings or use the local server time.

```
HRESULT UseUTCInListings([out, retval] VARIANT_BOOL* pVal);
HRESULT UseUTCInListings([in] VARIANT_BOOL newVal);
```

True=UTC; False=local server time.

Example (VBScript):

Retrieving:

```
If oServer.UseUTCInListings Then
' some code here
Else
' another code here
End If
```

Specifying:

```
oServer.UseUTCInListings = False
```

EFT v6.3 and later

- Property **UTF8Logs** As Boolean

Enable or disable logs in UTF-8 format.

```
HRESULT UTF8Logs([out, retval] VARIANT_BOOL *pVal);
HRESULT UTF8Logs([in] VARIANT_BOOL newVal);
```

EFT v8 and later

- Property **WSSeatsUsed** As Long

Number of Workspaces seats in use.

```
HRESULT WSSeatsUsed([out, retval] LONG* pVal);
```

EFT v7.3.3 and later

- Property **WebAdminEnabled** As Boolean

```
HRESULT WebAdminEnabled([out, retval] VARIANT_BOOL *pVal);  
HRESULT WebAdminEnabled([in] VARIANT_BOOL pNewVal);
```

EFT v7.4.6 and later

- Property **WebAdminPort** As Long

```
HRESULT WebAdminPort([out, retval] LONG *pVal);  
HRESULT WebAdminPort([in] LONG pNewVal);
```

EFT v7.4.6 and later

- Property **WSSeatsMax** As Long

Number of Workspaces seats activated.

```
HRESULT WSSeatsMax([out, retval] LONG* pVal);
```

EFT v7.33 and later

- Property **WTCSessionsUsed** As Long

Number of WTC licenses in use.

```
HRESULT WTCSessionsUsed([out, retval] LONG* pVal);
```

EFT v7.3.2 and later

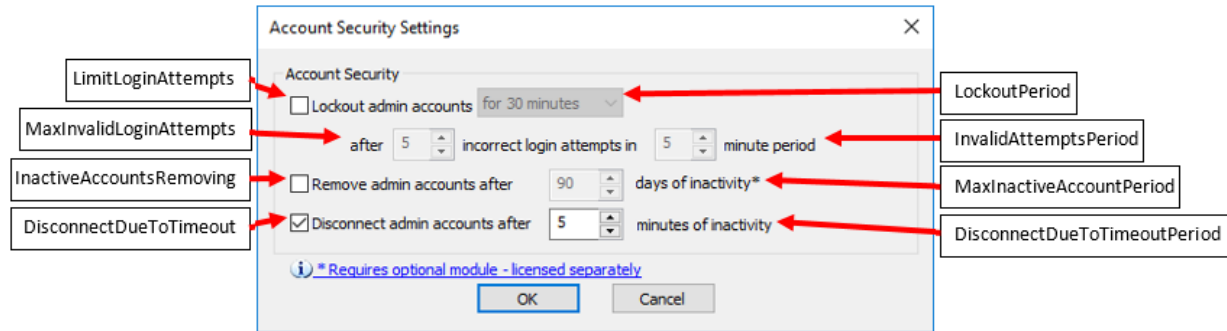
- Property **WTCSessionsMax** As Long

Number of WTC licenses activated.

```
HRESULT WTCSessionsMax([out, retval] LONG* pVal);
```

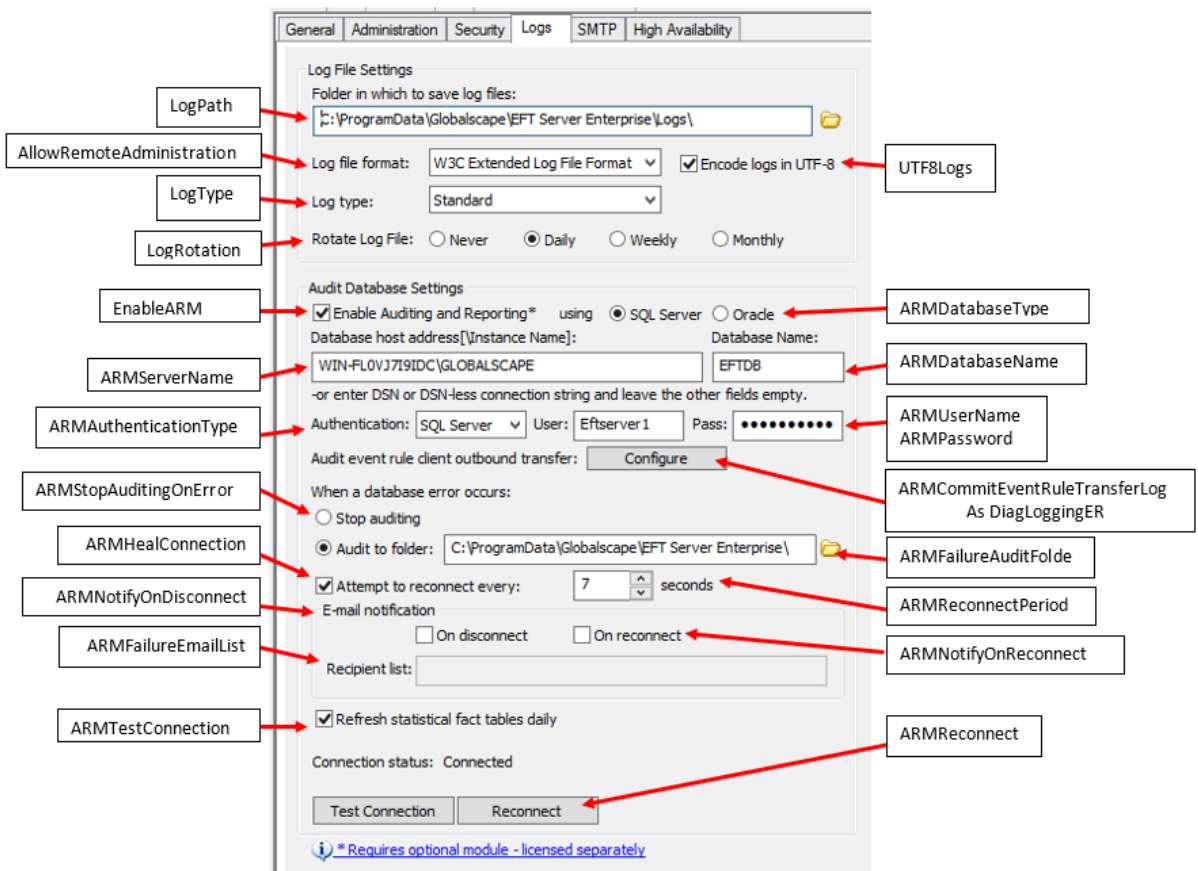
EFT v7.3.2 and later

Administrator Account Security Settings



Server ARM and Log Methods and Properties

The illustration of the Server node's Logs tab below is labeled with some of the methods and properties that are available in the ICIServer interface.



Server General Tab Methods and Properties

The screenshot displays the 'General' tab of the EFT Server configuration interface. On the left, a vertical list of property names is shown, with red arrows pointing to their corresponding values in the 'Statistics' section. On the right, a 'Stop service' button is annotated with a box containing the methods 'StartServerService' and 'StopServerService'. At the bottom right, the 'Directory listing date stamp settings' are shown, with a box explaining the 'UseUTCInListings' property: '1 = UTC' and '0 = localserver time'.

Properties and their values:

- IsServerServiceStarted: Service is started (indicated by a green dot)
- StartTime: Jul 11, 2016. 10:19:13 AM
- Uptime: 0 days, 00 hours, 52 minutes
- LastModifiedTime: Jul 11, 2016. 10:40:09 AM
- LastModifiedBy: Eftserver1
- ConnectedUsersNumber: 0
- ActiveUploads: 0
- ActiveDownloads: 0
- AverageSpeed: 0 bps
- WTCSessionsUsed: 1 used / 5 allowed
- WTCSessionsMax: (empty)
- ConfigurationFilePath: C:\ProgramData\Globalscape\EFT Server Enterprise\
- DefaultPollTimeout: Never refresh user list automatically

Methods:

- StartServerService
- StopServerService

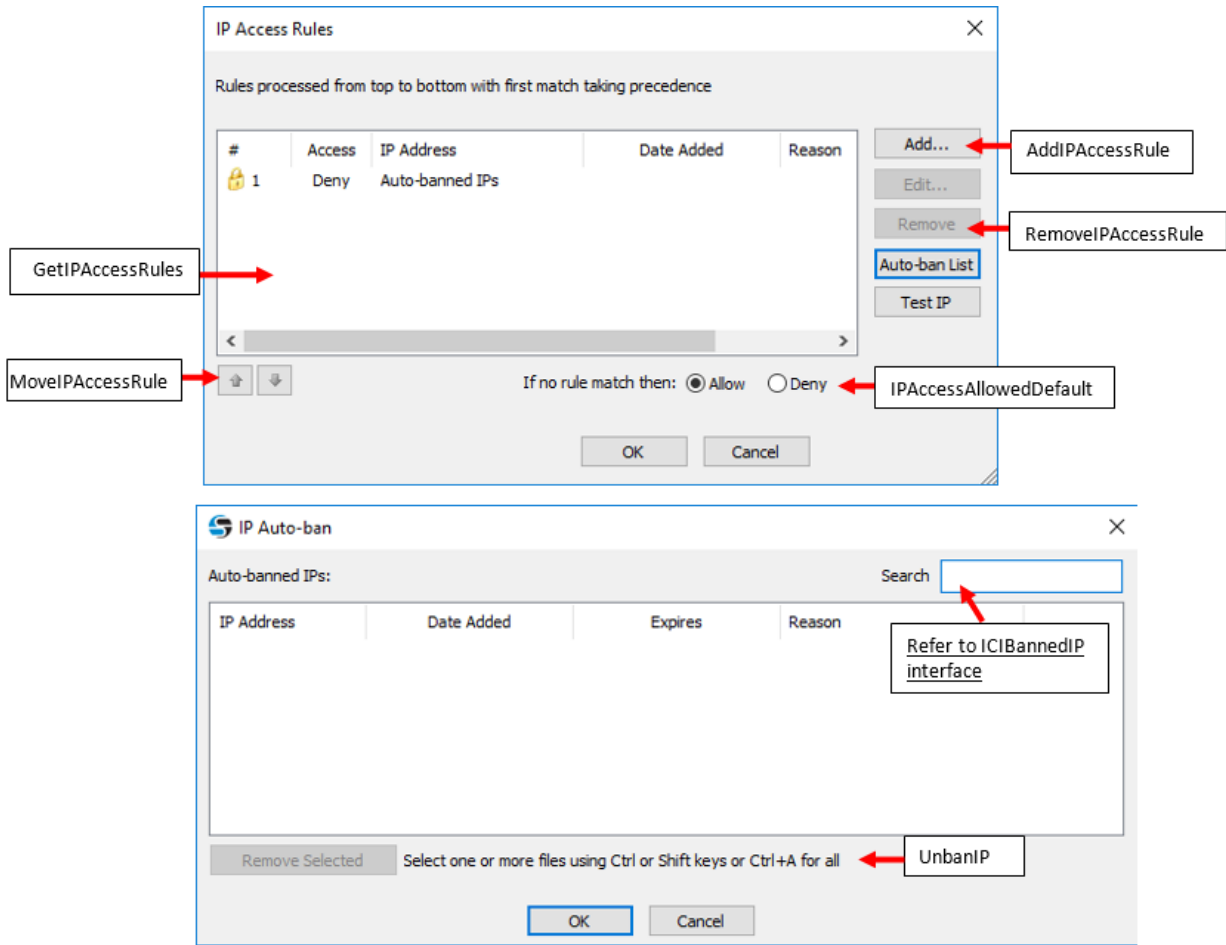
Directory listing date stamp settings:

- Use local server time (radio button)
- Use UTC/GMT time (radio button, selected)

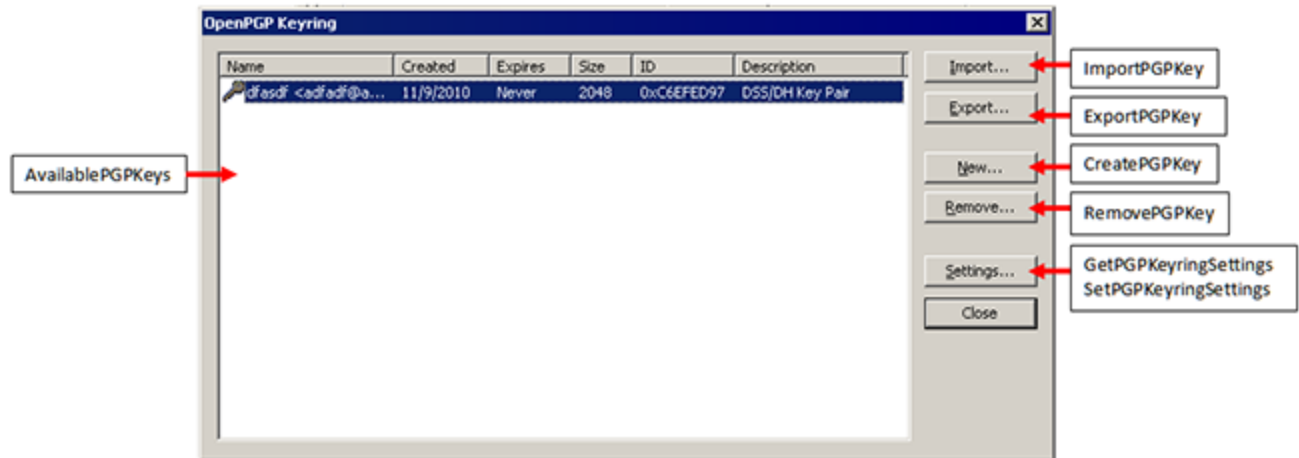
UseUTCInListings:

- 1 = UTC
- 0 = localserver time

Server IP Access Rules



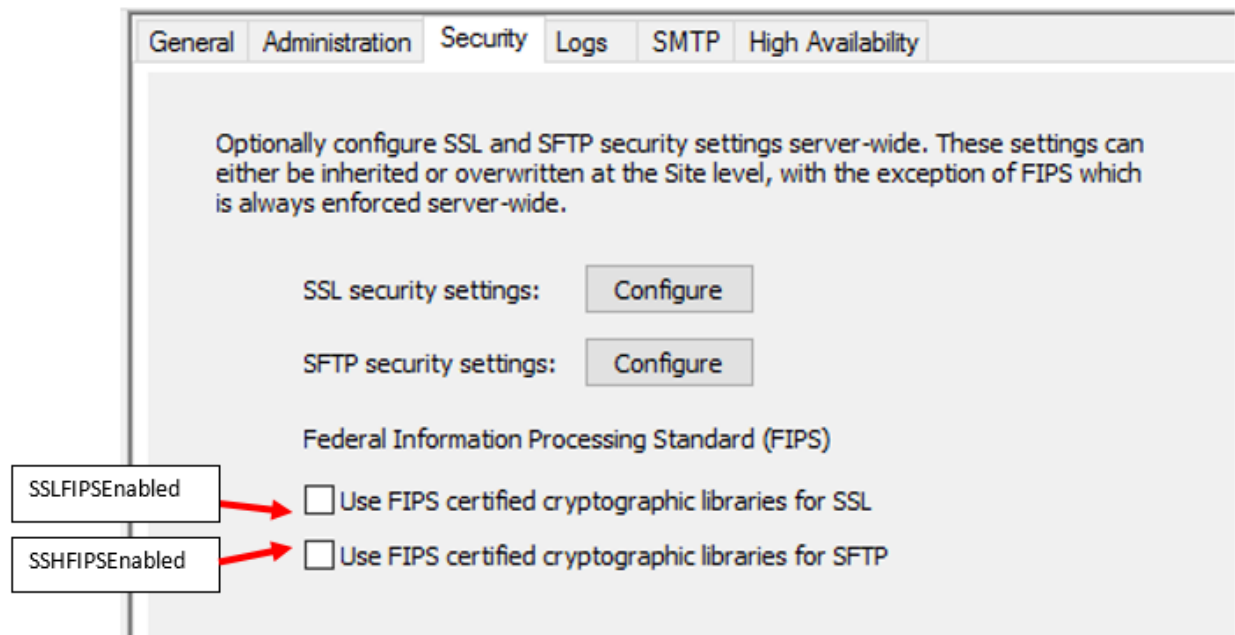
Server OpenPGP Keyring Methods and Properties

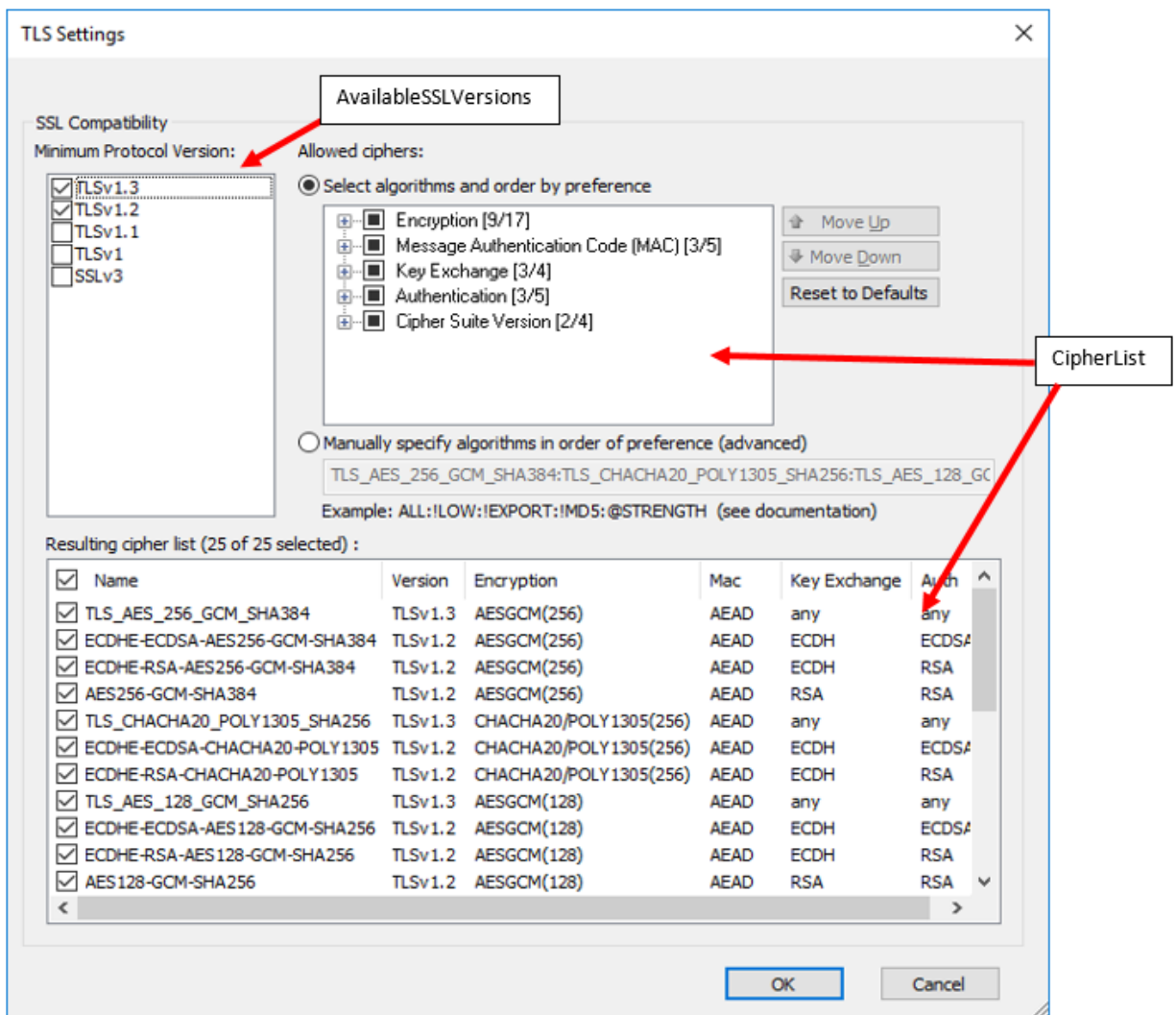


Server Security Methods and Properties

The ICIServer interface allows you to manage the Server, including the Server's security settings. In the administration interface, you can view/edit these settings on the Server's Security tab.

The illustrations of the Server node's Security tab below is labeled with the properties that are available in the ICIServer interface.





SFTPEncodingAlgorithms

SFTPMacAlgorithms

SFTPKexAlgorithms

SSH Settings

Allowed ciphers:

- ☒ aes256-gcm@openssh.com
- ☒ aes256-ctr
- ☒ aes256-cbc
- ☒ rijndael-cbc@lysator.liu.se
- ☒ aes192-ctr
- ☒ aes192-cbc
- ☒ aes128-gcm@openssh.com
- ☒ aes128-ctr
- ☒ aes128-cbc
- ☐ 3des-cbc
- ☐ chacha20-poly1305@openssh.com
- ☐ cast128-cbc
- ☐ blowfish-cbc
- ☐ arcfour

Allowed MACs:

- ☒ hmac-sha2-512-etm@openssh.com
- ☒ hmac-sha2-512
- ☒ hmac-sha2-256-etm@openssh.com
- ☒ hmac-sha2-256
- ☒ hmac-sha1-etm@openssh.com
- ☒ hmac-sha1
- ☐ umac-64-etm@openssh.com
- ☐ umac-64@openssh.com
- ☐ hmac-md5
- ☐ hmac-sha1-96
- ☐ hmac-md5-96

Allowed KEX ciphers:

- ☒ ecdh-sha2-nistp521
- ☒ ecdh-sha2-nistp384
- ☒ ecdh-sha2-nistp256
- ☐ diffie-hellman-group18-sha512
- ☒ diffie-hellman-group16-sha512
- ☒ diffie-hellman-group14-sha256
- ☒ diffie-hellman-group-exchange-sha256
- ☐ diffie-hellman-group14-sha1
- ☐ curve25519-sha256
- ☐ curve25519-sha256@libssh.org
- ☐ sntrup4591761x25519-sha512@tinyssh.org
- ☐ diffie-hellman-group-exchange-sha1
- ☐ diffie-hellman-group1-sha1

Optionally specify:

protocol:
SSH-2.0-

software version:
8.1.0.0_openssh

comments:
Globalscape

SFTPSoftwareVersion

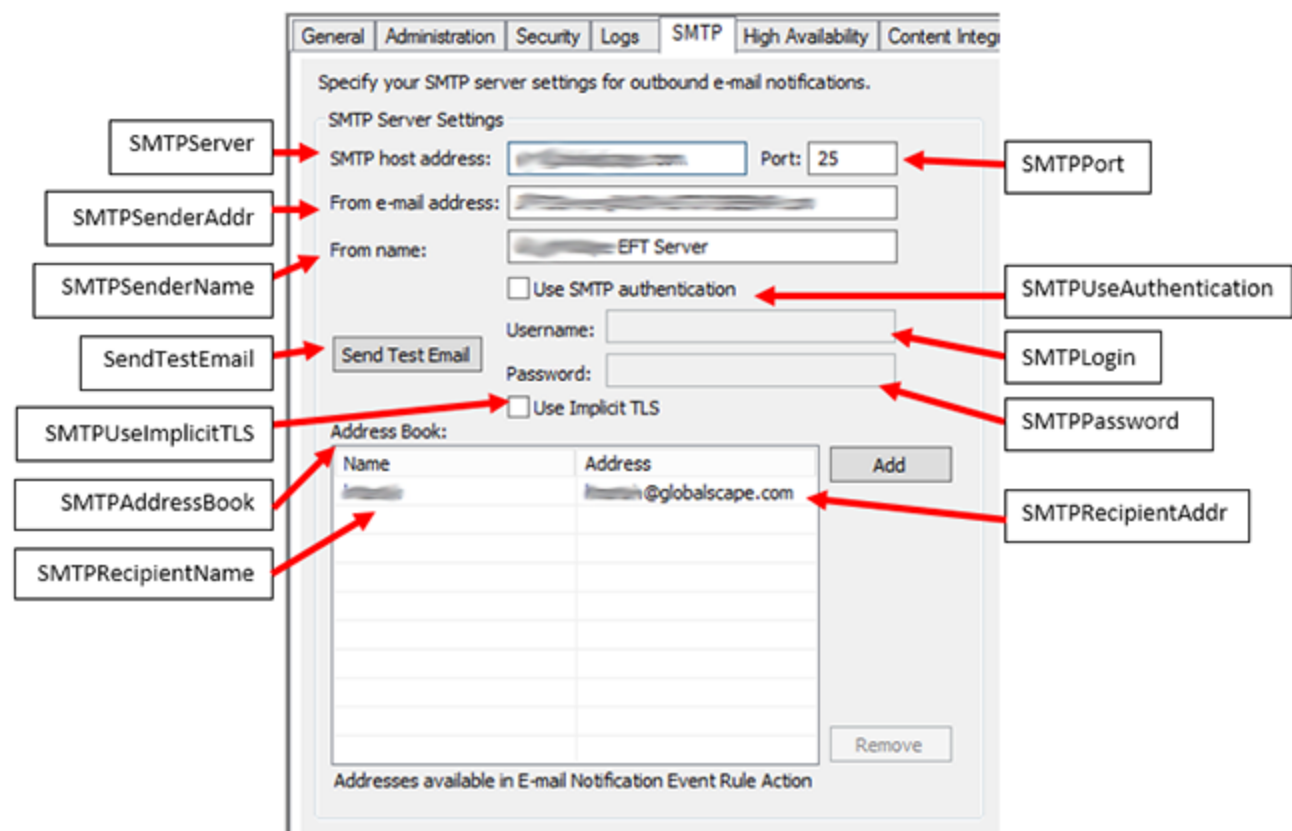
SFTPIdentificationComment

OK

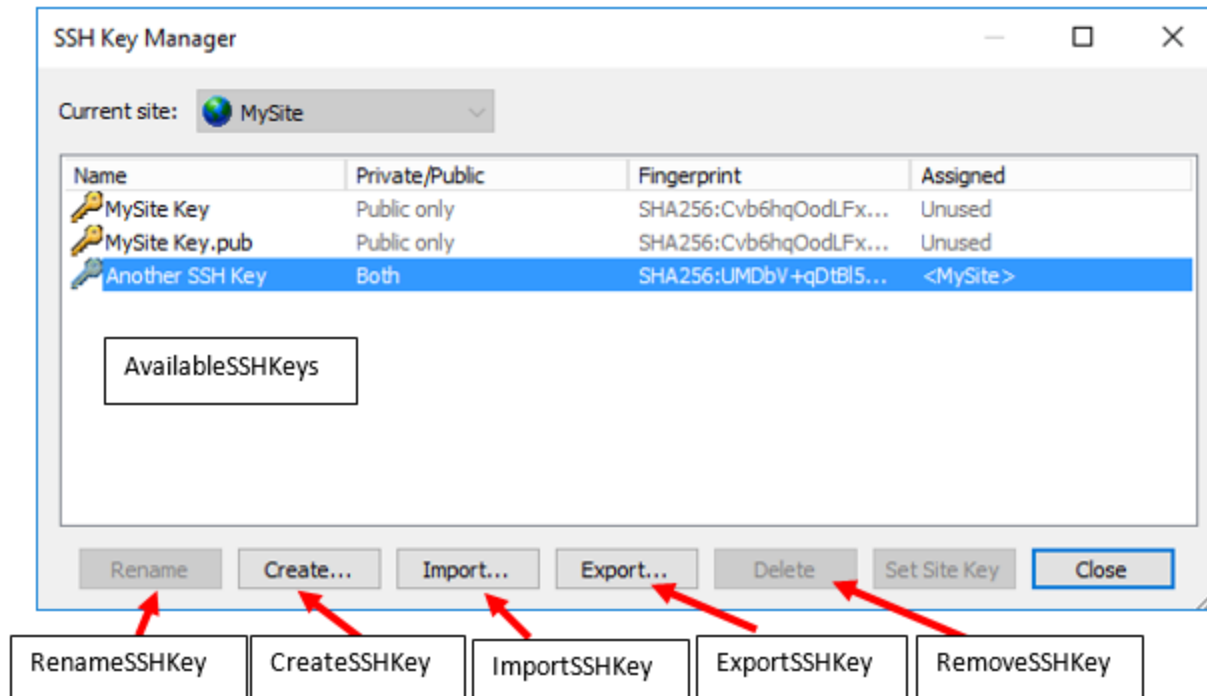
Cancel

Server SMTP Methods and Properties

The ICIServer interface allows you to manage the Server, including the Server's SMTP settings. The illustrations of the Server node's SMTP tab below is labeled with some of the methods and properties that are available in the ICIServer interface.

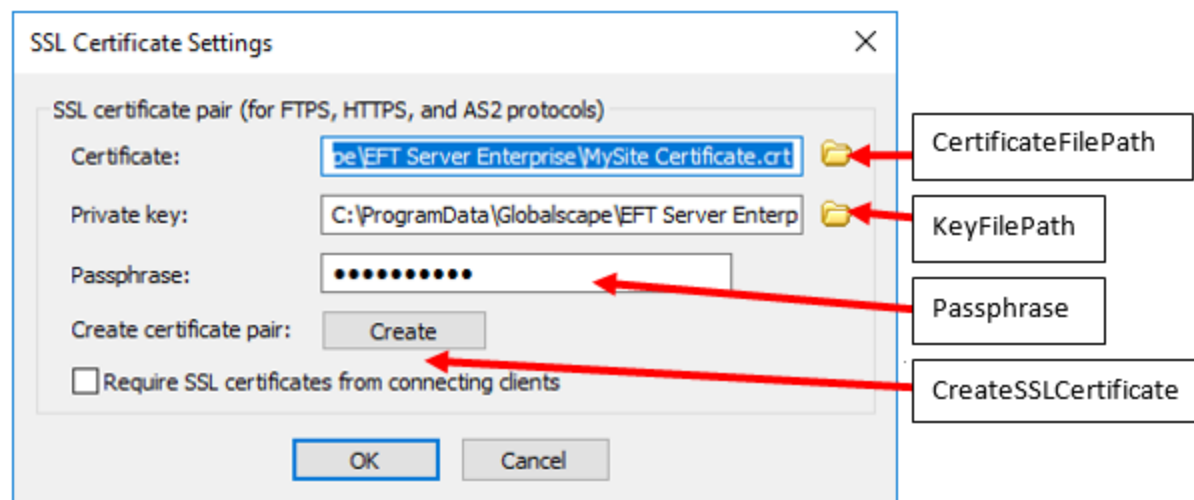


Server SSH Key Manager Methods and Properties



Server SSL Certificate Settings

The ICIServer interface allows you to manage the Server's SSL Certificate for remote administration connections. In the administration interface, you can access the SSL certificate settings on the Administration tab by clicking Configure next to the Require SSL for remote administration check box.



ICIServerInfo Interface

The ICIServerInfo interface is used to specify and view node configuration.

Members (listed alphabetically)

- Property **NodeCount** As Long

```
HRESULT NodeCount ([out, retval] LONG *pVal);
```

EFT v7.2.1 and later

- Function **NodeInfo**(index As Long) As Object

This function returns an object of type [ICIServerNodeInfo](#) for the server of the given index

```
HRESULT NodeInfo([in] LONG index, [out, retval] IDispatch** pVal);
```

EFT v7.2.1 and later

ICIServerNodeInfo

The ICIServerNodeInfo interface is used to specify and view node configuration.

Members (Listed alphabetically)

- Property **ActiveClientDownloadBytesPerSecond** As Long

```
HRESULT ActiveClientDownloadBytesPerSecond ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **ActiveClientDownloadCount** As Long

```
HRESULT ActiveClientDownloadCount ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **ActiveClientUploadBytesPerSecond** As Long

```
HRESULT ActiveClientUploadBytesPerSecond ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **ActiveClientUploadCount** As Long

```
HRESULT ActiveClientUploadCount ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerDownloadBytesPerSecond** As Long

```
HRESULT ActiveServerDownloadBytesPerSecond ([out, retval]  
Long *pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerDownloadCount** As Long

```
HRESULT ActiveServerDownloadCount ([out, retval] Long  
*pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerUploadBytesPerSecond** As Long

```
HRESULT ActiveServerUploadBytesPerSecond ([out, retval] Long  
*pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerUploadCount** As Long

```
HRESULT ActiveServerUploadCount ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **ActiveWTCSessions** As Long

```
HRESULT ActiveWTCSessions([out, retval] LONG* plVal);
```

EFT 7.3

- Property **AdminLastModified** As String

```
HRESULT AdminLastModified ([out, retval] BSTR *pVal);
```

EFT 7.2.1 and later

- Property **ConnectedAdminCount** As Long

```
HRESULT ConnectedAdminCount ([out, retval] LONG *pVal);
```

- Property **IsEventRuleBalancingMaster** As Boolean


```
HRESULT IsEventRuleBalancingMaster ([out, retval] VARIANT_BOOL *pVal);
```

EFT 7.2.1 and later

- Property **IsInCluster** As Boolean

```
HRESULT IsInCluster ([out, retval] VARIANT_BOOL *pVal);
```

EFT 7.2.1 and later

- Property **IsStarted** As Boolean

```
HRESULT IsStarted ([out, retval] VARIANT_BOOL *pVal);
```

EFT 7.2.1 and later

- Property **LastAdminLoginTime** As DATE

```
HRESULT LastAdminLoginTime ([out, retval] DATE *pVal);
```

EFT 7.2.1 and later

- Property **LastModificationTime** As DATE

```
HRESULT LastModificationTime ([out, retval] DATE *pVal);
```

EFT 7.2.1 and later

- Property **ListenIPs** As Variant

```
HRESULT ListenIPs([out, retval] VARIANT* pvarVal);
```

EFT 7.3 and later

- Function **ModuleLicenseCount**(module As [ServerModule](#)) As Long

```
HRESULT ModuleLicenseCount([in] ServerModule module, [out, retval] LONG *pVal);
```

EFT 7.2.1 - 7.3.1 (See [TotalModuleLicenseCount](#))

- Function **ModuleRegistrationState**(module As [ServerModule](#)) As RegistrationState

```
HRESULT ModuleRegistrationState([in] ServerModule module, [out, retval] RegistrationState *pVal);
```

EFT 7.2.1 and later

- Property **Name** As String

```
HRESULT Name ([out, retval] BSTR *pVal);
```

EFT 7.2.1 and later

- Property **RunningAWTaskCount** As Long

```
HRESULT RunningAWTaskCount ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **RunningEventRuleCount** As Long

```
HRESULT RunningEventRuleCount ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Property **SiteCount** As Long

```
HRESULT SiteCount ([out, retval] Long *pVal);
```

EFT 7.2.1 and later

- Function **SiteInfo**(index As Long) As Object

This function returns an object of type [ICIServerNodeSiteInfo](#) for the site of the given index.

```
HRESULT SiteInfo([in] LONG index, [out, retval] IDispatch** pVal);
```

EFT 7.2.1 and later

- Property **StartTime** As DATE

```
HRESULT StartTime ([out, retval] DATE *pVal);
```

EFT 7.2.1 and later

- Function **TotalModuleLicenseCount**(module As [ServerModule](#)) As Long

```
HRESULT TotalModuleLicenseCount([in] ServerModule module, [out, retval] LONG *pVal);
```

EFT 7.3.2 and later

- Property **UsedModuleLicenseCount**(m As [ServerModule](#)) As Long

```
HRESULT UsedModuleLicenseCount([in] ServerModule m, [out,
retval] LONG* pValue);
```

EFT 7.3 and later

- Property **Version As String**

```
HRESULT Version ([out, retval] BSTR *pVal);
```

EFT 7.2.1 and later

ICIServerNodeSiteInfo

The ICIServerNodeSiteInfo interface provides the properties specific to a Site node.

Members (Listed alphabetically)

- Property **ActiveClientDownloadBytesPerSecond As Long**

```
HRESULT ActiveClientDownloadBytesPerSecond ([out, retval]
LONG *pVal);
```

EFT 7.2.1 and later

- Property **ActiveClientDownloadCount As Long**

```
HRESULT ActiveClientDownloadCount ([out, retval] LONG
*pVal);
```

EFT 7.2.1 and later

- Property **ActiveClientUploadBytesPerSecond As Long**

```
HRESULT ActiveClientUploadBytesPerSecond ([out, retval] LONG
*pVal);
```

EFT 7.2.1 and later

- Property **ActiveClientUploadCount As Long**

```
HRESULT ActiveClientUploadCount ([out, retval] LONG *pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerDownloadBytesPerSecond As Long**

```
HRESULT ActiveServerDownloadBytesPerSecond ([out, retval]
LONG *pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerDownloadCount** As Long

```
HRESULT ActiveServerDownloadCount ([out, retval] long
*pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerUploadBytesPerSecond** As Long

```
HRESULT ActiveServerUploadBytesPerSecond ([out, retval] LONG
*pVal);
```

EFT 7.2.1 and later

- Property **ActiveServerUploadCount** As Long

```
HRESULT ActiveServerUploadCount ([out, retval] LONG *pVal);
```

EFT 7.2.1 and later

- Property **AdminLastModified** As String

```
HRESULT AdminLastModified ([out, retval] BSTR *pVal);
```

EFT 7.2.1 and later

- Property **ConnectedUserCount** As Long

```
HRESULT ConnectedUserCount ([out, retval] LONG *pVal);
```

EFT 7.2.1 and later

- Property **GatewayConnected** As Boolean

```
HRESULT GatewayConnected([out, retval] VARIANT_BOOL* pVal);
```

EFT 7.3

- Property **GatewayIP** As String

```
HRESULT GatewayIP([out, retval] BSTR* pVal);
```

EFT 7.3

- Property **GatewayPort** As Long

```
HRESULT GatewayPort([out, retval] LONG* pVal);
```

EFT 7.2.3

- Property **HostName** As String

```
HRESULT HostName ([out, retval] BSTR *pVal);
```

EFT 7.2.1 and later

- Property **LastModificationTime** As DATE

```
HRESULT LastModificationTime ([out, retval] DATE *pVal);
```

EFT 7.2.1 and later

- Property **LastUserLoginTime** As DATE

```
HRESULT LastUserLoginTime ([out, retval] DATE *pVal);
```

EFT 7.2.1 and later

- Property **ListenIPs** As Variant

```
HRESULT ListenIPs ([out, retval] VARIANT *pVal);
```

EFT 7.2.1 and later

- Property **Name** As String

```
HRESULT Name ([out, retval] BSTR *pVal);
```

EFT 7.2.1 and later

- Property **RunningAWTaskCount** As Long

```
HRESULT RunningAWTaskCount ([out, retval] LONG *pVal);
```

EFT 7.2.1 and later

- Property **RunningEventRuleCount** As Long

```
HRESULT RunningEventRuleCount ([out, retval] LONG *pVal);
```

EFT 7.2.1 and later

- Property **StartTime** As DATE

```
HRESULT StartTime ([out, retval] DATE *pVal);
```

EFT 7.2.1 and later

- Property **State** As [ServerNodeSiteState](#)

```
HRESULT State ([out, retval] ServerNodeSiteState *pVal);
```

EFT 7.2.1 and later

ICIBackupArchiveInfo Interface

Use the ICIBackupArchiveInfo interface members to retrieve or specify backup information. For details of backing up or restoring EFT Server, refer to "Backing Up or Restoring Server Configuration" in the EFT Server documentation.

```
interface ICIBackupArchiveInfo : IDispatch
```

NOTE: "Sub" indicates that the member is a method; "Property" indicates that it is a property.

Members

- Property **ARMAAuthenticationType** As [ARMAAuthenticationType](#)

Retrieves or specifies audit database authentication type – SQL server authentication or Windows authentication.

```
HRESULT ARMAAuthenticationType([out, retval]
ARMAAuthenticationType* pVal);
HRESULT ARMAAuthenticationType([in] ARMAAuthenticationType
newVal);
```

Example (VBScript):

Retrieving:

```
If WindowsAuthentication = oBackup.ARMAAuthenticationType
Then
' ...
End If
```

Specifying:

```
oBackup.ARMAAuthenticationType = SQLServerAuthentication
```

EFT v6.3 and later

- Property **ARMDatabaseName** As String

Retrieves or specifies ARM database name.

```
HRESULT ARMDatabaseName([out, retval] BSTR *pVal);
HRESULT ARMDatabaseName([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "ARM database name is: " & oBackup.ARMDatabaseName
```

Specifying:

```
oBackup.ARMDatabaseName = "EFTDB"
```

EFT v6.3 and later

- Property **ARMDatabaseType** As [ARMDatabaseType](#)

Retrieves or specifies audit database type.

```
HRESULT ARMDatabaseType([out, retval] ARMDatabaseType*
pVal);
HRESULT ARMDatabaseType([in] ARMDatabaseType newVal);
```

Example (VBScript):**Retrieving:**

```
If Oracle = oBackup.ARMDatabaseType Then
'...
End If
```

Specifying:

```
oBackup.ARMDatabaseType = SQLServer
```

EFT v6.3 and later

- Property **ARMFailureAuditFolder** As String

Retrieves or specifies folder to save auditing data when the database has an error.

```
HRESULT ARMFailureAuditFolder([out, retval] BSTR* pVal);
HRESULT ARMFailureAuditFolder([in] BSTR newVal)
```

Example (VBScript):**Retrieving:**

```
Auditfolder = oBackup.ARMFailureAuditFolder
```

Specifying:

```
oBackup.ARMFailureAuditFolder = "C:\some_folder"
```

EFT v6.3 and later

- Property **ARMFailureEmailList** As String

Retrieves or specifies a failure notification list of emails. The property is a string with the following format:

```
email ::= recipient_name<addr@server.domain> |
addr@server.domain
email_list ::= email; email_list | e
HRESULT ARMFailureEmailList([out, retval] BSTR* pVal);
HRESULT ARMFailureEmailList([in] BSTR newVal);
```

Example (VBScript):

```
strEmailList = oBackup.ARMFailureEmailList
If strEmailList <> "" Then
strEmailList = strEmailList + "; "
End If
strEmailList = strEmailList +
"NewRecipientName<SomeAddr@SomeMailServer.com>"
oBackup.ARMFailureEmailList = strEmailList
```

EFT v6.3 and later

- Property **ARMHealConnection** As Boolean

Enables or disables the ARM connection healing mechanism.

```
HRESULT ARMHealConnection([out, retval] VARIANT_BOOL *pVal);
HRESULT ARMHealConnection([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
If MyBackupArchiveInfo.ARMHealConnection Then
'Healing connection mechanism is enabled
...
End If
```


Specifying:

```
MyBackupArchiveInfo.ARMHealConnection = False
'Healing connection mechanism have disabled
```

EFT v7 and later

- Property **ARMNotifyOnDisconnect** As Boolean

Enables or disables email notification about ARM database disconnect (recipients defined in [ARMFailureEmailList](#)).

```
HRESULT ARMNotifyOnDisconnect([out, retval] VARIANT_BOOL
*pVal);
HRESULT ARMNotifyOnDisconnect([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
If MyBackupArchiveInfo.ARMNotifyOnDisconnect Then
'Notify on ARM database disconnect is enabled
...
End If
```

Specifying:

```
MyBackupArchiveInfo.ARMNotifyOnDisconnect = False
'Notify on ARM database disconnect have disabled
```

EFT v7and later

- Property **ARMNotifyOnReconnect** As Boolean

Enables or disables email notification about ARM database reconnect (recipients defined in [ARMFailureEmailList](#)).

```
HRESULT ARMNotifyOnReconnect([out, retval] VARIANT_BOOL
*pVal);
HRESULT ARMNotifyOnReconnect([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
If MyBackupArchiveInfo.ARMNotifyOnReconnect Then
  'Notify on ARM database Reconnect is enabled
...
End If
```

Specifying:

```
MyBackupArchiveInfo.ARMNotifyOnReconnect = False
'Notify on ARM database reconnect have disabled
```

EFT v7 and later

- Property **ARMPassword As String**

Retrieves or specifies ARM database password.

```
HRESULT ARMPassword([out, retval] BSTR *pVal);
HRESULT ARMPassword([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "ARM database password is: " & oBackup.ARMPassword
```

Specifying:

```
oBackup.ARMPassword = "swordfish"
```

EFT v6.3 and later

- Property **ARMReconnectPeriod As Long**

Retrieve or specify the reconnect period to ARM database. If a database error occurs and [ARMHealConnection](#) is enabled, then EFT tries to reconnect to the database every **ARMReconnectPeriod** seconds, from 7 seconds to 86,400 seconds (24 hours).

```
HRESULT ARMReconnectPeriod([out, retval] LONG *pVal);
HRESULT ARMReconnectPeriod([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Reconnect period: " & CStr
(MyBackupArchiveInfo.ARMReconnectPeriod)
```

Specifying:

```
MyBackupArchiveInfo.ARMReconnectPeriod = 60 'every one
minute
```

EFT v7 and later

- Property **ARMServerName** As String

Retrieves or specifies database host address.

```
HRESULT ARMServerName([out, retval] BSTR *pVal);
HRESULT ARMServerName([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "ARM database host address is: " &
oBackup.ARMServerName
```

Specifying:

```
oBackup.ARMServerName = "192.168.0.55"
```

EFT v6.3 and later

- Property **ARMStopAuditingOnError** As Boolean

Retrieves or specifies server behavior in case of database error. If ARMStopAuditingOnError is true, then after database error, audit will stop. Otherwise, auditing will be continued in the folder specified by the [ARMFailureAuditFolder](#) property.

```
HRESULT ARMStopAuditingOnError([out, retval] VARIANT_BOOL*
pVal);
HRESULT ARMStopAuditingOnError([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
If oBackup.ARMStopAuditingOnError Then
'...
End If
```

Specifying:

```
oBackup.ARMStopAuditingOnError = true
```

EFT v6.3 and later

- Property **ARMUserName** As String

Retrieves or specifies database user name.

```
HRESULT ARMUserName([out, retval] BSTR *pVal);
HRESULT ARMUserName([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "ARM database username is: " & oBackup.ARMUserName
```

Specifying:

```
oBackup.ARMUserName = "User_1"
```

EFT v6.3 and later

- Property **BackupArchivePath** As String read-only

Retrieves path to the backup archive file.

```
HRESULT BackupArchivePath([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "File backup archive path: " &
oBackup.BackupArchivePath
```

EFT v6.3 and later

- Property **CreatedBy** As String read-only

Retrieves administrator account name that created the backup.

```
HRESULT CreatedBy([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Backup created by: " & oBackup.CreatedBy
```

EFT v6.3 and later

- Property **CreationTime** As Date read-only

Retrieves backup creation time.

```
HRESULT CreationTime([out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Backup created: " & oBackup.CreationTime
```

EFT v6.3 and later

- Property **EnableARM** As Boolean

Retrieves or specifies enable/disable ARM.

```
HRESULT EnableARM([out, retval] VARIANT_BOOL *pVal);
HRESULT EnableARM([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "ARM is enabled: " & CStr(oBackup.EnableARM)
```

Specifying:

```
oBackup.EnableARM = True
```

EFT v6.3 and later

- Function **GetSiteBackupInfo**(nIndex As Long) As Object

Retrieves [ICISiteBackupInfo](#) interface by index.

```
HRESULT GetSiteBackupInfo([in] LONG nIndex, [out, retval]
IDispatch** pVal);
```

Example (VBScript):

```
Set oSiteBackupInfo = oBackup.GetSiteBackupInfo(0)
```

EFT v6.3 and later

- Sub **SetSiteBackupInfo**(newVal As Object)

Specifies Site backup information.

```
HRESULT SetSiteBackupInfo([in] IDispatch* newVal);
```

Example (VBScript):

```
oBackup.SetSiteBackupInfo oSiteBackupInfo
```

EFT v6.3 and later

- Property **SitesCount** As Long read-only
Retrieves number of Sites in the backup archive.

```
HRESULT SitesCount([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Sites count: " & CStr(oBackup.SitesCount)
```

EFT v6.3 and later

- Property **type** As String read-only
Retrieves string representation of backup type.

```
HRESULT Type([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Backup type: " & oBackup.Type
```

EFT v6.3 and later

ICIAdminAccount Interface

The ICIAdminAccount interface is used to view, add, or remove permissions on administrator accounts. See also [ICIAdminPermission - Administrator Permission Interface](#).

```
interface ICIAdminAccount : IDispatch
```

NOTE: "Function" and "Sub" indicate that the member is a method; "Property" indicates that it is a property.

Members (Listed alphabetically)

- Sub **AddPermission**(plAdminPermission As Object)
Adds permission to an administrator account.

```
AddPermission(  
[in] IDispatch* pIAdminPermission);
```

Example (VBScript):

```
oAdmin.AddPermission oCOMPerm
```

EFT v6.3 and later

- Function **GetPermission**(nIndex As DWord) As Object

For an example using this method, refer to the **RetrieveAdminUser** script in [Script Examples](#).

See also **GetPermissionsCount**, below.

Retrieves the permission on an administrator account.

```
HRESULT GetPermission(
[in] ULONG nIndex,
[out, retval] IDispatch** pIAdminPermission);
```

Example (VBScript):

```
Set oAdminPerm = oAdmin.GetPermission(0)
```

Example (C#):

```
CIAdminPermission permission =
(CIAdminPermission)admin.GetPermission((uint)i);
```

EFT v6.3 and later

- Function **GetPermissionsCount**() As DWord

For an example using this method, refer to the **RetrieveAdminUser** script in [Script Examples](#).

See also **GetPermission**, above.

Retrieves the number of permissions on an administrator account.

```
GetPermissionsCount(
[out, retval] ULONG* pVal);
```

Example (VBScript):

```
nPermCount = oAdmin.GetPermissionsCount()
```

Example (C#):

```
for (int i = 0; i < admin.GetPermissionsCount(); i++)
```

EFT v6.3 and later

- Property **Login As String**, read-only

For an example using this property, refer to the **RetrieveAdminUser** script in [Script Examples](#).

Retrieves the username of the account.

```
HRESULT Login(
    [out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Admin login: " & oAdmin.Login
```

Example (C#):

Retrieving (admin info):

```
Console.WriteLine(string.Format("Admin user  
    [{0}] has the following permissions:", admin.Login));
```

Specifying (admin account):

```
ICIAdminAccount adminUser = (ICIAdminAccount)m_
server.CreateAdmin("Test_Administrator",
    "Tester!1", AdminAccountType.EFTAccount, false);
Console.WriteLine(string.Format("Admin login  
    {0} created.", adminUser.Login));
```

EFT v6.3 and later

- Sub **RemoveAllPermissions()**

Removes all administrator permissions.

```
RemoveAllPermissions();
```

Example (VBScript):

```
oAdmin.RemoveAllPermissions()
```

EFT v6.3 and later

- Sub **RemovePermission(plAdminPermission As Object)**

Removes an administrator permission.


```
RemovePermission(
[in] IDispatch* pIAdminPermission);
```

Example (VBScript):

```
oAdmin.RemovePermission oCOMPerm
```

EFT v6.3 and later

- Sub **RemovePermissionByIndex**(nIndex As DWord)

Removes a permission by index.

```
RemovePermissionByIndex(
[in] ULONG nIndex);
```

Example (VBScript):

```
oAdmin.RemovePermissionByIndex 0
```

EFT v6.3 and later

- Property **type** As [AdminAccountType](#) read-only

Retrieves whether the account is an EFT-managed account, local user account, or an AD account.

```
HRESULT Type(
[out, retval] AdminAccountType* pVal);
```

Example (VBScript):

```
Const EFTAccount = 0
Const LocalComputerAccount = 1
Const ADAccount = 2
if EFTAccount = oAdmin.Type then
MsgBox "It is GS admin"
end if
```

EFT v6.3 and later

ICIAdminID

The AdminID interface is used in the [ICIServer - Server Interface::CreateAdminID](#) method. All properties are read-only.

Members (Listed alphabetically)

- Property **IsGroup** As Boolean

```
HRESULT IsGroup([out, retval] VARIANT_BOOL* pVal);
```

EFT v7.2 and later

- Property **Login** As String

```
HRESULT Login([out, retval] BSTR* pVal);
```

EFT v7.2 and later

- Property type As [AdminAccountType](#)

```
HRESULT Type([out, retval] AdminAccountType* pVal);
```

EFT v7.2 and later

ICIAdminPermission Interface

The ICIAdminPermission interface is used to retrieve access and permission information about an administrator account. See also [ICIAdminAccount - Administrator Account Interface](#).

```
interface ICIAdminPermission : IDispatch
```

Properties

- Property **Permission** As [AdminPermissionsPolicy](#)

Retrieves or specifies the permission on an administrator account.

```
HRESULT Permission(  
[out,retval]AdminPermissionsPolicy* pVal);  
HRESULT Permission(AdminPermissionsPolicy newVal);
```

Example (VBScript):

```
if ServerManagement = oAdminPerm.Permission then  
MsgBox "It is server admin."  
end if
```

EFT v6.3 and later

- Property **SiteName** As String

Retrieves the Site names for SiteManagement, STManagement, UserCreation and ChangePassword permissions.

```
HRESULT SiteName(
    [out,retval]BSTR* pVal);
HRESULT SiteName(BSTR newVal);
```

Example (VBScript):

Retrieving:

```
strSiteName = oAdminPerm.SiteName
```

Specifying:

```
oAdminPerm.SiteName = "MySite"
```

EFT v6.3 and later

- Property **TemplateName** As String

Retrieve/specifies the Settings Template name for STManagement, UserCreation, and ChangePassword permissions.

```
HRESULT TemplateName(
    [out,retval]BSTR* pVal);
HRESULT TemplateName(BSTR newVal);
```

Example (VBScript):

Retrieving:

```
strSTName = oAdminPerm.TemplateName
```

Specifying:

```
oAdminPerm.TemplateName = "My Settings Template"
```

EFT v6.3 and later

ICIReportInfo Interface

ICIReportInfo interface is used to retrieve information about EFT's reports. (Also refer to [ICIServer::AvailableReports property](#)). The interface provides the set of read-only properties. The Name and DefinitionPath properties together allow you to identify the report among other reports. (Note that one definition file can contain definitions of multiple

reports and the same report name can be used for different reports in different definition files.)

```
interface ICIReportInfo : IDispatch
```

Example (VBScript):

```
MsgBox "Report: " & Report.Name & "(" & Report.DefinitionPath &
") "
```

Properties

- Property **DefinitionPath** As String read-only
Retrieves the path to the report definition file.

```
HRESULT DefinitionPath([out, retval] BSTR* pVal);
```

EFT Server v6.1 and later

ICISLVersion

The ICISLVersion interface contains read-only properties used to specify SSL version information.

Members (Listed alphabetically)

- Property **BitFlag** As Int read-only

```
HRESULT BitFlag([out, retval] int *pVal);
```

The bit flag value associated with the SSL version

EFT v7.1.1

- Property **Name** As String read-only

```
HRESULT Name([out, retval] BSTR *pVal);
```

The name of the SSL protocol version (i.e. SSLv3)

EFT v7.1.1

ICISLVersions

The ICISLVersions interface contains read-only properties used to discover the number or index of SSL versions available.

- Function **Count()** As Long

```
HRESULT Item([in] long nIndex, [out, retval] ICISLVersion**  
prop);
```

Returns the number of SSL protocol versions defined.

EFT v7.1.1

- Function **Item**(nIndex As Long) As [CISLVersion](#)

```
HRESULT Count([out, retval] long* prop);
```

Returns an SSL protocol version item by index.

EFT v7.1.1

Site-Level Interfaces

This chapter contains Site-related COM objects.

ICISites Interface

Use the ICISites interface to create new Sites or to retrieve information about the Sites on the Server. (See also [ICISite](#) for details about an individual Site.)

```
interface ICISites : IDispatch
```

Members (Listed alphabetically)

- Function **Add**(bstrName As String, bstrRootFolder As String, nAMID As Int, bstrAMDB As String, bEncryptPasswords As Boolean, nIP As Int, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [ICISite](#)

See also [AddEx](#) and [AddSite](#).

Creates a new Site.

```
HRESULT Add(
    [in] BSTR bstrName,
    [in] BSTR bstrRootFolder,
    [in] int nAMID,
    [in] BSTR bstrAMDB,
    [in] VARIANT_BOOL bEncryptPasswords,
    [in] int nIP,
    [in] long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);
```

bstrName - New site name

bstrRootFolder - Site root folder, as a full path for example, "c:\inetpub\ftproot\new site"

nAMID - Authentication manager identifier; this must be a number between 0 and 3 (inclusive), as follows:

- 0 : Globalscape EFT Authentication
- 1 : NTLM/Active Directory Authentication
- 2 : ODBC Authentication
- 3 : LDAP Authentication

bstrAMDB - Authentication manager database configuration string. This is specific to the Authentication Manager chosen by the nAMID parameter.

- NT: "NTLM=1;(DC=<<domain controller name>>)(GROUP=<<group name>>);"
- AD: ";(DC=<<domain controller name>>)(GROUP=<<group name>>);"
- ODBC: "<<dsn connection string>>"
- LOCAL: "<<path to database file>>"
- LDAP: "LDAPDB=<<path to database file>>;LDAPSERVER=<<host>>;LDAPPORT=<<port>>;BASEDN=<<base dn for users>>;USERFILTER=<<user query filter>>;USERATTRIBUTE=<<logon attribute>>;SSL={0|1};(USERNAME=<<ldap simple bind username>>;PASSWORD=<<ldap simple bind password>>);"

bEncryptPasswords - TRUE = database should contain encrypted passwords. This applies only to ODBC Authentication sites, and determines if password in the database are to be considered one-way hashed passwords or stored as plain-text in the columns (useful if you have external processes manipulating rows in the database).

nIP - Index of the site IP address. A "0" value means "all incoming." A number greater than 0 refers to the index in the IP Address list for all addresses on the computer.

nPort - Port number of the site

bCreateHomeFolder - TRUE means that a home folder for new site users should be created

bPermHomeFolder - TRUE means that new site users should be granted full permissions in their home folders

bAutoStart - TRUE means that the site should start automatically.

All versions

- Function **AddADSite**(bstrName As String, bstrRootFolder As String, bstrDomainContext As String, bstrAllowGroup As String, nIP As Int, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

See [ICIADAuthManagerSettings Interface](#) for AD-specific settings.

See also [AddADSiteEx](#), below.

Creates an AD-authenticated Site.

```
HRESULT AddADSite(
    [in] BSTR bstrName,
    [in] BSTR bstrRootFolder,
    [in] BSTR bstrDomainContext,
    [in] BSTR bstrAllowGroup,
    [in] int nIP, long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);
```

bstrName - New Site name, e.g., "AD Site"

bstrRootFolder - Site root folder, e.g., "C:\sites\AD"

bstrDomainName - Default domain context for Active Directory, e.g., "" (required, but can be null)

bstrAllowGroup - Group name, e.g., "" (required, but can be null)

nIP - Index of the Site IP address, e.g., 0

nPort - Port number of the Site, e.g., 21

bCreateHomeFolder - TRUE means that a home folder for new Site users should be created

bPermHomeFolder - TRUE means that new Site users should be granted full permissions in their home folders

bAutoStart - TRUE means that the Site should start automatically

bCreateFTPFolders - TRUE means to create standard FTP folders (Usr, Bin, Incoming, Pub) (v6.1 and later.)

Example (VBScript):

```
Set Site = Sites.AddADSite("AD Site", "C:\sites\AD", "", "",
    0, 21, True, True, True, True)
```

All versions

- Function **AddADSiteEx**(bstrName As String, bstrRootFolder As String, bstrDomainContext As String, bstrAllowGroup As String, IP, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

Creates a new AD site, extended version, for multiple listening IPs.

```
HRESULT AddADSiteEx(
    [in] BSTR bstrName,
    [in] BSTR bstrRootFolder,
    [in] BSTR bstrDomainContext,
    [in] BSTR bstrAllowGroup,
    [in] VARIANT IP, long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);
```

EFT v6.4 and later

- Function **AddEx** (bzName As String, bzRootFolder As String, pdispAuthManagerSettings As Object, lIP As Int, lPort As Long, bCreateHomeFolder As Boolean, bFullPermissionForHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

Creates a new Site.

```
HRESULT AddEx(
    [in] BSTR bzName,
    [in] BSTR bzRootFolder,
    [in] IDispatch* pdispAuthManagerSettings,
    [in] int lIP,
    [in] long lPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bFullPermissionForHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out,retval] ICISite** pNewSite);
```

EFT v6.3 and later

- Function **AddLDAPSite**(bstrName As String, bstrRootFolder As String, bstrDomainContext As String, nIP As Int, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

See [ICLDAPAuthManagerSettings - LDAP Authentication Manager Settings Interface](#) for LDAP-specific settings.

See also [AddLDAPSiteEx](#), below.

Creates an LDAP-authenticated Site.

```
HRESULT AddLDAPSite(
    [in] BSTR bstrName,
    [in] BSTR bstrRootFolder,
    [in] BSTR bstrDomainContext,
    [in] int nIP, long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);
```

bstrName - New site name

bstrRootFolder - Site root folder

bstrDomainContext - A semicolon-separated list of name/value pairs that defines the LDAP domain components. (See **Configuration Notes** below for details of how to compose this string.)

bstrAllowGroup - LDAP group name (not used in v6.1)

nIP - Index of the site IP address

nPort - Port number of the site

bCreateHomeFolder - TRUE means that a home folder for new site users should be created

bPermHomeFolder - TRUE means that new site users should be granted full permissions in their home folders

bAutoStart - TRUE means that the site should start automatically

bCreateFTPFolders - TRUE means to create standard FTP folders (Usr, Bin, Incoming, Pub) (v6.1 and later)

You can also call **Add()**, setting "3" as the nAMID parameter to create an LDAP site.

Configuration Notes

The **bstrDomainContext** string must be composed properly to define the parameters of the LDAP Site. The syntax is a semicolon-separated list of name/value pairs that defines the LDAP options.

The entire string is:

```
LDAPSERVER={IP or Hostname for LDAP server};LDAPPORT={port on LDAP server};BASEDN={base dn for users};USERFILTER={filter for LDAP query to return users};USERATTRIBUTE={what user attribute to use as client login};TYPE={1 for anonymous bind, 0 for simple bind};USERNAME={cn of the user to bind as for querying users};PASSWORD={password for that user};LDAPDB={path to database file};SSL={0 for no SSL, 1 for SSL};USESEARCHSCOPE={1 to set search scope, 0 otherwise};SEARCHSCOPE={search scope: 0 => base, 1 => one level, => subtree};USESEARCHSIZELIMIT={1 to override search page size, 0 otherwise};SEARCHSIZELIMIT={new search page size};USETIMEOUT={1 to use timeout, 0 otherwise};TIMEOUT={timeout in seconds};USEVERSION={1 to use specific version, otherwise 0};VERSION={LDAP version (2 or 3)};USEPAGECONTROL={1 to use LDAP server side page control, 0 otherwise};USEREFERRALS={1 to turn on referral changing; 0 otherwise};USESEARCHATTRIBUTES={1 to select attributes; 0 otherwise };SEARCHATTRIBUTES={search attributes};
```

You do not have to include the entire string. Any options from **USESEARCHSCOPE** to the end can be omitted and EFT will use the default values.

EFT ignores the **TYPE** attribute; it is only used here to set the radio button on the **LDAP Authentication Options** dialog box.

Example (VBScript):

(type all on one line without spaces; shown with line breaks below for readability):

```
Set Site = Sites.AddLDAPSite("LDAP Site", "C:\sites\LDAP",  
"", 0, 21, True, True, True, True)  
LDAPSERVER=fileserv  
LDAPPOR=389;  
BASEDN=dc=s5development,c=local;  
USERFILTER=(objectClass=person);
```

```

USERATTRIBUTE=cn;
TYPE=0;
USERNAME=cn=Ivan Smithe,cn=users,dc=development,dc=local;
PASSWORD=XXX;
LDAPDDB=C:\EFT AppData\MySite.aud;SSL=0;
USESEARCHSCOPE=1;
SEARCHSCOPE=2;
USESEARCHSIZELIMIT=1;
SEARCHSIZELIMIT=1000;
USETIMEOUT=1;
TIMEOUT=60;
USEVERSION=1;
VERSION=3;
USEPAGECONTROL=1;
USEREFERRALS=0;
USESEARCHATTRIBUTES=1;
SEARCHATTRIBUTES=mail,e-mail,name,cn;

```

EFT v4.3.4 and later

- Function **AddLDAPSiteEx**(bstrName As String, bstrRootFolder As String, bstrDomainContext As String, IP, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

Creates a new AD Site, extended version for multiple listening IP addresses.

```

HRESULT AddLDAPSiteEx(
    [in] BSTR bstrName, [in] BSTR bstrRootFolder,
    [in] BSTR bstrDomainContext,
    [in] VARIANT IP, long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);

```

EFT v6.4 and later

- Function **AddLocalSite**(bstrName As String, bstrRootFolder As String, bstrFilePath As String, nIP As Int, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

For an example using this method, refer to the CreateSite script in [Script Examples](#).

See also [AddLocalSiteEx](#), below.

Creates a new unauthenticated Site and retrieve the Site's ICISite interface.

```
HRESULT AddLocalSite(
    [in] BSTR bstrName,
    [in] BSTR bstrRootFolder,
    [in] BSTR bstrFilePath,
    [in] int nIP,
    [in] long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);
```

bstrName - New site name

bstrRootFolder - Site root folder

bstrFilePath - Path to the file that will contain user data

nIP - Index of the Site IP address

nPort - Port number of the Site

bCreateHomeFolder - TRUE = home folder for new Site users should be created

bPermHomeFolder - TRUE = new Site users should be granted full permissions in their home folders

bAutoStart - TRUE = Site should start automatically

bCreateFTPFolders - TRUE = create standard FTP folders (Usr, Bin, Incoming, Pub) (v6.1 and later)

Example (VBScript):

```
Set Site = Sites.AddLocalSite("Local Site",
    "C:\sites\Local", _
    "C:\sites\Local.aud", 0, 21, True, True, True, True)
```

Example (C#):

```
CISite newSite = m_server.Sites().AddLocalSite("MySite",
    @"C:\Inetpub\EFTRoot\MySite",
    @"C:\Documents and Settings\All Users\Application
```

```
Data\GlobalSCAPE\EFT Enterprise\MySite.aud",
0, 21, true, true, true, true);
```

All versions

- Function **AddLocalSiteEx**(bstrName As String, bstrRootFolder As String, bstrFilePath As String, IP, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As CISite

Creates a new GS Site, extended version for multiple listening IP addresses.

```
HRESULT AddLocalSiteEx(
[in] BSTR bstrName,
[in] BSTR bstrRootFolder,
[in] BSTR bstrFilePath,
[in] VARIANT IP, [in] long nPort,
[in] VARIANT_BOOL bCreateHomeFolder,
[in] VARIANT_BOOL bPermHomeFolder,
[in] VARIANT_BOOL bAutoStart,
[in] VARIANT_BOOL bCreateFTPFolders,
[out, retval] ICISite** prop);
```

EFT v6.4 and later

- Function **AddNTLMSite**(bstrName As String, bstrRootFolder As String, bstrDomainName As String, bstrAllowGroup As String, nIP As Int, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

See [ICIADAuthManagerSettings - Active Directory Authentication Manager Settings Interface](#) for Windows-specific options.

See also [AddNTLMSiteEx](#), below.

Creates an NTLM-authenticated Site.

```
HRESULT AddNTLMSite(
[in] BSTR bstrName,
[in] BSTR bstrRootFolder,
[in] BSTR bstrDomainName,
[in] BSTR bstrAllowGroup,
[in] int nIP,
[in] long nPort,
[in] VARIANT_BOOL bCreateHomeFolder,
[in] VARIANT_BOOL bPermHomeFolder,
```

```
[in] VARIANT_BOOL bAutoStart,
[in] VARIANT_BOOL bCreateFTPFolders,
[out, retval] ICISite** prop);
```

bstrName - New site name

bstrRootFolder - Site root folder

bstrDomainName - NT domain name

bstrAllowGroup - NT group name

nIP - Index of the site IP address

nPort - Port number of the site

bCreateHomeFolder - TRUE = home folder for new site users should be created

bPermHomeFolder - TRUE = new site users should be granted full permissions in their home folders

bAutoStart - TRUE = site should start automatically

bCreateFTPFolders - TRUE = create standard FTP folders (Usr, Bin, Incoming, Pub) (v6.1 and later)

Example (VBScript):

```
Set Site = Sites.AddNTLMSite("NTLM Site", "C:\sites\NTLM",
"", 0, 21, True, True, True, True)
```

All versions

- Function **AddNTLMSiteEx**(bstrName As String, bstrRootFolder As String, bstrDomainName As String, bstrAllowGroup As String, IP, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

Creates a new NT Site, extended version for multiple listening IP addresses.

```
HRESULT AddNTLMSiteEx(
[in] BSTR bstrName,
[in] BSTR bstrRootFolder,
[in] BSTR bstrDomainName,
[in] BSTR bstrAllowGroup,
[in] VARIANT IP,
[in] long nPort,
[in] VARIANT_BOOL bCreateHomeFolder,
```

```
[in] VARIANT_BOOL bPermHomeFolder,
[in] VARIANT_BOOL bAutoStart,
[in] VARIANT_BOOL bCreateFTPFolders,
[out, retval] ICISite** prop);
```

EFT v6.4 and later

- Function **AddODBCSite**(bstrName As String, bstrRootFolder As String, bstrDSN As String, bEncryptPasswords As Boolean, nIP As Int, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

See [ICIOCBCAuthManagerSettings - ODBC Authentication Manager Settings Interface](#) for ODBC-specific settings.

See also [AddODBCSiteEx](#), below.

Creates an ODBC-authenticated Site.

```
HRESULT AddODBCSite(
[in] BSTR bstrName,
[in] BSTR bstrRootFolder,
[in] BSTR bstrDSN,
[in] VARIANT_BOOL bEncryptPasswords,
[in] int nIP,
[in] long nPort,
[in] VARIANT_BOOL bCreateHomeFolder,
[in] VARIANT_BOOL bPermHomeFolder,
[in] VARIANT_BOOL bAutoStart,
[in] VARIANT_BOOL bCreateFTPFolders,}
[out, retval] ICISite** prop);
```

bstrName - New site name

bstrRootFolder - Site root folder

bstrDSN - Database DSN

bEncryptPasswords - TRUE = database should contain encrypted passwords

nIP - Index of the site IP address

nPort - Port number of the site

bCreateHomeFolder - TRUE = home folder for new site users should be created

bPermHomeFolder - TRUE= new site users should be granted full permissions in their home folders

bAutoStart - TRUE = site should start automatically

bCreateFTPFolders - TRUE = create standard FTP folders (Usr, Bin, Incoming, Pub) (v6.1 and later)

Example (VBScript):

```
Set Site = Sites.AddODBCSite("ODBC Site",
"C:\sites\ODBC", "myDSN;", True, 0, 21, True, True, True,
True)
```

All versions

- Function **AddODBCSiteEx**(bstrName As String, bstrRootFolder As String, bstrDSN As String, bEncryptPasswords As Boolean, IP, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

Creates a new ODBC Site, extended version for multiple listening IP addresses.

```
HRESULT AddODBCSiteEx(
[in] BSTR bstrName,
[in] BSTR bstrRootFolder,
[in] BSTR bstrDSN,
[in] VARIANT_BOOL bEncryptPasswords,
[in] VARIANT IP,
[in] long nPort,
[in] VARIANT_BOOL bCreateHomeFolder,
[in] VARIANT_BOOL bPermHomeFolder,
[in] VARIANT_BOOL bAutoStart,
[in] VARIANT_BOOL bCreateFTPFolders,
[out, retval] ICISite** prop);
```

EFT v6.4 and later

- Function **AddSite**(bstrName As String, bstrRootFolder As String, nAMID As Int, bstrAMDB As String, bEncryptPasswords As Boolean, IP, nPort As Long, bCreateHomeFolder As Boolean, bPermHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As [CISite](#)

Creates a new site.

```

HRESULT AddSite(
    [in] BSTR bstrName,
    [in] BSTR bstrRootFolder,
    [in] int nAMID,
    [in] BSTR bstrAMDB,
    [in] VARIANT_BOOL bEncryptPasswords,
    [in] VARIANT IP,
    [in] long nPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bPermHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out, retval] ICISite** prop);

```

EFT v6.4 and later

- Function **AddSiteEx**(bzName As String, bzRootFolder As String, pdispAuthManagerSettings As Object, IP, IPort As Long, bCreateHomeFolder As Boolean, bFullPermissionForHomeFolder As Boolean, bAutoStart As Boolean, bCreateFTPFolders As Boolean) As CISite

Creates a new Site, extended version for multiple listening IP addresses.

```

HRESULT AddSiteEx(
    [in] BSTR bzName,
    [in] BSTR bzRootFolder,
    [in] IDispatch* pdispAuthManagerSettings,
    [in] VARIANT IP,
    [in] long lPort,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bFullPermissionForHomeFolder,
    [in] VARIANT_BOOL bAutoStart,
    [in] VARIANT_BOOL bCreateFTPFolders,
    [out,retval] ICISite** pNewSite);

```

EFT v6.4 and later

- Function **Count()** As Long

Returns the number of Sites defined.

```

HRESULT Count([out, retval] long* prop);

```

All versions

- Function **Item**(nIndex As Long) As [CISite](#)

For an example using this method, refer to the CreateSite or RemoveSite script in [Script Examples](#).

Returns an item by index. The **Index** value must be between 0 and (Count-1).

```
HRESULT Item([in] long nIndex,[out, retval] ICISite** prop);
```

Example (VBScript):

```
Set Sites = Server.Sites 'Retrieve list of sites
Set Site = Sites.Item(0) 'Instantiate the first one
```

Example (C#):

```
CISite site = sites.Item(i);
```

All versions

- Function **SiteByID**(ID As Long) As [CISite](#)

Retrieves the properties of a Site by ID.

```
HRESULT SiteByID(
[in] long ID,
[out, retval] ICISite** prop);
```

All versions

ICISite Interface

The ICISite interface represents an individual Site. (Refer to [ICISites - Multi-Site Management](#) to view/make changes to all of the Sites at once or to create a new Site.)

For example: Set Site = Sites.Item(0)

```
interface ICISite : IDispatch
```

Jump to: [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#)

Scroll down for [dialog box equivalents](#).

For SFTPAdvBool, used below:

- abInherited = -2 (&HFFFFFFFE)
- abFalse = 0
- abTrue = 1

For PwdType, used below:

- 0 = Regular (hashed)
- 1 = Anonymous (regular with empty password)(removed in v7.3.3.21 and later)
- 2 = Anonymous (require email as password) (removed in v7.3.3.21 and later)
- 3 = OTP-MD4 (removed in v7.3.3.21 and later)
- 4 = OTP-MD5
- 5 = Literal (no hash) - For use when importing shadow passwords from a *nix password file. The format must follow: \$1\$*\$* For example, user: test, password: test would be: site.ChangeUserPassword "test", "\$1\$Bn3YdWKv\$.BS3Qb2UQTV4SmV1JNN.w/", 5
- 6 = nPwdType - For importing hashes of unknown type from another source. (When the user logs into EFT, EFT would cycle through possible hashes for the plaintext password to see which one matches the supplied hash. When a match is found, the user is able to log in. As well, EFT will rehash the password with its current default hashing algorithm and store that as the user's hashed password.)

NOTE: DES-format passwords are not supported. These are password formats without the salt defined, that is, without the \$1\$.

Members (Listed alphabetically)

- Property **AccountManagementPath** As String (read only)

In the EFT administration interface, this property is enabled on the Site's Connection tab.

Retrieves the account management URL at which users can reset their passwords via HTTPS. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT AccountManagementPath ([out, retval] BSTR*
    pVal);
```

Example (VBScript):

```
MsgBox "Account management over https path is:
      " & CStr(oSite.AccountManagementPath)
```

EFT v6.3 and later

- Function **AddAdvancedWorkflow**(pdisplParams As Object) As Long
Adds new Advanced Workflow to this Site.

```
HRESULT AddAdvancedWorkflow([in] IDispatch* pdispParams,
    // AW params // as ICIAdvancedWorkflowParams
    *l[out, retval] long* plIndex); // zero-based index of
created AW
```

Example (VBScript):

```
set AWParams = CreateObject
("SFTPCOMInterface.CIAdvancedWorkflowParams")
'...setup workflow
site.AddAdvancedWorkflow(AWParams)
```

EFT v6.5 and later

- Function **AddCertificateToTrusted**(IID As Long) As Long
Moves a pending certificate to trusted.

```
HRESULT AddCertificateToTrusted([in] long IID,
    [out, retval] long *lNewID);
```

Example (VBScript):

```
'Loop through pending certificates
'and assign certificate to a client
'if the certificate common name
'coincides with client's full name
For Each pID in Site.GetPendingCertificates
If Site.GetPendingCertificateInfo(pID).SubjectCName =
C.FullName Then
tID = Site.AddCertificateToTrusted(pID)
C.SetSSLAuthenticationType "key"
C.SetSSLKeyID tID
End If
Next
```

All versions

- Function **AddConnectionProfile**(pdispParams As Object) As Long
See also [RemoveConnectionProfile](#).

```
HRESULT AddConnectionProfile(
    [in] IDispatch* pdispParams,
    [out, retval] LONG* plIndex);
```

EFT v7.2 and later

- Method **AddContentIntegrityControlProfile**

Add CIC profile:

```
HRESULT AddContentIntegrityControlProfile([in] IDispatch*
pProfile);
```

EFT v8

- Sub **AddEncryptedFolder**(pIEncryptedFolder As Object)

```
HRESULT AddEncryptedFolder([in] IDispatch*
pIEncryptedFolder);
```

EFT v7.4.13 and later

- Sub [AddIPAccessRule](#)

Adds allowed or denied IP mask for the Site. (default value and Reason added in v8)

```
HRESULT AddIPAccessRule(
[in] BSTR bstrMask,
[in] VARIANT_BOOL bAllow,
[in] long nRulePos,
[in, defaultvalue("")] BSTR bstrReason);
```

Example (VBScript):

```
oSite.AddIPAccessRule("1.2.3.4", false, 2)
```

All versions

See also the following interfaces:

- [ICIAutobanIPAccessRule Interface](#)
- [ICIBannedIP Interface](#)
- [ICIIPAccessRule Interface](#)
- [ICIManualIPAccessRule Interface](#)
- Sub **AddUploadForm**(pUploadForm As Object)

Used to add a new Upload Form to the Site.

```
HRESULT AddUploadForm([in] IDispatch* pUploadForm);
```

EFT v8.0.4 and later

- Sub [AddUserToPermissionGroup](#)(bstrUser As String, bstrGroup As String)

Adds a user to a permission group.

```
HRESULT AddUserToPermissionGroup(
    [in] BSTR bstrUser,
    [in] BSTR bstrGroup);
```

All versions

- Property **AdvancedWorkflowsCount** As Long

Retrieves the total count of site's Advanced Workflows.

```
HRESULT AdvancedWorkflowsCount([out, retval] long* plValue);
```

Example (VBScript):

```
If Site.AdvancedWorkflowsCount = 0 Then
    'Site has no Advanced Workflows
...
End If
```

EFT v6.5 and later

- Property [AllowChangePassword](#) As Boolean

Allows you to retrieve or specify whether Site users are allowed to change their passwords. The value can be overridden at the Settings Template or user account level.

```
HRESULT AllowChangePassword([out, retval] VARIANT_BOOL
    *pVal);
HRESULT AllowChangePassword([in] VARIANT_BOOL val);
```

Examples (VBScript):

Retrieving:

```
If Site.AllowChangePassword Then
    Site.ForcePasswordResetOnInitialLogin = True
End If
```

Specifying:

```
If Site.AllowChangePassword = False
```

EFT v6.2 and later

- Property **AllowClearCommandChannel** As Boolean

Used to enable or disable clear command channel.

```
HRESULT AllowClearCommandChannel([in] VARIANT_BOOL newVal);
HRESULT AllowClearCommandChannel([out, retval] VARIANT_BOOL
*pVal);
```

EFT v8.0.5 and later

- Property [AllowEmailLoginCredentials](#) As Boolean

Retrieves or specifies whether allow login credentials to be sent in emails. If AllowEmailLoginCredentials is enabled, [EmailLoginCredentialsMode](#) retrieves how they are sent (username and password in the same email, separate emails, or username only).

```
HRESULT AllowEmailLoginCredentials([out, retval] VARIANT_
BOOL* pVal);
HRESULT AllowEmailLoginCredentials([in] VARIANT_BOOL
newVal);
```

TRUE=Email login credentials; FALSE=Do not email login credentials

Example (VBScript):

Retrieving:

```
if False = oSite.AllowEmailLoginCredentials then
MsgBox "Disabled"
End If
```

Specifying:

```
oSite.AllowEmailLoginCredentials = True
```

EFT v6.3 and later

- Property **AllowUnprotectedDataChannel** As Boolean

Used to enable or disable unprotected data channel.

```
HRESULT AllowUnprotectedDataChannel([out, retval] VARIANT_
BOOL *pVal);
```



```
HRESULT AllowUnprotectedDataChannel([in] VARIANT_BOOL
newVal);
```

EFT v8.0.5 and later

- Property **AllowSuppressForgotPassword** As Boolean

```
HRESULT AllowSuppressForgotPassword([out, retval] VARIANT_
BOOL *pVal);
HRESULT AllowSuppressForgotPassword([in] VARIANT_BOOL
newVal);
```

EFT v7.4.11

- Property [AllowUPVariables](#) As Boolean

Retrieves or specifies whether to allow username and passwords credentials as variables in Event Rules.

```
HRESULT AllowUPVariables([out, retval] VARIANT_BOOL* pVal);
HRESULT AllowUPVariables([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
if False = oSite.AllowUPVariables then
MsgBox "Disabled"
End If
```

Specifying:

```
oSite.AllowUPVariables = True
```

EFT v6.3 and later

- Property **AllowWorkspacesOAIRetry** As Boolean

```
HRESULT AllowWorkspacesOAIRetry([out, retval] VARIANT_BOOL
*pVal);
HRESULT AllowWorkspacesOAIRetry([in] VARIANT_BOOL newVal);
```

EFT v7.4.7 and later

- Property [AS2AccountManagementPath](#) As String (read only)

Retrieves URL for AS2 account management.

```
HRESULT AS2AccountManagementPath([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "AS2 management path is: " &  
oSite.AS2AccountManagementPath
```

EFT v6.3 and later

- Property [AS2Certificate](#) As String

Retrieves or specifies AS2 certificate path.

```
HRESULT AS2Certificate([out, retval] BSTR* pVal);  
HRESULT AS2Certificate([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 certificate is: " & oSite.AS2Certificate
```

Specifying:

```
oSite.AS2Certificate = "C:\MyCerts\AS2Cert.crt"
```

EFT v6.3 and later

- Property [AS2Identifier](#) As String

Retrieves or specifies the AS2 identifier.

```
HRESULT AS2Identifier([out, retval] BSTR* pVal);  
HRESULT AS2Identifier([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 identifier is: " & oSite.AS2Identifier
```

Specifying:

```
oSite.AS2Identifier = "my_as2_id"
```

EFT v6.3 and later

- Property [AS2Passphrase](#) As String

Retrieves or specifies the AS2 passphrase.

```
HRESULT AS2Passphrase([out, retval] BSTR* pVal);
HRESULT AS2Passphrase([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 passphrase is: " & oSite.AS2Passphrase
```

Specifying:

```
oSite.AS2Passphrase = "new_passphrase"
```

EFT v6.3 and later

- Property [AS2PrivateKey](#) As String

Retrieves or specifies the AS2 private key.

```
HRESULT AS2PrivateKey([out, retval] BSTR* pVal);
HRESULT AS2PrivateKey([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "AS2 certificate is: " & oSite.AS2PrivateKey
```

Specifying:

```
oSite.AS2PrivateKey = "C:\MyCerts\AS2Cert.crt"
```

EFT v6.3 and later

- Sub **AssignEventRuleCustomCommandWorkingFolder**(bstrCustomCommandName As String, bstrWorkingFolder As String)

Updates all instances of a given custom command in the event rule list to have a specified working folder.

```
HRESULT AssignEventRuleCustomCommandWorkingFolder(
    [in] BSTR bstrCustomCommandName,
    [in] BSTR bstrWorkingFolder);
```

Example (VBScript):

```
Dim strCommandName: strCommandName = "RunScript"
Dim strNewWorkingFolder: strNewWorkingFolder =
"C:\Script\folder"
oSite.AssignEventRuleCustomCommandWorkingFolder
(strCommandName,
    strNewWorkingFolder)
```

EFT v4.3.4 and later

- Property [AssignFullPermissionsForHomeFolder](#) As Boolean

In the EFT administration interface, this property is enabled on the Site's Security tab, in the Account Security area.

Retrieves whether new users have full permissions in their home folders, and to either grant or deny users full permissions to their home folders. True = Granted (check box selected); False = Denied (check box cleared).

The [AutoCreateHomeFolder](#) property must be set to TRUE before you can use this property. If you do not grant users full permissions to their home folders, they will inherit permissions from their groups.

```
HRESULT AssignFullPermissionsForHomeFolder(
[out, retval] VARIANT_BOOL *pVal);
[in] VARIANT_BOOL newVal);
```

All versions

- Property [AutoBanIPsPermanently](#) As Boolean

Retrieves or specifies whether to automatically ban IP addresses permanently.

```
HRESULT AutoBanIPsPermanently([out, retval] VARIANT_BOOL
*pVal);
HRESULT AutoBanIPsPermanently([in] VARIANT_BOOL newVal);
```

True=Permanently; False=Temporarily

All versions

- Property [AutobanSensitivity](#) As [AutobanSensitivity](#)

Retrieves or specifies autoban sensitivity.

```
HRESULT AutobanSensitivity([out, retval] AutobanSensitivity
*pVal);
HRESULT AutobanSensitivity([in] AutobanSensitivity newVal);
```

Example (VBScript)::

Retrieving:

```
If Site.AutobanSensitivity <> SensitivityOff Then
MsgBox "Flood/Hammer Autoban is on for the site"
End If
```

Specifying:

```
Site.AutobanSensitivity = SensitivityVeryHigh
```

EFT v6.1 and later

- Property [AutoCreateHomeFolder](#) As Boolean

In the EFT administration interface, this property is enabled on the Site's Security tab, in the Account Security area.

Retrieves whether the Site will automatically create a home folder for users, and to either enable or disable the automatic folder creation.

```
HRESULT AutoCreateHomeFolder([out, retval] VARIANT_BOOL
*pVal);
[in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

NOTE: The AutoCreateHomeFolder property must be set to TRUE before you can use the AssignFullPermissionsForHomeFolder property. If you do not grant users full permissions to their home folders, they will inherit permissions from their groups.

All versions

- Property [AverageSpeed](#) As Long (read only)

Retrieves the average upload/download speed on the Site.

```
HRESULT AverageSpeed([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Average speed: " & CStr(oSite.AverageSpeed)
```

EFT v6.3 and later

- Property **BanIPForInvalidLoginAttempts** As Boolean

When banning an IP address for invalid login attempts, specify whether to count incorrect username and correct username + incorrect password or only incorrect username attempts.

```
HRESULT BanIPForInvalidLoginAttempts([out, retval] VARIANT_BOOL* pVal);
HRESULT BanIPForInvalidLoginAttempts([in] VARIANT_BOOL newVal);
```

TRUE = Count both 'incorrect username' and 'correct username + incorrect password'

FALSE = Count only incorrect username attempts

EFT v6.4 and later

- Property **BanIPForInvalidLoginAttemptsWithExistingUsername** As Boolean

Retrieves or specifies whether to ban an IP address after excessive invalid login attempts with existing username and wrong password.

```
HRESULT BanIPForInvalidLoginAttemptsWithExistingUsername([out, retval] VARIANT_BOOL* pVal);
HRESULT BanIPForInvalidLoginAttemptsWithExistingUsername([in] VARIANT_BOOL newVal);
```

EFT v6.4 and later

- Property [BanIPOnDOS](#) As Boolean

Retrieves or specifies whether to ban an IP address after excessive invalid commands.

```
HRESULT BanIPOnDOS([out, retval] VARIANT_BOOL *pVal);
HRESULT BanIPOnDOS([in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

All versions

- Property [BlockAntiTimeOut](#) As Boolean

Retrieves or specifies whether to block anti-timeout schemes (NOOP).

```
HRESULT BlockAntiTimeOut([out, retval] VARIANT_BOOL *pVal);
HRESULT BlockAntiTimeOut( [in] VARIANT_BOOL newVal);
```

True = Blocked; False = Allowed

All versions

- Property [BlockSiteToSite](#) As Boolean

Retrieves or specifies whether to block Site-to-Site (FXP) transfers.

```
HRESULT BlockSiteToSite([out, retval] VARIANT_BOOL *pVal);
HRESULT BlockSiteToSite( [in] VARIANT_BOOL newVal);
```

True = Blocked; False = Allowed

All versions

- Function **CancelTransfer**(bstrPath As String) As Boolean

Close all HTTP sockets that upload/download (including POST upload) a file specified by the path parameter. Note that this is not effective for FTP or SFTP.

```
HRESULT CancelTransfer( [in] BSTR bstrPath, [out, retval]
VARIANT_BOOL *pSuccess);
```

EFT v4.3.4 and later

- Sub **ChangeUserPassword**(bstrUser As String, bstrPwd As String, nPwdType As Int)

Changes a user's password and password type.

```
HRESULT ChangeUserPassword([in] BSTR bstrUser,
[in] BSTR bstrPwd,
[in] int nPwdType);
```

Example:

```
site.ChangeUserPassword "test",
"$1$Bn3YdWKv$.BS3Qb2UQTV4SmV1JNN.w/", 5
```

6 = nPwdType - For importing hashes of unknown type from another source. (When the user logs into EFT, EFT would cycle through possible hashes for the plaintext password to see which one matches the supplied hash. When a match is found, the user is able to log in. As well, EFT will rehash the password with its current default hashing algorithm and store that as the user's hashed password.)

NOTE: DES-format passwords are not supported. These are password formats without the salt defined, that is, without the \$1\$.

All versions

- Property **ConnectedUsers** As Variant (read only)

See also [ICConnectedUser - Connected User Interface](#).

Retrieves an array of users currently connected to the Site.

```
HRESULT ConnectedUsers([out, retval] VARIANT* pVal);
```

Example (VBScript):

```
For Each u In oSite.ConnectedUsers
  MsgBox "User: " & u.Login
Next
```

EFT v6.3 and later

- Property [ConnectMessage](#) As String

Retrieves or specifies the connect message.

```
HRESULT ConnectMessage([out, retval] BSTR *pVal);
HRESULT ConnectMessage([in] BSTR newVal);
```

All versions

- Property **ContentIntegrityControlProfileList**

Provides read-only list of CIC profiles

```
HRESULT ContentIntegrityControlProfileList([out, retval]
  VARIANT *pVal);
```

EFT v8

- Function **CreateCommand**(bstrName As String) As [ICCommandSettings](#)

Creates a new command.

```
HRESULT CreateCommand(
  [in] BSTR bstrName,
  [out, retval] ICCommandSettings **prop);
```


All versions

- Function **CreateComplexPassword**([bstrUserName]) As String

Creates a complex password that meets the complexity criteria specified for the site.

```
HRESULT CreateComplexPassword(
    [in, optional] VARIANT bstrUserName,
    [out, retval] BSTR *bstrPassword);
```

Example (VBScript):

The following example generates a password that adheres to the complexity policy enforced for an existing user:

```
Dim strNewPassword
strNewPassword = oUserSettings.CreateComplexPassword("test")
WScript.Echo "New Password: " strNewPassword
Call oUserSettings.ChangeUserPassword("test",
    strNewPassword, 1)
```

This example illustrates how to obtain a password from the Server that meets the complexity requirements for the Settings Template that is configured as the default for the Site, which is useful when creating new users:

```
Dim strNewPassword
strNewPassword = oUserSettings.CreateComplexPassword()
WScript.Echo "New Password: " strNewPassword
Call oUserSettings.CreateUser("test", strNewPassword,
    0, "New User Description")
```

EFT v5.0.1 and later

- Function **CreateComplexPasswordEx**([bstrUserName], [bstrSettingsLevel]) As String

Creates a complex password that meets the complexity criteria specified for the Settings Template.

```
HRESULT CreateComplexPasswordEx(
    [in, optional] VARIANT bstrUserName,
    [in, optional] VARIANT bstrSettingsLevel,
    [out, retval] BSTR *bstrPassword);
```

Example (VBScript):

```
strPass = oSite.CreateComplexPasswordEx("", "")
```

EFT v6.3 and later

- Sub **CreateGuestSettingsLevel**(bstrName As String)

```
HRESULT CreateGuestSettingsLevel([in] BSTR bstrName);
```

EFTv7.4 and later

See also [GuestSettingsLevels](#) property.

- Sub [CreatePermissionGroup](#)(bstrName As String)

Creates a new permission group. Permission groups control user access to files and folders.

```
HRESULT CreatePermissionGroup([in] BSTR bstrName);
```

All versions

- Function **CreatePGPKey**(bstrFullName As String, bstrEmailAddress As String, dtExpirationDate As Date, nKeyType As [PGPKeyType](#), nKeyCipher As [PGPKeyCipher](#), lKeySize As Long, bstrPassphrase As String) As String

Create a PGP Key on the Site.

```
HRESULT CreatePGPKey(
    [in] BSTR bstrFullName,
    [in] BSTR bstrEmailAddress,
    [in] DATE dtExpirationDate,
    [in] PGPKeyType nKeyType,
    [in] PGPKeyCipher nKeyCipher,
    [in] LONG lKeySize,
    [in] BSTR bstrPassphrase,
    [out, retval] BSTR* pVal);
```

Example (VBScript):

```
keyId = oSite.CreatePGPKey("test
    key", "example@globalscape.com", expDate, PGPKeyType_RSA,
    PGPKeyCipher_IDEA, 1024, "password")

MsgBox "New key id is " & keyId
```

See also [RemovePGPKey](#), [ExportPGPKey](#), and [ImportPGPKey](#).

EFT v8 and later

- Sub **CreatePhysicalFolder**(bstrNewFolder As String)

Given a relative path, creates a new physical folder. (Use forward slashes instead of back slashes.)

```
HRESULT CreatePhysicalFolder([in] BSTR bstrNewFolder);
```

Example (VBScript):

```
Call Object.CreatePhysicalFolder("/folderpath/")
'another example:
MySite.CreatePhysicalFolder("/Usr/jsmith/")
'Note that backslashes "\" will not work:
MySite.CreatePhysicalFolder("\Usr\jsmith\") <--will fail!
```

All versions

- Sub **CreateSettingsLevel**(bstrName As String, bstrDescription As String, [bSetAsDefault As Boolean = 0])

Creates a Settings Template. The "BOOL bMakeDefault" parameter allows you to specify whether the new Settings Template is the default template. When the "BOOL bMakeDefault" parameter is not used, the Settings Template is not made the default.

```
HRESULT CreateSettingsLevel(
[in] BSTR bstrName,
[in] BSTR bstrDescription),
[in] BOOL bMakeDefault);
```

All versions

- Sub **CreateSSHKeyPair**(keyBits As Long, bstrPassword As String, pPub, pPriv)

Creates SSH key pair. The keys bodies are returned from this method by two byte arrays. (In v8.0.4 and later, changed keyBits to KeyParams)

```
HRESULT CreateSSHKeyPair(
[in] LONG keyBits,
[in] BSTR bstrPassword,
[out] VARIANT* pPub,
[out] VARIANT* pPriv);
```

Example (VBScript):

```
oSite.CreateSSHKeyPair 1024, "password", pub, priv
For Each b In priv
```

```
WScript.StdOut.Write(Chr(b))
Next
```

NOTE: When creating a key with RSA/8192, the EFT administration interface is locked and the user cannot select or use any functions/options while COM script is running until finished. The user will sometimes see a spinning circle until COM has finished.

See also [RemoveSSHKey](#) and [RenameSSHKey](#).

EFT v8 and later

- Sub **CreateUser**(bstrLogin As String, bstrPwd As String, nPwdType As Int, bstrDescription As String, [abTwoFactorAuthentication As SFTPAAdvBool = -2])

Creates a user with the default values. (The CreateUser method creates users with the default values.)

```
HRESULT CreateUser(
[in] BSTR bstrLogin,
[in] BSTR bstrPwd,
[in] int nPwdType,
[in] BSTR bstrDescription,
[in, defaultvalue(abInherited)] SFTPAAdvBool
abTwoFactorAuthentication);
```

```
oSite.CreateUser "tuser1", "testpassword", 0, "New User"
```

For the method to work, you must:

Disable enforcement of complex passwords, provide a complex password. For example:

```
oSite.CreateUser "test",
"$1$Bn3YdWKv$.BS3Qb2UQTV4SmV1JNN.w/", 0, "New User")
```

Or call the CreateComplexPassword method and use the resulting string in the CreateUser method. For example, in pseudo code:

```
strUserPass = oUserSettings.CreateComplexPassword()
oSite.CreateUser "test", strUserPass, 0, "New User")
```

See also: [Difference between VARIANT_BOOL and SFTPAAdvBool](#)

All versions

- Sub **CreateUserEx**(bstrLogin As String, bstrPwd As String, nPwdType As Int, bstrDescription As String, bstrFullName As String, bCreateHomeFolder As Boolean, bFullPermissionsForHomeFolder As Boolean, [bstrSettingsLevel As String], [abTwoFactorAuthentication As SFTPAAdvBool = -2])

Creates a user and allows you to set values for all parameters. (The CreateUserEx method allows you to set values for all parameters.)

```
HRESULT CreateUserEx(
    [in] BSTR bstrLogin,
    [in] BSTR bstrPwd,
    [in] int nPwdType,
    [in] BSTR bstrDescription,
    [in] BSTR bstrFullName,
    [in] VARIANT_BOOL bCreateHomeFolder,
    [in] VARIANT_BOOL bFullPermissionsForHomeFolder,
    [in, optional] BSTR bstrSettingsLevel,
    [in, defaultvalue(abInherited)] SFTPAAdvBool
    abTwoFactorAuthentication);
```

Example (C#):

```
selectedSite.CreateUserEx(j.ToString(),j.ToString
(),0,j.ToString(),j.ToString(),true,
    false,"Default settings", SFTPAAdvBool.abFalse);
```

See also: [Difference between VARIANT_BOOL and SFTPAAdvBool](#)

For an example using this method, refer to the CreateUsers script in [Script Examples](#).

NOTE: The CreateUserEx method will fail if a simple password is provided when enforce complex passwords is enabled in EFT. For example, the following will fail if enforce complex passwords is enabled in EFT:

```
Site.CreateUserEx "tuser1", "testpassword",
    "New User"
```

For the method to work, you must do one of the following:

Disable enforcement of complex passwords.

Provide a complex password. For example:

```
Site.CreateUserEx "test",
"$1$Bn3YdWKv$.BS3Qb2UQTV4SmV1JNN.w/",
0, "New User"
```

Call the `CreateComplexPassword` method and use the resulting string in the `CreateUserEx` method. For example (in pseudo code):

```
strUserPass = oUserSettings.CreateComplexPassword()
Site.CreateUserEx "test", strUserPass,
0, "New User"
```

All versions

- Sub **CreateUserEx2**(pdispParams As Object)

Creates a new user and includes all of the user/client details, including the email address; ensures that the email address is available if an Event Rules is configured to send an e-mail to the user.

```
HRESULT CreateUserEx2 ([in] IDispatch* pdispParams);
```

Refer to [ICINewUserData Interface](#) for details.

EFT v6.3.8 and later

- Sub **CreateVirtualFolder**(bstrNewFolder As String, bstrTarget As String)

Given a relative path and a physical target path, creates a new virtual folder.

```
HRESULT CreateVirtualFolder(
[in] BSTR bstrNewFolder,
[in] BSTR bstrTarget),
[in, defaultvalue(-1)] VARIANT_BOOL bCheckTarget); (This was
added for 6.3.8)
```

Example (VBScript):

```
Call oSite.CreateVirtualFolder("/pub/temp/",
"C:\Temp\")
```

All versions

- Sub **CreateVirtualFolders**(paths, checkEachPhysicalPath As Boolean)

```
HRESULT CreateVirtualFolders(
    [in] VARIANT paths,
    [in] VARIANT_BOOL checkEachPhysicalPath);
```

EFT v7.4.13

- Property [DataSanitizationMethod](#) As [DataSanitizationMethod](#)

Retrieves or specifies the data sanitization method.

```
HRESULT DataSanitizationMethod([out, retval]
    DataSanitizationMethod* pVal);
HRESULT DataSanitizationMethod([in] DataSanitizationMethod
    newVal);
```

Example (VBScript):

Retrieving:

```
if DataSanitizationNoWipe = oSite.DataSanitizationMethod
    then
MsgBox "No wipe method is specified"
End If
```

Specifying:

```
oSite.DataSanitizationMethod =
    DataSanitizationPseudorandomData
```

EFT v6.3 and later

- Sub **DecryptFolder**(bstrFolder As String, vbResursively As Boolean)

Specifies whether to decrypt a folder.

```
HRESULT DecryptFolder(
    [in] BSTR bstrFolder,
    [in] VARIANT_BOOL vbResursively);
```

Example (VBScript):

```
oSite.DecryptFolder "/some_folder", True
```

EFT v6.3 and later

- Property **DefaultPGPKeyID** As String (read-only)

Retrieves the default PGP Key ID or empty string if Site has not default key.

```
HRESULT DefaultPGPKeyID([out, retval] BSTR* pVal);
```

See also [SetDefaultPGPKey](#)

EFT v6.4 and later

- Property **DefaultPGPKeyPassphrase** As String (read-only)

Retrieves the default PGP Key passphrase.

```
HRESULT DefaultPGPKeyPassphrase([out, retval] BSTR* pVal);
```

See also [SetDefaultPGPKey](#)

EFT. v6.4 and later

- Property **DefaultSettingsLevel** as String

Retrieves or specifies which template is currently selected as the default.

```
HRESULT DefaultSettingsLevel([out, retval] BSTR* bstrName);
```

```
HRESULT DefaultSettingsLevel([in] BSTR bstrName);
```

EFT v7.0 and later

- Method **DeleteContentIntegrityControlProfile**

Delete CIC profile by profile name

```
HRESULT DeleteContentIntegrityControlProfile([in] BSTR  
pProfileName);
```

EFT v8

- Sub **DeleteEncryptedFolder**(pIEncryptedFolder As Object)

```
HRESULT DeleteEncryptedFolder([in] IDispatch*  
pIEncryptedFolder);
```

EFT v7.4.13

- Sub **DeleteUploadForm**(formName As String)

Used to delete an Upload Form from a Site.

```
HRESULT DeleteUploadForm([in] BSTR formName);
```


EFT v8.0.4

- Sub **DisableInheritPermissions**(bstrFolder As String, bRemovePerm As Boolean)
Disable Inherit permissions on a specified folder.

```
HRESULT DisableInheritPermissions(
    [in] BSTR bstrFolder,
    [in] VARIANT_BOOL bRemovePerm);
```

TRUE=Disabled; FALSE=Enabled

Example (VBScript):

```
oSite.DisableInheritPermissions "/some_folder", False
```

EFT v6.3 and later

- Sub [DisableSFTP](#)()
Disables SFTP access on the Site.

```
HRESULT DisableSFTP(void);
```

EFT v6.1 and later

- Property [DisconnectOnDOS](#) As Boolean
Retrieves or specifies whether to disconnect a user after a certain number of invalid commands.

```
HRESULT DisconnectOnDOS([out, retval] VARIANT_BOOL *pVal);
HRESULT DisconnectOnDOS([in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

All versions

- Property **DMZConnected** As Boolean (read only)
Retrieves whether the Site is connected to DMZ Gateway.

```
HRESULT DMZConnected([out, retval] VARIANT_BOOL* pVal);
```

Example (VBScript):

```
if True = oSite.DMZConnected then
    MsgBox "DMZ is connected."
else
```

```
MsgBox "DMZ is not connected."
end if
```

EFT v6.3 and later

- Property **DMZGateway** As Object (read-only)

Retrieves the properties of the DMZ Gateway with [CIDMZGateway](#).

```
HRESULT DMZGateway([out, retval] IDispatch** prop);
```

Example (VBScript):

```
Set oDMZ = oSite.DMZGateway
```

EFT v6.3 and later

- Function **DoesUsernameExist**(bstrUserName As String) As Boolean

Retrieves or specifies whether a username exists.

```
HRESULT DoesUsernameExist(
[in] BSTR bstrUserName,
[out, retval] VARIANT_BOOL *bExist);
```

True=Exists; False=Does not exist

EFT v6.3 and later

- Property [DOSLimit](#) As Long

Retrieves or specifies the number of consecutive invalid commands that will cause the Site to disconnect a user.

```
HRESULT DOSLimit([out, retval] long *pVal);
HRESULT DOSLimit([in] long newVal);
```

All versions

- Property **DropOffPortalPath** As String

```
HRESULT DropOffPortalPath([out, retval] BSTR *pVal);
HRESULT DropOffPortalPath([in] BSTR newVal);
```

EFT v7.4.2 and later

- Property [EmailLoginCredentialsMode](#) As [EmailLoginCredentialsMode](#)

If [AllowEmailLoginCredentials](#) is enabled, EmailLoginCredentialsMode retrieves how they are sent (username and password in the same email, separate emails, or username only).

```
HRESULT EMailLoginCredentialsMode([out, retval]
EMailLoginCredentialsMode* pVal);
HRESULT EMailLoginCredentialsMode([in]
EMailLoginCredentialsMode newVal);
```

Example (VBScript):

Retrieving:

```
if EmailCredentialsBoth = oSite.EMailLoginCredentialsMode
    then
MsgBox "Username and password together."
End If
```

Specifying:

```
oSite.EmailLoginCredentialsMode =
EmailCredentialsUsernameOnly
```

EFT v6.3 and later

- Property **EnableAccountManagementOverHTTPS** As Boolean

(The Account Management (change password) page is not available when CAC is enabled.)

Retrieves or specifies whether account management over HTTPS is enabled. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT EnableAccountManagementOverHTTPS([out, retval]
VARIANT_BOOL* pVal);
HRESULT EnableAccountManagementOverHTTPS([in] VARIANT_BOOL
newVal);
```

True = Enabled; False = Disabled

Example (VBScript):

Retrieving:

```
MsgBox "AS2 managing over https is enabled:
" & CStr(oSite.EnableAccountManagementOverHTTPS)
```

Specifying:

```
oSite.EnableAccountManagementOverHTTPS = True
```

EFT v6.3 and later

- Property [EnableAS2](#) As Boolean

Retrieves or specifies whether AS2 is enabled on the Site.

```
HRESULT EnableAS2([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableAS2([in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

Example (VBScript):

Retrieving:

```
if False = oSite.EnableAS2 then
  MsgBox "Disable"
else
  MsgBox "Remove"
end if
```

Specifying:

```
oSite.EnableAS2 = True
```

EFT v6.3 and later

- Property **EnableCAC** As Boolean

(The Account Management change password page is not available when CAC is enabled.)

Retrieves or specifies whether CAC is enabled on the Site. (Common Access Card) authentication is available in EFT Enterprise with the High Security module (in versions before 7.4.7) and Advanced Security module (in v7.4.7 and later) on LDAP Sites with SSL (HTTPS or FTPS) enabled.)

```
HRESULT EnableCAC([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableCAC([in] VARIANT_BOOL newVal);
```

EFT v6.4.3 and later

- Property **EnableHSTS** As Boolean

(The site must be restarted)

```
HRESULT EnableHSTS([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableHSTS([in] VARIANT_BOOL newVal);
```

EFT v7.4.13 and later

- Sub **EnableInheritPermissions**(bstrFolder As String, bRemovePerm As Boolean)

Enables Inherit permissions on a specified folder.

```
HRESULT EnableInheritPermissions(
[in] BSTR bstrFolder,
[in] VARIANT_BOOL bRemovePerm);
```

TRUE=Enabled; FALSE=Disabled

Example (VBScript):

```
oSite.EnableInheritPermissions "/some_folder",
False
```

EFT v6.3 and later

- Property **EnableMfa** As Boolean

Used to enable or disable MFA on the Site.

```
HRESULT EnableMfa([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableMfa([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **EnableMTC** As Boolean

Enables MTC on a Site.

```
HRESULT EnableMTC([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableMTC([in] VARIANT_BOOL newVal);
```

EFT v6.5.16 and later

- Property [EnablePasswordHistory](#) As Boolean

Retrieves or specifies whether password history is enabled.

```
HRESULT EnablePasswordHistory([out, retval] VARIANT_BOOL*
pVal);
HRESULT EnablePasswordHistory([in] VARIANT_BOOL newVal);
```

TRUE=Password history is enabled; FALSE=Password history is not enabled.

Example (VBScript):

Retrieving:

```
if False = oSite.EnablePasswordHistory then
MsgBox "Password history is disabled"
End If
```

Specifying:

```
oSite.EnablePasswordHistory = True
```

EFT v6.3 and later

- Property [EnablePGPLogging](#) As Boolean

Retrieves or specifies whether OpenPGP logging is enabled.

```
HRESULT EnablePGPLogging([out, retval] VARIANT_BOOL* pVal);
HRESULT EnablePGPLogging([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; FALSE=Disabled

Example (VBScript):

Retrieving:

```
MsgBox "PGP logging: " & CStr(oSite.EnablePGPLogging)
```

Specifying:

```
oSite.EnablePGPLogging = False
```

EFT v6.3 and later

- Property [EnablePortRange](#) As Boolean

Retrieves or specifies whether to enable the PASV port range.

```
HRESULT EnablePortRange([out, retval] VARIANT_BOOL *pVal);
[in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

All versions

- Property **EnableProxyProtocol** As Boolean

Enables or disables the proxy protocol.

```
HRESULT EnableProxyProtocol([out, retval] VARIANT_BOOL*
pVal);
HRESULT EnableProxyProtocol([in] VARIANT_BOOL newVal);
```

EFT v8.0.5

- Sub [EnableSFTP\(SFTPKeyFile](#) As String, [SFTPKeyPassphrase](#) As String)

Enables SFTP access on the Site. Specify the [SFTPKeyFile](#) and [SFTPKeyPassphrase](#).

In EFT v8 and later: If SFTPKeyFile is non-empty, then EFT tries to import the key to the Site and enable SFTP; otherwise, EFT tries to enable SFTP using the current key (see [GetSSHKeyForSFTP](#)/ [SetSSHKeyForSFTP](#)).

```
HRESULT EnableSFTP(BSTR SFTPKeyFile,BSTR SFTPKeyPassphrase);
```

EFT v6.1 and later

- Property **EnableUsers** As Boolean

Retrieves or specifies whether Web Services is enabled. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT EnableWebServices([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableWebServices([in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

Example (VBScript):

Retrieving:

```
MsgBox "SFTP port: " & CStr(oSite.SFTPPort)
```

Specifying:

```
oSite.SFTPPort = 40
```

EFT v6.3 and later

- Sub **EnableUsers**(arrUsers, val As [SFTPAAdvBool](#))

This method is used to enable a user account. (1=enable; 0=disable)

```
HRESULT EnableUsers([in] VARIANT arrUsers, [in] SFTPAAdvBool val);
$site.EnableUsers("Felipe",1)
```

EFT v8.0.3

- Property [EnableWebServices](#) As Boolean

Retrieves or specifies whether Web Services is enabled. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT EnableWebServices([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableWebServices([in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

Example (VBScript):

Retrieving:

```
MsgBox "SFTP port: " & CStr(oSite.SFTPPort)
```

Specifying:

```
oSite.SFTPPort = 40
```

EFT v6.3 and later

- Property **EnableWebSSO** As Boolean

```
HRESULT EnableWebSSO([in] VARIANT_BOOL newVal);
HRESULT EnableWebSSO([out, retval] VARIANT_BOOL* pVal);
```

EFT v7.3.3

- Property **EnableWorkspaces** As Boolean

Enable or disable use of Workspaces on the Site.

```
HRESULT EnableWorkspaces([out, retval] VARIANT_BOOL* pVal);
HRESULT EnableWorkspaces([in] VARIANT_BOOL pVal);
```

EFT v7.1 and later

- Property **EnableWorkspacesDropOff** As Boolean


```
HRESULT EnableWorkspacesDropOff([out, retval] VARIANT_BOOL
*pVal);
HRESULT EnableWorkspacesDropOff([in] VARIANT_BOOL pVal);
```

EFT v7.4.2 and later

- Property **EnableWorkspacesExternalUsers** As Boolean

Enable or disable use of Workspaces by external users.

```
HRESULT EnableWorkspacesExternalUsers([out, retval] VARIANT_
BOOL* pVal);
HRESULT EnableWorkspacesExternalUsers([in] VARIANT_BOOL
pVal);
```

EFT v7.1 and later

- Property **EnableWorkspacesOAI** As Boolean

Enable or disable use of EFT Outlook Add-In on the Site.

```
HRESULT EnableWorkspacesOAI([in] VARIANT_BOOL pVal);
HRESULT EnableWorkspacesOAI([out, retval] VARIANT_BOOL*
pVal);
```

EFT v7.3.3 and later

- Property **EnableWorkspacesRequestFile** As Boolean

Enable Workspaces users to request files and specify the path to the Reply portal.

```
HRESULT EnableWorkspacesRequestFile([out, retval] VARIANT_
BOOL *pVal);
HRESULT EnableWorkspacesRequestFile([in] VARIANT_BOOL pVal);
```

Examples

```
If oSite.EnableWorkspacesRequestFile
Then
    MsgBox "Request File Portal URL is " &
oSite.RequestFilePortalPath
End if
```

```
oSite.EnableWorkspacesRequestFile
    = True
oSite.RequestFilePortalPath = "/requestPortal"
```

EFT v8.0

- Property **EncryptedFolders** As Variant (read only)

```
HRESULT EncryptedFolders([out, retval] VARIANT* pVal);
```

EFT v7.4.13 and later

- Property **EncryptionKey** As String

Used to provide the encryption key for encrypted folders.

```
HRESULT EncryptionKey([in] BSTR newVal);
HRESULT EncryptionKey([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

- Property **EnforcedWorkspacesOAIMinFileSizeMB** As DWord

Specify file size above which they will be managed by Workspaces instead of Outlook.

```
HRESULT EnforcedWorkspacesOAIMinFileSizeMB([out, retval]
ULONG* pVal);
HRESULT EnforcedWorkspacesOAIMinFileSizeMB([in] ULONG pVal);
```

EFT v7.3.3 and later

- Property **EnforceWorkspacesOAIWhenFilesGreaterThan** As Boolean

Enable or disable Workspaces management of files.

```
HRESULT EnforceWorkspacesOAIWhenFilesGreaterThan(
[out, retval] VARIANT_BOOL* pVal);
HRESULT EnforceWorkspacesOAIWhenFilesGreaterThan(
[in] VARIANT_BOOL pVal);
```

EFT v7.3.3 and later

- Sub **EncryptFolder**(bstrFolder As String, vbResursively As Boolean)

Specifies whether folder is encrypted.

```
HRESULT EncryptFolder([in] BSTR bstrFolder,[in] VARIANT_BOOL
vbResursively);
```

Example (VBScript):

```
oSite.EncryptFolder "/some_folder", True
```

EFT v6.3 and later

- Function **EventRuleFolders()** As Object

Retrieve list of Event Rules folders.

```
HRESULT EventRuleFolders([out, retval] IDispatch**
ppdispEventRuleFolders);
```

EFT v7.2 and later

- Function **EventRules**(type As [EventType](#)) As Object

Retrieves the list of Event Rules of a certain type (e.g., all Timer Event Rules).

```
HRESULT EventRules([in] EventType type,
[out, retval] IDispatch** ppdispRules);
```

Example (VBScript):

```
Set timerRules = site.EventRules (&H1001)
'Retrieve all the Timer event rules
```

Example (C#):

```
CIEventRules rules = (CIEventRules)site.EventRules
(EventType.MonitorFolder);
```

For an example using this method, refer to the EventRuleCreation script in [Script Examples](#).

EFT v5.2 and later

- Property [ExitMessage](#) As String

Retrieves or specifies exit message.

```
HRESULT ExitMessage([out, retval] BSTR *pVal);
HRESULT ExitMessage([in] BSTR newVal);
```

All versions

- Sub **ExportEventRules**(bstrDstFile As String)

See also [ImportEventRules](#).

```
HRESULT ExportEventRules([in] BSTR bstrDstFile);
```

EFT v7.2 and later

- Sub **ExportPGPKey**(bstrKeyId As String, vbIncludePrivate As Boolean, bstrFileName As String)

Export PGP key from Site to file by key's ID.

```
HRESULT ExportPGPKey(
    [in] BSTR bstrKeyId,
    [in] VARIANT_BOOL vbIncludePrivate,
    [in] BSTR bstrFileName);
```

Example (VBScript):

```
oSite.ExportPGPKey "0x12345678", False, "C:\PGP\pgpkey.asc"
```

See also [CreatePGPKey](#), [ImportPGPKey](#), and [RemovePGPKey](#)

EFT v8 and later

- Sub **ExportSSHPrivateKey**(bstrKeyId As String, vbIncludePrivate As Boolean, bstrFileName As String)

Export SSH key as array of bytes by key's index in SSHKeys array.

```
HRESULT ExportSSHPrivateKey(
    [in] LONG keyIndex, [out, retval] VARIANT* pPub);
```

Example (VBScript):

```
priv = oSite.ExportSSHPrivateKey(keyIndex)
```

See also [ImportSSHPrivateKey](#)

EFT v8 and later

- Sub **ExportSSHPublicKey**(keyIndex As Long) As Variant

Export SSH key as an array of bytes by key's index in SSHKeys array.

```
HRESULT ExportSSHPublicKey(
    [in] LONG keyIndex, [out, retval] VARIANT* pPub);
```

Example (VBScript):

```
priv = oSite. ExportSSHPrivateKey(keyIndex)
```

See also [ImportSSHPublicKey](#)

EFT v8 and later

- Function **ExportTrustedCertificate**(IID As Long) As String

Exports a trusted certificate.

```
HRESULT ExportTrustedCertificate(
[in] long IID,[out, retval] BSTR *bstrCertData);
```

All versions

- Property [ForcePasswordResetOnInitialLogin](#) As Boolean

Retrieves or specifies whether the option to force users to change their password upon next login is turned on. The value can be overridden in the Settings Template or user account by the ICIClientSettings::SetChangePwd method. If the option is ON, it does not mean the user will be necessary forced to change password on next login, because the user's password might have been changed since you switched the option from OFF to ON in the administration interface.

```
HRESULT ForcePasswordResetOnInitialLogin([out, retval]
VARIANT_BOOL *pVal);
HRESULT ForcePasswordResetOnInitialLogin([in] VARIANT_BOOL
val);
```

Example (VBScript):

Retrieving:

```
If Not Site.ForcePasswordResetOnInitialLogin Then
MsgBox "Reset password is turned off for this site"
End If
```

Specifying:

```
Site.ForcePasswordResetOnInitialLogin = False
Site.ForcePasswordResetOnInitialLogin = True 'switching from
OFF to ON
'will force all users to
```

```
change password on next login
(if not overridden at lower levels)
```

EFT v6.2 and later

- Sub ForceSynchronizeUser(bstrUser As String)

```
HRESULT ForceSynchronizeUser([in] BSTR bstrUser);
```

EFT v7.4 and later

- Sub ForceSynchronizeUserDatabase()

Launches (asynchronously) the user database synchronization thread in EFT (equivalent to clicking View > Refresh User Database). This method does not change the client's COM object. You must use the method [Server.RefreshSettings](#) after waiting a suitable amount of time for the database synchronization to complete.

```
HRESULT ForceSynchronizeUserDatabase();
```

All versions

- Property FTPEncoding as string

Refer to [TransferEncodingType](#) in the Enum Reference.

Retrieves or specifies the transfer encoding mode for FTP and FTPS.

TransferEncodingUTF8 and TransferEncodingASCII are the only valid values for this property.

```
HRESULT FTPEncoding([out, retval] TransferEncodingType*
pVal);
HRESULT FTPEncoding([in] TransferEncodingType newVal);
```

Example (VBScript):

Retrieving:

```
If Site.FTPEncoding = 2 Then
  'ASCII mode
End If
```

Specifying:

```
'set ASCII mode for SFTP
Site.FTPEncoding = 2
```

EFT v6.5 and later

- Property [FTPSExplicitPort](#) As Long

Retrieves FTPS explicit port. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT FTPSExplicitPort([out, retval] LONG* pVal);
HRESULT FTPSExplicitPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "FTPS explicit port: " & CStr(oSite.FTPSExplicitPort)
```

Specifying:

```
oSite.FTPSExplicitPort = 40
```

EFT v6.3 and later

- Property [FTPSImplicitPort](#) As Long

Retrieves FTPS implicit port. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT FTPSImplicitPort([out, retval] LONG* pVal);
HRESULT FTPSImplicitPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "FTPS implicit port: " & CStr(oSite.FTPSImplicitPort)
```

Specifying:

```
oSite.FTPSImplicitPort = 40
```

EFT v6.3 and later

- Function [GetACLForAdvancedWorkflowsRoot\(\)](#) As Object

See also [SetACLForAdvancedWorkflowsRoot](#).

```
HRESULT GetACLForAdvancedWorkflowsRoot([out, retval]
IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetACLForConnectionProfilesRoot()** As Object

See also [SetACLForConnectionProfilesRoot](#).

```
HRESULT GetACLForConnectionProfilesRoot([out, retval]
IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetACLForCustomCommandsRoot()** As Object

```
HRESULT GetACLForCustomCommandsRoot([out, retval]
IDispatch** ppVal);
```

EFT v7.2 and later

- Function **GetACLForEventRulesRoot()** As Object

```
HRESULT GetACLForEventRulesRoot([out, retval] IDispatch**
ppVal);
```

EFT v7.2 and later

- Function **GetAdvancedWorkflowIndex** (bzName As String) As Long

Retrieves the zero-based index of the Advanced Workflow with the given name.

```
HRESULT GetAdvancedWorkflowIndex(
[in] BSTR bzName, // AW name
[out, retval] long* plIndex);
// zero-based index of the AW
// or -1 if not found
```

Example (VBScript)

```
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")
if idxAW < 0 Then
    'Work with workflow via its index
End If
```

EFT v6.5 and later

- Function **GetAdvancedWorkflowParams**(lIndex As Long) As Object

(See also [SetAdvancedWorkflowParams](#))

Given Advanced Workflow zero-based index, retrieves the Workflow's parameters.

```
HRESULT GetAdvancedWorkflowParams (
    [in] long lIndex,
    [out, retval] IDispatch** ppdispIParams);
```

Example (VBScript):

```
'Display the name in message for each workflow
For idxAW = 0 to site.AdvancedWorkflowsCount - 1
Set AWparams = site.GetAdvancedWorkflowParams(idxAW)
MsgBox AWparams.Name
Next
```

EFT v6.5 and later

- Function **GetAllowCOMB**() As Boolean

(For information about using the COMB command with EFT, refer to [Allowing Multipart Transfers \(COMB Command\)](#) in the *EFT User Guide*.)

For an example using this method, refer to the ConfigureSite script in [Script Examples](#).

Retrieves whether users on the Site are allowed to send the COMB command for multipart transfers.

```
HRESULT GetAllowCOMB([out, retval] VARIANT_BOOL *pVal);
```

True = Enabled; False = Disabled

Example (VBScript):

```
if False = oSite.GetAllowCOMB() then
MsgBox "Disallowed"
end if
```

Example (C#):

```
if (selectedSite.GetAllowCOMB())
```

EFT v6.3 and later

- Function **GetAllowFXP**() As Boolean

For an example using this method, refer to the ConfigureSite script in [Script Examples](#).

Retrieves whether users on the Site are allowed to perform site-to-site (FXP) transfers.

```
HRESULT GetAllowFXP([out, retval] VARIANT_BOOL *pVal);
```

True = Enabled; False = Disabled

Example (VBScript):

```
if False = oSite.GetAllowFXP() then
MsgBox "Disallowed"
end if
```

Example C#:

```
if (selectedSite.GetAllowFXP())
```

EFT v6.3 and later

- Function [GetAllowNoop\(\)](#) As Boolean

Retrieves whether users on the Site are allowed to use the NOOP command to keep a connection open.

```
HRESULT GetAllowNoop([out, retval] VARIANT_BOOL *pVal);
```

True = Enabled; False = Disabled

Example (VBScript):

```
if False = oSite.GetAllowNoop() then
MsgBox "Disallowed"
end if
```

All versions

- Function [GetAllowXCRC\(\)](#) As Boolean

Retrieves whether users on the Site are allowed to send the XCRC commands to confirm successful transfer.

```
HRESULT GetAllowXCRC([out, retval] VARIANT_BOOL *pVal);
```

True = Enabled; False = Disabled

Example (VBScript):

```
if False = oSite.GetAllowXCRC() then
MsgBox "Disallowed"
```

```
end if
```

All versions

- Function **GetAMParams()** As Object

For an example using this method, refer to the Windows Authentication script in [Script Examples](#).

Retrieves authentication manager (AM) parameters.

```
HRESULT GetAMParams([out,retval] IDispatch** pVal)
```

Example (VBScript):

```
Set oAMParams = oSite.GetAMParams()
```

EFT v6.3 and later

- Function **GetAS2Transactions**(vbShowSuccesses As Boolean, vbShowFailures As Boolean, vbShowInProgress As Boolean, bstrMessageIDFilter As String, bstrFileNameFilter As String, lLastDays As Long) As Variant

Retrieves AS2 transactions.

```
HRESULT GetAS2Transactions(
[in] VARIANT_BOOL vbShowSuccesses,
[in] VARIANT_BOOL vbShowFailures,
[in] VARIANT_BOOL vbShowInProgress,
[in] BSTR bstrMessageIDFilter,
[in] BSTR bstrFileNameFilter,
[in] LONG lLastDays,
[out,retval] VARIANT* pTransInfo);
```

Example (VBScript):

```
For Each trans In oSite.GetAS2Transactions(True,
      True, True, "*", "*", 7)
MsgBox "Transaction id: " & trans.TransactionID
Next
```

EFT v6.3 and later

- Function [GetAuthManagerID\(\)](#) As Long

Retrieves the Site's authentication manager ID (GS Auth, NT/AD, or ODBC).

```
HRESULT GetAuthManagerID([out, retval] long *prop);
```

All versions

- Function **GetAwaitingCertsPath()** As String

Retrieves the Awaiting Certificate file path.

```
HRESULT GetAwaitingCertsPath([out, retval] BSTR *prop);
```

All versions

- Function **GetBlankPermission**(bstrFolder As String, bstrClient As String) As [Permission](#)

NOTE: After you have defined the permissions, you must use the SetPermisson method to apply the changes you have made.

Given Folder and Client creates a blank permission.

```
HRESULT GetBlankPermission(  
[in] BSTR bstrFolder,  
[in] BSTR bstrClient,  
[out, retval] IPermission **pPermission);
```

All versions

- Function [GetCertFilePath\(\)](#) As String

Retrieves the Site certificate file path.

```
HRESULT GetCertFilePath([out, retval] BSTR *prop);
```

All versions

- Function [GetCheckClientCert\(\)](#) As Boolean

Determines if the server requires certificates from users connecting via implicit SSL. If the Server requires certificates, the GetCheckClientCert method returns a value of TRUE.

```
HRESULT GetCheckClientCert([out, retval] VARIANT_BOOL  
*prop);
```

All versions

- Method **GetContentIntegrityControlProfile**

Get CIC profile by name

```
HRESULT GetContentIntegrityControlProfile(
    [in] BSTR profileName,
    [out, retval] long *profileIndex);
```

EFT v8

- Function **GetClientDataField**(ID As [ClientDataField](#)) As Object

Retrieve client data field object ([ICIClientDataField](#))

```
HRESULT GetClientDataField(
    [in] ClientDataField id,
    [out, retval] IDispatch** pdispField);
```

Example (VBScript):

```
set email = oSite.GetClientDataField(ClientDataFieldId_
Email)
wscript.echo
    "=== Email data field ==="
wscript.echo
    "Display name : " & email.DisplayName
wscript.echo
    "Default value : " & email.DefaultValue
wscript.echo
    "IsPersonalData : " & email.IsPersonalData
wscript.echo
    "IsModifiable : " & email.IsModifiable
wscript.echo
    "IsEnabled : " & email.IsEnabled
```

See also [SetClientDataField](#)

EFT v8 and later

- Function **GetCommands**() As Variant
- Retrieves an array of command names.

```
HRESULT GetCommands([out, retval] VARIANT *aCommands);
```

All versions

- Function **GetCommandSettings**(bstrCommand As String) As [CICommandSettings](#)

Given a command name retrieves its settings.

```
HRESULT GetCommandSettings(
    [in] BSTR bstrCommand,
    [out, retval] ICICommandSettings **prop);
```

All versions

- Function [GetComplexPasswordSettings\(\)](#) As Object

Retrieve complex password settings

```
HRESULT GetComplexPasswordSettings(
    [out, retval] IDispatch** ppIComplexPasswordSettings);
```

Example (VBScript):

```
Set oPwdSettings = oSite.GetComplexPasswordSettings()
```

EFT v6.3 and later

- Function [GetConnectedCount\(\)](#) As Long

Retrieves number of users connected to the Site.

```
HRESULT GetConnectedCount([out, retval] long* prop);
```

All versions

- Function **GetConnectedUsers()** As Variant

Retrieves an array of connected users.

```
HRESULT GetConnectedUsers([out, retval] VARIANT* pVal);
```

Example (VBScript):

```
dim arConnectedUsers: oSite.GetConnectedUsers()
For iCount = LBound(arConnectedUsers) To UBound
    (arConnectedUsers)
    WScript.Echo arConnectedUsers(iCount)
Next
```

EFT v4.3.4 and later

- Function **GetConnectionProfileParams**(IIndex As Long) As Object

```
HRESULT GetConnectionProfileParams (
    [in] LONG lIndex,
    [out, retval] IDispatch** ppdispParams);
```

EFT v7.2 and later

- Function **GetConnectionProfilesCount()** As Long

```
HRESULT GetConnectionProfilesCount (
    [out, retval] LONG* plValue);
```

EFT v7.2 and later

- Function [GetDownloadCount\(\)](#) As Long

Retrieves the number of active downloads.

```
HRESULT GetDownloadCount([out, retval] long *pCnt);
```

All versions

- Function **GetDownloadSpeed()** As Long

Retrieves the download speed of a Site.

```
HRESULT GetDownloadSpeed([out, retval] long *pSpeed);
```

All versions

- Function **GetFolderList(bstrFolderAlias As String) As String**

For an example of using this method, refer to the VFS script in [Script Examples](#).

Given a folder alias, retrieves a list of its subfolders (physical and virtual) separated with ODOA (line return).

```
HRESULT GetFolderList (
    [in] BSTR bstrFolderAlias,
    [out, retval] BSTR *prop);
```

Each folder that appears in the folder list MIGHT have a trailing character to indicate something special about that folder.

The colon character (:) is used to denote that a folder is EFS encrypted. If you need to use the subfolder name in your script or program, parse out the colon character.

The double quotation mark (") is used to denote that the folder has subfolders. If you need to use the subfolder name in your script or program, parse out the double quotation mark.

For example, `CISite.GetFolderList( "/" )` might return something like:

```
:Bin
Pub
Usr"
```

Where:

The colon : in front of Bin indicates that the folder is EFS encrypted.

The quotation mark " at the end of Usr indicates that the folder has subfolders.

The absence of a colon or quotation mark after Pub indicates that it is not EFS encrypted and does not have subfolders.

Example (C#):

```
string folderList = site.GetFolderList(VFAliasRoot);
```

All versions

- Function **GetFolderPermissions**(bstrFolder As String) As Variant

See also [ICIPermission - VFS Permission Interface](#).

For an example of using this method, refer to the VFS script in [Script Examples](#).

Retrieves an array of folder permissions (including inherited).

```
HRESULT GetFolderPermissions(
[in] BSTR bstrFolder,
[out, retval] VARIANT *aPermissions);
```

Example (VBScript):

```
Dim arPermissions: arPermissions = oSite.GetFolderPermissions("/usr/test")
```

```
For iCount = LBound(arPermissions) To UBound(arPermissions)
Set oPermission = arPermissions(iCount)
WScript.Echo oPermission.Folder & " - " & oPermission.Client
Next
```


Example (C#):

```

Permission[] folderPermissions = (Permission
[])site.GetFolderPermissions(folderPath);
foreach (Permission perm in (object
[])site.GetFolderPermissions(folderPath))
{
    Console.WriteLine(" " + perm.Client.PadRight(16, ' '));
    Console.WriteLine((perm.IsGroup) ? "group " : "user ");
    Console.WriteLine((perm.IsInherited) ? "Yes " : "No ");
    Console.WriteLine(perm.InheritedFrom);
    Console.WriteLine();
}

```

All versions

- Function [GetFTPAccess\(\)](#) As Boolean

Returns TRUE if FTP access is enabled.

```

HRESULT GetFTPAccess([out, retval] VARIANT_BOOL *prop);

```

True = Access enabled; False = Access disabled

All versions

- Function **GetGDPRSettings()** As Object

Retrieve/set GDPR settings ([ICIGDPRSettings](#)).

```

HRESULT GetGDPRSettings([out, retval] IDispatch
**pdispParams);
HRESULT SetGDPRSettings([in]
    IDispatch *pdispParams);

```

Example:

```

set gdpr = oSite.GetGDPRSettings

```

See also [SetGDPRSettings](#)

EFT v8

- Function [GetHTTPAccess\(\)](#) As Boolean

Returns TRUE if HTTP access is enabled.

```

HRESULT GetHTTPAccess([out, retval] VARIANT_BOOL *prop);

```

True = Access enabled; False = Access disabled

EFT v4.3.4 and later

- Function [GetHTTPPort\(\)](#) As Long

Retrieves the HTTP port number of the Site.

```
HRESULT GetHTTPPort([out, retval] long *prop);
```

EFT v4.3.4 and later

- Function [GetHTTPSAccess\(\)](#) As Boolean

Returns TRUE if HTTPS access is enabled.

```
HRESULT GetHTTPSAccess([out, retval] VARIANT_BOOL *prop);
```

True = Access enabled; False = Access disabled

EFT v4.3.4 and later

- Function [GetHTTPSPort\(\)](#) As Long

Retrieves the HTTP port number of the Site.

```
HRESULT GetHTTPSPort([out, retval] long *prop);
```

EFT v4.3.4 and later

- Function [GetIP\(\)](#) As Long

Retrieves the index of the Site Listen IP

```
HRESULT GetIP([out, retval] long *prop);
```

All versions

- Function [GetIPAccessRules\(\)](#) As Variant

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Retrieves an array of allowed IP masks.

```
HRESULT GetIPAccessRules([out, retval] VARIANT *aMasks);
```

Example (VBScript):

```
rules = oSite.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
WScript.echo "AutoBan"
End if
If key.type = 1 Then
WScript.echo "Manual " + key.address
End if
Next
```

EFT v6.3 and later

- Function [GetKeyFilePath\(\)](#) As String

Retrieves the Site SSL private key file path.

```
HRESULT GetKeyFilePath([out, retval] BSTR *prop);
```

All versions

- Function GetPCIDSSComplianceStatus(plPassed As Long, plWarning As Long, plComControl As Long, plFailed As Long)

```
HRESULT GetPCIDSSComplianceStatus(
[out] LONG* plPassed,
[out] LONG* plWarning,
[out] LONG* plComControl,
[out] LONG* plFailed);
```

EFT v7.3 and later

- Function [GetPassPhrase\(\)](#) As String

Retrieves the Site SSL Private key passphrase.

```
HRESULT GetPassPhrase([out, retval] BSTR *prop);
```

All versions

- Function GetPendingCertificateInfo(IID As Long) As [CICertInfo](#)

Given an ID, retrieves a pending certificate info.

```
HRESULT GetPendingCertificateInfo(
    [in] long lID,
    [out, retval] ICICertInfo **prop);
```

All versions

- Function **GetPendingCertificates()** As Variant

Retrieves a list of pending certificate IDs.

```
HRESULT GetPendingCertificates([out, retval] VARIANT
    *aCerts);
```

All versions

- Function [GetPermissionGroupList](#)(bstrGroup As String) As Variant

Retrieves an array of permission groups.

```
HRESULT GetPermissionGroupList(
    [in] BSTR bstrGroup,
    [out, retval] VARIANT *aUsers);
```

Example (VBScript):

```
Dim arUsers: arUsers = oSite.GetPermissionGroupList
("Administrative")
For iCount = LBound(arUsers) To UBound(arUsers)
    WScript.Echo arUsers(iCount)
Next
```

All versions

- Function [GetPermissionGroups](#)() As Variant

Retrieves an array of users in a permission group. (Permission Groups control user access to files and folders.)

```
HRESULT GetPermissionGroups([out, retval] VARIANT *aGroups);
```

All versions

- Function [GetPermissionGroupsOfUser](#)(bstrUser As String) As Variant

Given a user account, retrieve its permission groups.

```
HRESULT GetPermissionGroupsOfUser (
    [in] BSTR bstrUser,
    [out, retval] VARIANT *aGroups);
```

All versions

- Function **GetPermPathsList**(bstrOptions As String) As String

Returns a list of all paths that have a unique set of permissions. (All other paths, which are not included in the set returned by this call, inherit the permissions as set in their parent.)

```
HRESULT GetPermPathsList (
    [in] BSTR bstrOptions,
    [out, retval] BSTR *prop);
```

Supported value for bStrOptions is -do (decorate orphans)—When specified, a folder that exists in the Virtual File System (VFS), but whose physical path does not exist, will be decorated with a trailing asterisk (*).

EFT v6.3.1 and later

- Function **GetPhysicalPath**(bstrFolderAlias As String) As String

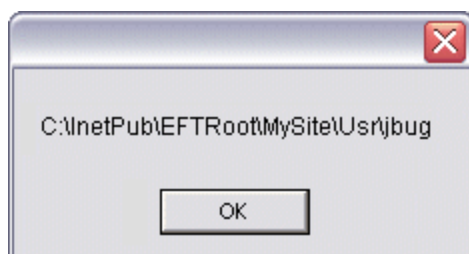
Transforms a virtual path to physical one.

```
HRESULT GetPhysicalPath (
    [in] BSTR bstrFolderAlias,
    [out, retval] BSTR *pbstrPhysicalPath);
```

Example (VBScript):

```
Msgbox Site.GetPhysicalPath("/Usr")
```

Returns a message box displaying the physical path, such as:



EFT v5.2.5 and later

- Function [GetPort](#)() As Long

Retrieves the port number of the Site.

```
HRESULT GetPort([out, retval] long *prop);
```

All versions

- Function **GetPrivacyPolicy()** As Object ()

Retrieves the Privacy Policy settings ([ICIPrivacyPolicy](#))

```
HRESULT GetPrivacyPolicy([out, retval] IDispatch  
**pdispParams);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
```

See also [SetPrivacyPolicy](#)

EFT v8 and later

- Function [GetResetPasswordSettings\(\)](#) As Object

Use ICISite::GetResetPasswordSettings and [ICISite::SetResetPasswordSettings](#) to retrieve and set the password settings for a Site.

Retrieves the password reset settings.

```
HRESULT GetResetPasswordSettings(  
[out, retval] ICIResetPasswordSettings**  
ppIResetPasswordSettings); (v6.3 - 6.3.3)  
[out, retval] IDispatch** ppIResetPasswordSettings); (v6.3.8  
and later)
```

Example (VBScript):

```
Set oPwdSettings = oSite.GetResetPasswordSettings()
```

EFT v6.3 and later

- Function [GetRootFolder\(\)](#) As String

Retrieves the root folder for an individual site.

```
HRESULT GetRootFolder([out, retval] BSTR *prop);
```

All versions

- Function **GetSettingsLevels()** As Variant

For an example using this method, refer to the UserSettingsTemplates script in [Script Examples](#).

Retrieves an array of User Settings Templates.

```
HRESULT GetSettingsLevels([out, retval] VARIANT *aLevels);
```

Example (C#):

```
CIClientSettings userSettings = selectedSite.GetUserSettings(
    userName);
object settingsLevels = selectedSite.GetSettingsLevels();
```

All versions

- Function **GetSettingsLevelSettings(bstrGroup As String)** As [CIClientSettings](#)

Given a Settings Template name retrieves its settings.

```
HRESULT GetSettingsLevelSettings(
    [in] BSTR bstrGroup,
    [out, retval] ICIClientSettings **prop);
```

All versions

- Function **GetSettingsLevelUsers(bstrGroup As String)** As Variant

For an example using this method, refer to the UserSettingsTemplates script in [Script Examples](#).

Given a Settings Template, retrieves an array of its users.

```
HRESULT GetSettingsLevelUsers(
    [in] BSTR bstrGroup,
    [out, retval] VARIANT *aUsers);
```

Example (C#):

```
CIClientSettings userSettings = selectedSite.GetUserSettings(
    user);
object settingsLevels = selectedSite.GetSettingsLevels();
```

All versions

- Function **GetSFTPAccess()** As Boolean

Returns TRUE if SFTP access is enabled.

```
HRESULT GetSFTPAccess([out, retval] VARIANT_BOOL* pbVal);
```

EFT v6.4 and later

- Function [GetSSHKeyFilePath\(\)](#) As String

Retrieves SSH key file path.

```
HRESULT GetSSHKeyFilePath([out, retval] BSTR *prop);
```

EFT v4.3.4 - 7.4.x (deprecated in EFT v8)

- Function **GetSSHKeyForSFTP()** As Long

Retrieves Site's SSH key by key's index in SSHKeys array.

```
HRESULT GetSSHKeyForSFTP([out, retval] LONG* keyIndex);
```

Example (VBScript):

```
keyIndex = oSite.GetSSHKeyForSFTP()
```

See also [SetSSHKeyForSFTP](#)

EFT v8 and later

- Function [GetSSLAuth\(\)](#) As Boolean

Returns TRUE if SSL authentication is enabled.

```
HRESULT GetSSLAuth([out, retval] VARIANT_BOOL *prop);
```

True = Enabled; False = Disabled

All versions

- Function [GetSSLImp\(\)](#) As Boolean

Returns TRUE if SSL implicit is enabled.

```
HRESULT GetSSLImp([out, retval] VARIANT_BOOL *prop);
```

True = Enabled; False = Disabled

All versions

- Function [GetStartTime\(\)](#) As String

Retrieves the time the Site was started.

```
HRESULT GetStartTime([out, retval] BSTR *bstrStartTime);
```

All versions

- Function **GetTermsOfService()** As Object

Retrieves Terms of Service settings ([ICITermsOfService](#)).

```
HRESULT GetTermsOfService([out, retval] IDispatch  
**pdispParams)
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
```

See also [SetTermsOfService](#)

EFT v8 and later

- Function **GetTransactionalWorkspacesIDs(bstrParentId As String) As Variant**

```
HRESULT GetTransactionalWorkspacesIDs(  
[in] BSTR bstrParentId,  
[out, retval] VARIANT* pVal);
```

EFT v7.4.2 and later

- Function **GetTrustedCertificateInfo(IID As Long) As [CICertInfo](#)**

For an example using this method, refer to the GetCertInfo script in [Script Examples](#).

Given an ID, retrieves a trusted certificate info.

```
HRESULT GetTrustedCertificateInfo(  
[in] long IID,  
[out, retval] ICICertInfo **prop);
```

All versions

- Function **GetTrustedCertificates()** As Variant

For an example using this method, refer to the GetCertInfo script in [Script Examples](#).

Retrieves a list of trusted certificate IDs.

```
HRESULT GetTrustedCertificates(
    [out, retval] VARIANT *aCerts);
```

All versions

- Function **GetTrustedCertsPath()** As String

Retrieves the Trusted Certificate file path.

```
HRESULT GetTrustedCertsPath([out, retval] BSTR *prop);
```

All versions

- Function [GetUploadCount\(\)](#) As Long

Retrieves the number of active uploads.

```
HRESULT GetUploadCount([out, retval] long *pCnt);
```

All versions

- Function **GetUploadForm**(formName As String) As Long

Retrieves an Upload Form from a Site. (Use [UploadFormsList](#) to see a list of Upload Forms.)

```
HRESULT GetUploadForm(
    [in] BSTR formName,
    [out, retval] long *formIndex);
```

EFT v8.0.4

- Function **GetUploadSpeed()** As Long

Retrieves the upload speed of a site.

```
HRESULT GetUploadSpeed([out, retval] long *pSpeed);
```

All versions

- Function **GetUsers()** As Variant

For an example using this method, refer to the ServerIPBanEntry script in [Script Examples](#).

Retrieves an array of users.

```
HRESULT GetUsers([out, retval] VARIANT *aUsers);
```

Example (C#):

```
object[] users = (object[])selectedSite.GetUsers();
```

All versions

- Function **GetUserSettings**(bstrUser As String) As [CIClientSettings](#)

For an example using this method, refer to the GetCertInfo, ServerIPBanEntry, ConfigureUser, or UserSettingsTemplates script in [Script Examples](#).

Given a username, retrieves its settings.

```
HRESULT GetUserSettings(
    [in] BSTR bstrUser,
    [out, retval] ICIClientSettings **prop);
```

Example (VBScript):

```
Site.GetUserSettings("kmarx")
```

GetUserSettings is case *insensitive*. That is, in many non-windows environments "KMarx" and "kmarx" are two different usernames and case matters. With this method, "KMarx", "KMARX", and "kmarx" are all the same user account.

Examples (C#):

```
ICIClientSettings client = selectedSite.GetUserSettings
((string)users[0]);
CIClientSettings userSettings = selectedSite.GetUserSettings
(userName);
object settingsLevels = selectedSite.GetSettingsLevels();
```

All versions

- Function **GetUserSettingsLevel**(bstrUser As String)

Given a user, retrieves its settings template.

```
HRESULT GetUserSettingsLevel(
    [in] BSTR bzUser,
    [out, retval] BSTR* pbzSettingsLevel);
```

EFT v6.4 and later

- Function **GetVirtualFolderList**(bstrOptions As String) As String

Returns a list of all paths in the VFS that represent virtual folders. Follow with [GetPhysicalPath](#) to find out which physical folder the virtual folder represents.

```
HRESULT GetVirtualFolderList(
    [in] BSTR bstrOptions,
    [out, retval] BSTR *prop);
```

(Added in v8) Calls with parameter `site.GetVirtualFolderList("/some/site/folder/path")` - if `/some/site/folder/path` folder exists in the site - returns virtual folders that are inside the folder `/some/site/folder/path` recursively.

e.g. call `site.GetVirtualFolderList("/usr/testuser")` might return:

```
/usr/testuser/virtual_folder0
/usr/testuser/physical_folder/virtual_folder1
/usr/testuser/physical_folder/another_physical_
folder/virtual_folder2
```

All others calls should fail.

`bstrOptions` is included to support the possible addition of functionality in later versions. Currently, this option is ignored.

EFT v6.3.1 and later

- Function **GetWorkspace**(`bstrGUID As String`) As Object

Given [Workspace GUID](#), retrieves the Workspaces

```
HRESULT GetWorkspace(
    [in] BSTR bstrGUID,
    [out, retval] IDispatch** ppdispWorkspace);
```

EFT v7.1 and later

- Function **GetWorkspacesDropOffAddressBook**() As Object and Sub **SetWorkspacesDropOffAddressBook**(`pdispParams As Object`)

```
HRESULT GetWorkspacesDropOffAddressBook([out, retval]
    IDispatch **pdispParams);
HRESULT SetWorkspacesDropOffAddressBook([in] IDispatch
    *pdispParams);
```

EFT v7.4.2 and later

- Sub **GetWorkspacesDropOffMaxDuration**(pValue, pUnits) and Sub **SetWorkspacesDropOffMaxDuration**(TimeUnits As TimeUnits, val As DWord)

```
HRESULT GetWorkspacesDropOffMaxDuration(
    [out] VARIANT *pValue,
    [out] VARIANT *pUnits);
HRESULT SetWorkspacesDropOffMaxDuration(
    [in] TimeUnits timeUnits,
    [in] ULONG val);
```

EFT v7.4.2 and later

- Sub **GetWorkspacesDropOffMaxMessageSize**(pValue, pUnits) and Sub **SetWorkspacesDropOffMaxMessageSize**(SizeUnits As SizeUnits, val As DWord)

```
HRESULT GetWorkspacesDropOffMaxMessageSize(
    [out] VARIANT *pValue,
    [out] VARIANT *pUnits);
HRESULT SetWorkspacesDropOffMaxMessageSize(
    [in] SizeUnits sizeUnits,
    [in] ULONG val);
```

EFT v7.4.2 and later

- Function **GetWorkspaceGUID**(bstrFolder As String) As String

Given Workspace folder, retrieves the Workspace GUID

```
HRESULT GetWorkspaceGUID(
    [in] BSTR bstrFolder,
    [out, retval] BSTR* bstrGUID);
```

EFT v7.1 and later

- Sub **GetWorkspaceMaxDuration**(pTimeUnits As TimeUnits, pVal As DWord)

```
HRESULT GetWorkspaceMaxDuration(
    [out] TimeUnits* pTimeUnits,
    [out] ULONG* pVal);
```

See also [TimeUnits](#) enum and [SetWorkspaceMaxDuration](#).

EFT v7.3.3 and later

- Sub **GetWorkspacesOAIMaxDuration**(pTimeUnits As TimeUnits, pVal As DWord)

```
HRESULT GetWorkspacesOAIMaxDuration(
    [out] TimeUnits* pTimeUnits,
    [out] ULONG* pVal);
```

See also [TimeUnits](#) enum and [SetWorkspacesOAIMaxDuration](#).

EFT v7.3.3 and later

- Sub **GetWorkspacesOAIRetainFilesFor**

```
HRESULT GetWorkspacesOAIRetainFilesFor([out]
    TimeUnits* pTimeUnits, [out] ULONG* pVal);
```

See also [TimeUnits](#) enum and [SetWorkspacesOAIRetainFilesFor](#).

EFT v8 and later

- Function **GuestSettingsLevels()** As Variant

```
HRESULT GuestSettingsLevel([out, retval] BSTR* bstrName);
HRESULT GuestSettingsLevel([in] BSTR bstrName);
```

EFT v7.4 and later

See also [CreateGuestSettingsLevel](#).

- Property **GUID** As String

Returns associated GUID based on the database filename.

Read only

```
HRESULT GUID([out, retval] BSTR *pVal);
```

To get the GUID you can do the following (change the ConnectEx and Site.Name info to match your Site info):

```
dim template
dim oServer, oSites, oSite
set oServer = CreateObject ("SFTPCOMInterface.CIServer")
oServer.ConnectEx "server", 1100, 0, "user", "pass"
set oSites = oServer.Sites()
for i = 0 to oSites.Count() -1
set oSite = oSites.Item (i)
if oSite.Name = "MySite" Then
exit for
end if
```

```
next
guid = oSite.GUID
wscript.echo guid
```

EFT v7.3.9 and 7.4.6 and later

- Property [HasMaxConcurrentLogins](#) As Boolean

Retrieves or specifies whether the site has maximum concurrent logins enabled.

```
HRESULT HasMaxConcurrentLogins(
[out, retval] VARIANT_BOOL *pVal);
HRESULT HasMaxConcurrentLogins(
[in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

EFT v6.3 and later

- Property [HasMaxConnectionsPerAccount](#) As Boolean

Retrieves or specifies whether the Site has a maximum concurrent socket connections enabled.

```
HRESULT HasMaxConnectionsPerAccount(
[out, retval] VARIANT_BOOL *pVal);
HRESULT HasMaxConnectionsPerAccount(
[in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

All versions

- Property [HasMaxIPPerAccount](#) As Boolean

Retrieves or specifies whether the Site has maximum connections from same IP enabled.

```
HRESULT HasMaxIPPerAccount(
[out, retval] VARIANT_BOOL *pVal);
HRESULT HasMaxIPPerAccount(
[in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

All versions

- Property [HasMaxSpeed](#) As Boolean

Retrieves or specifies whether the Site has a maximum transfer speed enabled.

```
HRESULT HasMaxSpeed(
    [out, retval] VARIANT_BOOL *pVal);
HRESULT HasMaxSpeed(
    [in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

All versions

- Property [HasMaxUsers](#) As Boolean

Retrieves or specifies whether the Site has the maximum connections per user enabled.

```
HRESULT HasMaxUsers(
    [out, retval] VARIANT_BOOL *pVal);
HRESULT HasMaxUsers(
    [in] VARIANT_BOOL newVal)
```

True=Enabled; False=Disabled

All versions

- Property [HTTPDomain](#) As String

Retrieves or specifies the Site's domain name for HTTP access. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT HTTPDomain(
    [out, retval] BSTR* pVal);
HRESULT HTTPDomain(
    [in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Current http domain: " & oSite.HTTPDomain
```

Specifying:

```
oSite.HTTPDomain = "localhost"
```


EFT v6.3 and later

- Property **ID** As Long

Retrieves read-only version of the Site ID.

```
HRESULT ID([out, retval] long *pVal);
```

All versions

- Sub **ImportEventRules**(bstrSrcFile As String)

See also [ExportEventRules](#)

```
HRESULT ImportEventRules([in] BSTR bstrDstFile);
```

EFT v7.2 and later

- Sub **ImportPGPKey**(bstrSrcFile As String)

Import specified PGP key from file path.

```
HRESULT ImportPGPKey([in] BSTR bstrKeyPath, [out, retval]
BSTR* pNewKeyID);
```

Example (VBScript):

```
keyId = oSite.ImportPGPKey("C:\PGP\pgpkey.asc")
```

See also [ExportPGPKey](#), [CreatePGPKey](#), [RemovePGPKey](#)

EFT v8 and later

- Function **ImportSSHPrivateKey**(Name As String, priv, pass As String) As Long

Import SSH key to Site by key's body (array of bytes), return key's index in SSHKeys array. If specified name exists, the key name will be imported.

```
HRESULT ImportSSHPrivateKey(
[in] BSTR name,
[in] VARIANT priv,
[in] BSTR pass,
[out, retval] LONG* keyIndex);
```

Example (VBScript):

```
keyIndex = oSite.ImportSSHPrivateKey("myKey", priv,
"password")
```

See also [ExportSSHPrivateKey](#)

EFT v8 and later

- Function **ImportSSHPublicKey**(Name As String, pub) As Long

Imports public part of SSH key to site by public key's body (array of bytes), returns key's index in SSHKeys array. If specified name exists, the key name will be imported.

```
HRESULT ImportSSHPublicKey(
    [in] BSTR name,
    [in] VARIANT pub,
    [out, retval] LONG* keyIndex);
```

Example (VBScript):

```
keyIndex = oSite.ImportSSHPrivateKey("myKey", priv,
    "password")
```

See also [ExportSSHPublicKey](#)

EFT v8 and later

- Function **ImportTrustedCertificate**(bstrCertPath As String) As Long

Imports a trusted certificate.

```
HRESULT ImportTrustedCertificate(
    [in] BSTR bstrCertPath,
    [out, retval] long * lID);
```

In EFT v8 and later, you must specify the path to the server's local certificate. Therefore, the remote server where this call is being made from should have the certificate available locally.

For example:

```
$EFT = New-Object -ComObject 'SFTPCOMInterface.CIServer'
$eftAdminUserName = 'administrator'
$eftPassword = '12345678'
$eftServer = '192.168.107.155'
$eftPort = '1100'
$EFT.Connect($eftServer, $eftPort, $eftAdminUserName,
    $eftPassword)
$Sites = $EFT.Sites()
$site = $Sites.SiteByID(1)
```

```
$site.ImportTrustedCertificate("C:\EFTSites\GSSite_Download_
UserCert.crt")
```

All versions

- Property [InactiveAccountsMonitoring](#) As Boolean

For an example using this property, refer to the ConfigureSite script in [Script Examples](#).

Retrieves or specifies whether inactive accounts are monitored.

```
HRESULT InactiveAccountsMonitoring(
[out, retval] VARIANT_BOOL* pVal);
HRESULT InactiveAccountsMonitoring(
[in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
if False = oSite.InactiveAccountsMonitoring then
MsgBox "Disabled"
End If
```

Specifying:

```
oSite.InactiveAccountsMonitoring = True
```

Example (C#):

```
selectedSite.InactiveAccountsMonitoring = true;
```

EFT v6.3 and later

- Property [InvalidAttemptsPeriod](#) As Long

Retrieves or specifies the length of time for "Lock out / disable account after *N* invalid login attempts during time period" option.

```
HRESULT InvalidAttemptsPeriod(
[out, retval] long *pVal);
HRESULT InvalidAttemptsPeriod(
[in] long newVal);
```

Example (VBScript):

```
Site.InvalidAttemptsPeriod = 30
```

EFT v5.1.1 and later

- Property **InvalidLoginAttemptsCountToBanIP** As Long

Retrieves or specifies the number of login attempts after which to ban the IP address.

```
HRESULT InvalidLoginAttemptsCountToBanIP(
[out, retval] long* pVal);
HRESULT InvalidLoginAttemptsCountToBanIP(
[in] long newVal);
```

EFT v6.4 and later

- Property [IPAccessAllowedDefault](#) As Boolean

Retrieves or specifies whether IP access is allowed by default.

```
HRESULT IPAccessAllowedDefault(
[out, retval] VARIANT_BOOL *pVal);
HRESULT IPAccessAllowedDefault(
[in] VARIANT_BOOL newVal);
```

True = Allowed; False = Denied

All versions

- Function **IsEncrypted**(bstrFolder As String) As Boolean

(Was "IsFolderEncrypted" in v6.2.31)

For an example of using this method, refer to the VFS script in [Script Examples](#).

Retrieves or specifies whether a specified folder is encrypted.

```
HRESULT IsEncrypted(
[in] BSTR bstrFolder,
[out,retval] VARIANT_BOOL* pbResult);
```

Example (VBScript):

```
if False = oSite.IsEncrypted("/some_folder") then
MsgBox "Not encrypted."
end if
```

Example (C#):

```
Console.Write(" Is encrypted with EFS: ");
Console.WriteLine(site.IsEncrypted(folderPath) ?
"Yes" : "No");
```

EFT v6.3 and later

- Function **IsFolderInherited**(bstrFolder As String) As Boolean

Retrieves or specifies whether a specified folder is inherited.

```
HRESULT IsFolderInherited(
[in] BSTR bstrFolder,
[out,retval] VARIANT_BOOL* vbResult);
```

Example (VBScript):

```
if False = oSite.IsFolderInherited("/some_folder") then
MsgBox "Not inherited"
end if
```

EFT v6.2.18 and later

- Function **IsFolderVirtual**(bstrFolderAlias As String) As Boolean

For an example of using this method, refer to the VFS script in [Script Examples](#).

Retrieves or specifies whether a specified folder is a virtual folder.

```
HRESULT IsFolderVirtual(
[in] BSTR bstrFolderAlias,
[out,retval] VARIANT_BOOL* pbRes);
```

Examples (VBScript):

```
if True = oSite.IsFolderVirtual("/some_folder") then
MsgBox "/some_folder is virtual."
end if
```

Example (C#):

```
Console.Write(" Is virtual: ");
Console.WriteLine(site.IsFolderVirtual(folderPath) ? "Yes" :
"No");
```

EFT v6.2.18 and later

- Property [IsStarted](#) As Boolean

To start or stop a Site via the API, refer to:

[Start](#)

[Stop](#)

Retrieves TRUE if the Site is started, FALSE if it is not started.

```
HRESULT IsStarted(
    [out, retval] VARIANT_BOOL *pVal);
```

Example (VBScript):

```
If Not Site.IsStarted Then
    Site.Start
End If
```

All versions

- Property **IsSFTPSettingsInherited** As Boolean

Used to specify whether SFTP settings are inherited by the Settings Template and users on a Site.

```
HRESULT IsSFTPSettingsInherited(
    [in] VARIANT_BOOL newVal);
HRESULT IsSFTPSettingsInherited(
    [out, retval] VARIANT_BOOL* pVal);
```

EFT v8.0.5

- Property **IsSSLSettingsInherited** As Boolean

Used to specify whether SSL settings are inherited by the Settings Template and users on a Site.

```
HRESULT IsSSLSettingsInherited(
    [in] VARIANT_BOOL newVal);
HRESULT IsSSLSettingsInherited(
    [out, retval] VARIANT_BOOL* pVal);
```

EFT v8.0.5

- Function **KickUser**(nUserID As Long) As Boolean

Forcibly logs a user off of the Site.

```
HRESULT KickUser(
    [in] long nUserID,
    [out, retval] VARIANT_BOOL *pSuccess);
```

Example (VBScript):

```
Dim bResult: bResult = oSite.KickUser(1)
If bResult Then
    WScript.Echo "User disconnected successfully."
Else
    WScript.Echo "Failed to disconnect the user."
End If
```

EFT v4.3.4 and later

- Property [LastModifiedBy](#) As String (read only)

Retrieves username of account that last modified the Site.

```
HRESULT LastModifiedBy(
    [out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Last modified by " & CStr(oSite.LastModifiedBy)
```

EFT v6.3 and later

- Property [LastModifiedTime](#) As Date (read only)

Retrieves the date that the Site was last modified.

```
HRESULT LastModifiedTime(
    [out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Last modified time " & CStr(oSite.LastModifiedTime)
```

EFT v6.3 and later

- Property [LimitLoginAttempts](#) As Boolean

Retrieves or specifies whether the number of login attempts is limited.

```
HRESULT LimitLoginAttempts(
    [out, retval] VARIANT_BOOL *pVal);
```

```
HRESULT LimitLoginAttempts(
    [in] VARIANT_BOOL newVal);
```

True=Enabled; False=Disabled

EFT v5.1.1 and later

- Property **ListenIPs** As Variant

Retrieves or specifies the IP addresses on which the server listens.

```
HRESULT ListenIPs(
    [out, retval] VARIANT* pVal);
HRESULT ListenIPs(
    [in] VARIANT newVal);
```

EFT v6.4 and later

- Property [LockoutNotDisable](#) As Boolean

Retrieves or specifies whether account lockout is not disabled.

```
HRESULT LockoutNotDisable(
    [out, retval] VARIANT_BOOL *pVal);
```

```
HRESULT LockoutNotDisable(
    [in] VARIANT_BOOL newVal);
```

True=Lockout is not disabled; False=Lockout is disabled

EFT v5.1.1 and later

- Property [LockoutPeriod](#) As Long

Retrieves or specifies the lockout period value. Values are 30, 60, or 90 minutes.

```
HRESULT LockoutPeriod(
    [out, retval] long *pVal);
HRESULT LockoutPeriod(
    [in] long newVal);
```

EFT v5.1.1 and later

- Property [MaxConcurrentConnections](#) As Long

Retrieves or specifies the maximum concurrent connections value.


```
HRESULT MaxConcurrentConnections([out, retval] long *pVal);
HRESULT MaxConcurrentConnections([in] long newVal);
```

All versions

- Property [MaxConcurrentLogins](#) As Long

Retrieves or specifies the maximum concurrent logins value.

```
HRESULT MaxConcurrentLogins([out, retval] LONG *pVal);
HRESULT MaxConcurrentLogins([in] LONG newVal);
```

Secure Server, All versions ; EFT v6.3 and later

- Property [MaxConnectionsFromSameIP](#) As Long

Retrieves or specifies the maximum connections from the same IP address value.

```
HRESULT MaxConnectionsFromSameIP([out, retval] long *pVal);
HRESULT MaxConnectionsFromSameIP([in] long newVal);
```

All versions

- Property [MaxConnectionsPerUser](#) As Long

Retrieves or specifies the maximum socket connections per user value.

```
HRESULT MaxConnectionsPerUser([out, retval] long *pVal);
HRESULT MaxConnectionsPerUser([in] long newVal);
```

All versions

- Property [MaxInactivePeriod](#) As Long

For an example using this property, refer to the GetCertInfo, ConfigureSite, or ConfigureUser script in [Script Examples](#).

Retrieves or specifies the maximum inactive period value.

```
HRESULT MaxInactivePeriod([out, retval] LONG* pVal);
HRESULT MaxInactivePeriod([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Inactive period is: " & CStr
(oSite.MaxInactivePeriod)
```

Specifying:

```
oSite.MaxInactivePeriod = 90
```

Example (C#):

```
selectedSite.MaxInactivePeriod = 30;
```

EFT v6.3 and later

- Property [MaxInvalidLoginAttempts](#) As Long

Retrieves or specifies the maximum invalid login attempts value.

```
HRESULT MaxInvalidLoginAttempts([out, retval] long *pVal);
HRESULT MaxInvalidLoginAttempts([in] long newVal);
```

EFT v5.1.1 and later

- Property **MaxInvalidLoginAttemptsPeriodToBanIP** As Long

Retrieves or specifies the period during which to count invalid login attempts.

```
HRESULT MaxInvalidLoginAttemptsPeriodToBanIP([out, retval]
long* pVal);
HRESULT MaxInvalidLoginAttemptsPeriodToBanIP([in] long
newVal);
```

EFT v6.4 and later

- Property [MaxTransferSpeed](#) As Long

Retrieves or specifies the maximum transfer speed limit.

```
HRESULT MaxTransferSpeed([out, retval] long *pVal);
HRESULT MaxTransferSpeed([in] long newVal);
```

All versions

- Property **MfaType** As [MfaType](#)

Used to specify MFA type.

```
HRESULT MfaType([out, retval] MfaType* pVal);
HRESULT MfaType([in] MfaType newVal);
```

EFT v8.0.4 and later

- Sub [MoveIPAccessRule](#)(bstrMask As String, bAllow As Boolean, nRulePos As Long)

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Changes allowed or denied IP mask position for the Site.

```
HRESULT MoveIPAccessRule([in] long nRulePosFrom, [in] long
nRulePosTo);
```

Example (VBScript):

```
oSite.MoveIPAccessRule(1, 2)
```

EFT v6.3 and later

- Sub **MoveUserToSettingsLevel**(bstrUser As String, bstrGroup As String)

Moves a user to another Settings Template.

```
HRESULT MoveUserToSettingsLevel([in] BSTR bstrUser, [in] BSTR
bstrGroup);
```

All versions

- Property [MTCAAllowFileCaching](#) As Boolean

Turns on or off file caching for the Mobile Transfer Client.

```
HRESULT MTCAAllowFileCaching([in] VARIANT_BOOL newVal);
HRESULT MTCAAllowFileCaching([out, retval] VARIANT_BOOL*
pVal);
```

EFT v6.5.16 and later

- Property [MTCAAllowFileOpening](#) As Boolean

Turns on or off ability to open files in external apps for the Mobile Transfer Client.

```
HRESULT MTCAAllowFileOpening([in] VARIANT_BOOL newVal);
HRESULT MTCAAllowFileOpening([out, retval] VARIANT_BOOL*
pVal);
```

EFT v6.5.16 and later

- Property [MTCAAllowFileSharing](#) As Boolean

Turns on or off file sharing for the Mobile Transfer Client.

```
HRESULT MTCAAllowFileSharing([in] VARIANT_BOOL newVal);
HRESULT MTCAAllowFileSharing([out, retval] VARIANT_BOOL*
pVal);
```

EFT v6.5.16 and later

- Property [MTCAAllowSavingOffline](#) As Boolean

Turns on or off saving files offline for the Mobile Transfer Client.

```
HRESULT MTCAAllowSavingOffline([in] VARIANT_BOOL newVal);
HRESULT MTCAAllowSavingOffline([out, retval] VARIANT_BOOL*
pVal);
```

EFT v6.5.16 and later

- Property [MTCAAllowSavingPasswords](#) As Boolean

Turns on or off saving Mobile Transfer Client profile passwords on the device for the Mobile Transfer Client

```
HRESULT MTCAAllowSavingPasswords([in] VARIANT_BOOL newVal);
HRESULT MTCAAllowSavingPasswords([out, retval] VARIANT_BOOL*
pVal);
```

EFT v6.5.16 and later

- Property **Name** As String

Retrieves the name of the Site.

```
HRESULT Name([out, retval] BSTR *pVal);
```

All versions

- Property **ODBCSettings** As String

Contains all auth manager settings in the form of string (attributes are separated with semicolons).

```
HRESULT ODBCSettings([out, retval] BSTR *pVal);
HRESULT ODBCSettings([in] BSTR newVal);
```

For example, the logon name attribute for an AD\NTLM site is represented by the "ATTR=N" substring. N = 0, 1, 2, or 3:

```
0 = "NT4 Account Name"
1 = "Display Name"
2 = "User Principal Name"
3 = "Common Name"
```

EFT v4.3.4 and later

- Property **OverrideVFSCredentialsEnable** As Boolean

Retrieves or specifies the number of previous passwords to save in history.

```
HRESULT OverrideVFSCredentialsEnable([out, retval] VARIANT_
BOOL* pVal);
HRESULT OverrideVFSCredentialsEnable([in] VARIANT_BOOL
newVal)
```

EFT v8.0.1 and later

- Property **OverrideVFSCredentialsLogin** As String

Retrieves or specifies the number of previous passwords to save in history.

```
HRESULT OverrideVFSCredentialsLogin([out, retval] BSTR
*pVal);
HRESULT OverrideVFSCredentialsLogin([in] BSTR newVal);
```

EFT v8.0.1 and later

- Property **OverrideVFSCredentialsPasswords** As String

Retrieves or specifies the number of previous passwords to save in history.

```
HRESULT OverrideVFSCredentialsPassword([out, retval] BSTR
*pVal);
HRESULT OverrideVFSCredentialsPassword([in] BSTR newVal);
```

EFT v8.0.1 and later

- Property [PasswordHistorySize](#) As Long

Retrieves or specifies the number of previous passwords to save in history.

```
HRESULT PasswordHistorySize([out, retval] LONG* pVal);
HRESULT PasswordHistorySize([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Password history size is: " & CStr
(oSite.PasswordHistorySize)
```

Specifying:

```
oSite.PasswordHistorySize = 5
```

EFT v6.3 and later

- Property [PASVListenIP](#) As Long

Retrieves or specifies the PASV listening IP address.

```
HRESULT PASVListenIP([out, retval] long *pVal);
HRESULT PASVListenIP([in] long newVal);
```

All versions

- Property [PASVPortMax](#) As Long

Retrieves or specifies the maximum end of the PASV listening port range.

```
HRESULT PASVPortMax([out, retval] long *pVal);
HRESULT PASVPortMax([in] long newVal);
```

All versions

- Property [PASVPortMin](#) As Long

Retrieves or specifies the minimum end of the PASV listening port range.

```
HRESULT PASVPortMin([out, retval] long *pVal);
HRESULT PASVPortMin([in] long newVal);
```

All versions

- Property **PGPEmailExpirationRecipientsList** As String

```
HRESULT PGPEmailExpirationRecipientsList([out, retval] BSTR* pVal);
HRESULT PGPEmailExpirationRecipientsList([in] BSTR newVal);
```

EFT v8.0.6 and later

- Property **PGPSendEmailExpirationCopyToAssociatedUser** As Boolean

```
HRESULT PGPSendEmailExpirationCopyToAssociatedUser([out,
retval] VARIANT_BOOL* pVal);
HRESULT PGPSendEmailExpirationCopyToAssociatedUser([in]
VARIANT_BOOL newVal);
```

EFT v8.0.6 and later

- Property **PGPSendEmailPriorExpiration** As Boolean

```
HRESULT PGPSendEmailPriorExpiration([out, retval] VARIANT_
BOOL* pVal);
HRESULT PGPSendEmailPriorExpiration([in] VARIANT_BOOL
newVal);
```

EFT v8.0.6 and later

- Property **PGPSendEmailUponExpiration** As Boolean

```
HRESULT PGPSendEmailUponExpiration([out, retval] VARIANT_
BOOL* pVal);
HRESULT PGPSendEmailUponExpiration([in] VARIANT_BOOL
newVal);
```

EFT v8.0.6 and later

- Property **PGPTimePriorExpirationInDays** As Long

```
HRESULT PGPTimePriorExpirationInDays([out, retval] LONG*
pVal);
HRESULT PGPTimePriorExpirationInDays([in] LONG newVal);
```

EFT v8.0.6 and later

- Property **PGPKeys** As String

Retrieves a list of the Site's PGP Keys. Its value is a variant containing the array of [ICIPGPKeyInfo](#)

```
HRESULT PGPPKeys([out, retval] VARIANT* pVal);
```

Example (VBScript):

```
For Each key In oSite. PGPPKeys
  MsgBox key.Name & "key is available"
Next
```

EFT v8 and later

- Property [PGPLogFilePath](#) As String

Retrieves or specifies the log file path.

```
HRESULT PGPLogFilePath([out, retval] BSTR* pVal);
HRESULT PGPLogFilePath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "PGP log file path: " & oSite.PGPLogFilePath
```

Specifying:

```
oSite.PGPLogFilePath = "C:\PGP\logfile.txt"
```

EFT v6.3 and later

- Property [PGPLogLevel](#) As PGPLogLevel

Retrieves or specifies the OpenPGP logging level.

```
HRESULT PGPLogLevel([out, retval] PGPLogLevel* pVal);
HRESULT PGPLogLevel([in] PGPLogLevel newVal);
```

PGPLogLevel

- PGPLogLevelStandard = 0
- PGPLogLevelVerbose = 1

Example (VBScript):

Retrieving:


```
if PGPLogLevelStandard = oSite.PGPLogLevel
MsgBox "PGP log level is standard."
End If
```

Specifying:

```
oSite.PGPLogLevel = PGPLogLevelVerbose
```

EFT v6.3 and later

- Property [RadiusConnectionRetriesCount](#) As Long

Retrieves or specifies the number of connections retries allowed to connect to the RADIUS server.

```
HRESULT RadiusConnectionRetriesCount([out, retval] long*
pVal);
HRESULT RadiusConnectionRetriesCount([in] long newVal);
```

Example (VBScript):

```
Site.RadiusConnectionRetriesCount = 2
```

EFT v6.3 and later

- Property [RadiusNasIdentifier](#) As String

Retrieves or specifies the NAS identifier of the RADIUS server.

```
HRESULT RadiusNasIdentifier([out, retval] BSTR* pVal);
HRESULT RadiusNasIdentifier([in] BSTR newVal);
```

Example (VBScript):

```
MsgBox "NAS identifier: " & CStr(oSite.RadiusNasIdentifier)
```

EFT v6.3 and later

- Property [RadiusServerName](#) As String

Retrieves or specifies the name of the RADIUS server.

```
HRESULT RadiusServerName([out, retval] BSTR* pVal);
HRESULT RadiusServerName([in] BSTR newVal);
```

Example (VBScript):

```
MsgBox "Radius server name: " & CStr(oSite.  
RadiusServerName)
```

EFT v6.3 and later

- Property [RadiusServerPort](#) As Long

Retrieves or specifies the port number of the RADIUS server.

```
HRESULT RadiusServerPort([out, retval] long* pVal);  
HRESULT RadiusServerPort([in] long newVal);
```

Example (VBScript):

```
oSite. RadiusServerPort = 1645
```

EFT v6.3 and later

- Property [RadiusSharedSecret](#) As String

Retrieves or specifies the shared secret of the RADIUS server.

```
HRESULT RadiusSharedSecret([out, retval] BSTR* pVal);  
HRESULT RadiusSharedSecret([in] BSTR newVal);
```

Example (VBScript):

```
MsgBox "Radius shared secret: " & CStr(oSite.  
RadiusSharedSecret)
```

EFT v6.3 and later

- Property [RadiusTimeout](#) As Long

Retrieves or specifies the timeout for connection attempts to the RADIUS server in milliseconds.

```
HRESULT RadiusTimeout([out, retval] long* pVal);  
HRESULT RadiusTimeout([in] long newVal);
```

Example (VBScript):

```
oSite.RadiusTimeout = 10000
```

EFT v6.3 and later

- Property [RecalculateClientUsedSpace](#) (BSTR bstrUser)

Recalculates user disk quota for specified user.

Example (VBScript):

```
object[] users = site.GetUsers() as object[];
    foreach (string user in users)
    {
        site.RecalculateClientUsedSpace(user);
    }
```

EFT v7.4.13 and later

- Property [RedirectHTTPtoHTTPS](#) As Boolean

Retrieves or specifies whether to redirect all HTTP traffic to HTTPS. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT RedirectHTTPtoHTTPS([out, retval] VARIANT_BOOL*
pVal);
HRESULT RedirectHTTPtoHTTPS([in] VARIANT_BOOL newVal);
```

True = Enabled; False = Disabled

Example (VBScript):

Retrieving:

```
MsgBox "Redirecting is enabled: " & CStr
(oSite.RedirectHTTPtoHTTPS)
```

Specifying:

```
oSite.EnableAccountManagementOverHTTPS = True
```

EFT v6.3 and later

- Sub **RemapVirtualFolder**(bstrVFSFolder As String, bstrTarget As String)

Updates an existing virtual folder path to point to a physical folder.

```
HRESULT RemapVirtualFolder(
[in] BSTR bstrVFSFolder,
[in] BSTR bstrTarget);
```

Example (VBScript):

```
var bResult = oSite.RemapVirtualFolder( "/usr/name",
    \\file01\home\name")
var bResult = oSite.RemapVirtualFolder "/MyVirtualFolder",
    "c:\testfolder"
```

EFT v4.3.4 and later

- Sub **RemapVirtualFolders**(paths)

```
HRESULT RemapVirtualFolders([in] VARIANT paths);
```

EFT v7.4.13

- Function **RemoteEventRules**(type As [EventType](#)) As Object

```
HRESULT RemoteEventRules([in] EventType type, [out, retval]
    IDispatch** ppdispRules);
```

EFT v7.4.11 and later

- Sub **Remove**()

For an example using this method, refer to the RemoveSite script in [Script Examples](#).

Removes the Site. After removal, the Site object is no longer available.

```
HRESULT Remove();
```

Example (C#):

```
siteToRemove.Remove();
```

All versions

- Sub **RemoveAdvancedWorkflow**(lIndex As Long)

Given the Advanced Workflow zero-based index, removes the Workflow from this site

```
HRESULT RemoveAdvancedWorkflow([in] long lIndex);
    // zero-based index if the AW to remove
```

Example (VBScript):

```
`Remove workflow by name
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")
if idxAW < 0 Then
```

```
site.RemoveAdvancedWorkflow idxAW
End If
```

EFT v6.5 and later

- Sub **RemoveCommand**(bstrName As String)

Removes a command.

```
HRESULT RemoveCommand([in] BSTR bstrName);
```

All versions

- Sub **RemoveConnectionProfile**(lIndex As Long)

See also [AddConnectionProfile](#).

```
HRESULT RemoveConnectionProfile([in] LONG lIndex);
```

EFT v7.2 and later

- Sub **RemoveFolder**(bstrFolder As String)

Removes the specified folder.

```
HRESULT RemoveFolder([in] BSTR bstrFolder);
```

All versions

- Property [RemoveInactiveAccounts](#) As Boolean

For an example using this property, refer to the GetCertInfo, ConfigureSite, or ConfigureUser script in [Script Examples](#).

Retrieves or specifies whether to remove inactive accounts.

```
HRESULT RemoveInactiveAccounts([out, retval] VARIANT_BOOL*
pVal);
HRESULT RemoveInactiveAccounts([in] VARIANT_BOOL newVal);
```

See also [InactiveAccountsMonitoring](#).

Example (VBScript):

Retrieving:

```
if False = oSite.RemoveInactiveAccounts then
MsgBox "Disable"
```

```
else
MsgBox "Remove"
end if
```

Specifying:

```
oSite.RemoveInactiveAccounts = True
```

Example (C#):

```
selectedSite.RemoveInactiveAccounts = true;
```

EFT v6.3 and later

- Sub [RemoveIPAccessRule](#)(nRulePos As Long)

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Removes allowed or denied IP mask for the Site.

```
HRESULT RemoveIPAccessRule([in] long nRulePos);
```

Example (VBScript):

```
oSite.RemoveIPAccessRule(0)
```

All versions

- Sub **RemovePendingCertificate**(IID As Long)

Removes a pending certificate.

```
HRESULT RemovePendingCertificate([in] long IID);
```

All versions

- Sub **RemovePermission**(bstrFolder As String, bstrClient As String)

Given Folder and Client removes the permission.

```
HRESULT RemovePermission([in] BSTR bstrFolder,[in] BSTR
bstrClient);
```

All versions

- Sub [RemovePermissionGroup](#)(bstrName As String)

Removes a permission group.

```
HRESULT RemovePermissionGroup([in] BSTR bstrName);
```

All versions

- Sub **RemovePGPKey**(bstrKeyId As String)

Removes PGPKey from Site by key's ID.

```
HRESULT RemovePGPKey([in] BSTR bstrKeyId);
```

Example (VBScript):

```
oSite.RemovePGPKey "0x01234567"
```

See also [CreatePGPKey](#), [ExportPGPKey](#), and [ImportPGPKey](#).

EFT v8 and later

- Sub **RemoveSettingsLevel**(bstrName As String)

Removes a Settings Template.

```
HRESULT RemoveSettingsLevel([in] BSTR bstrName);
```

All versions

- Sub **RemoveSSHKey** (keyIndex As Long)

Removes SSH key from site by key's index in SSHKeys array.

```
HRESULT RemoveSSHKey([in] LONG keyIndex);
```

Example (VBScript):

```
oSite.RemoveSSHKey keyIndex
```

See also [CreateSSHKey](#) and [RenameSSHKey](#).

EFT v8

- Sub **RemoveTrustedCertificate**(IID As Long)

Removes a trusted certificate.

```
HRESULT RemoveTrustedCertificate([in] long IID);
```

All versions

- Sub **RemoveUser**(bstrLogin As String)

Removes a user.

```
HRESULT RemoveUser([in] BSTR bstrLogin);
```

All versions

- Sub [RemoveUserFromPermissionGroup](#)(bstrUser As String, bstrGroup As String)

Removes a user from a permission group.

```
HRESULT RemoveUserFromPermissionGroup([in] BSTR bstrUser,  
[in] BSTR bstrGroup);
```

All versions

- Sub **RemoveVirtualFolders**(paths)

```
HRESULT RemoveVirtualFolders([in] VARIANT paths);
```

EFT v7.4.13 and later

- Sub **RemoveWorkspace**(bstrGUID As String)

Remove a Workspaces folder

```
HRESULT RemoveWorkspace([in] BSTR bstrGUID);
```

EFT v7.1 and later

- Sub **RenameFolder**(bstrSrcFolder As String, bstrDstFolder As String)

Renames a folder.

```
HRESULT RenameFolder([in] BSTR bstrSrcFolder, [in] BSTR  
bstrDstFolder);
```

All versions

- Sub [RenamePermissionGroup](#)(bstrOldName As String, bstrNewName As String)

Renames a permission group.

```
HRESULT RenamePermissionGroup([in] BSTR bstrOldName, [in]
BSTR bstrNewName);
```

All versions

- Sub **RenameSettingsLevel**(bstrOldName As String, bstrNewName As String)

Renames a Settings Template.

```
HRESULT RenameSettingsLevel([in] BSTR bstrOldName, [in] BSTR
bstrNewName);
```

All versions

- Sub **RenameSSHKey**(keyIndex As Long, newName As String)

Renames the specified SSH Key by key's index in SSHKeys array.

```
HRESULT RenameSSHKey([in] LONG keyIndex, [in] BSTR newName);
```

Example (VBScript):

```
oSite.RenameSSHKey keyIndex, "new SSH key Name"
```

See also [CreateSSHKeyPair](#) and [RemoveSSHKey](#).

EFT v8 and later

- Sub **RenameUser**(bstrOldName As String, bstrNewName As String)

Renames a user.

```
HRESULT RenameUser([in] BSTR bstrOldName, [in] BSTR
bstrNewName);
```

All versions

- Property **RequestFilePortalPath** As string

Enable Workspaces users to request files and specify the path to the Reply portal.

```
HRESULT RequestFilePortalPath([out, retval] BSTR *pVal);
HRESULT RequestFilePortalPath([in] BSTR newVal);
```

Examples:

```
If oSite.EnableWorkspacesRequestFile
Then
Msgbox "Request File Portal URL is " &
oSite.RequestFilePortalPath
End if
```

```
oSite.EnableWorkspacesRequestFile = True
oSite.RequestFilePortalPath = "/requestPortal"
```

EFT v8.0

- Property [RequireStrongPasswords](#) As Boolean

Retrieves or specifies whether strong (complex) passwords are required.

```
HRESULT RequireStrongPasswords([out, retval] VARIANT_BOOL*
pVal);
HRESULT RequireStrongPasswords([in] VARIANT_BOOL newVal);
```

TRUE=Enforce strong passwords; FALSE=Do not enforce strong passwords

Example (VBScript):

Retrieving:

```
MsgBox "Strong passwords for site: " & CStr
(oSite.RequireStrongPasswords)
```

Specifying:

```
oSite.RequireStrongPasswords = False
```

EFT v6.3 and later

- Sub **ResetSubFolders**(bstrFolder As String)

Resets subfolders to default permissions.

```
HRESULT ResetSubFolders([in] BSTR bstrFolder);
```

Example (VBScript):

```
oSite.ResetSubFolder "/some_folder"
```

EFT v6.3 and later

- Property [RSAConfigurationPath](#) As String

Retrieves the path to the RSA configuration file.

```
HRESULT RSAConfigurationPath([out, retval] BSTR* pVal);
HRESULT RSAConfigurationPath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "RSA config path: " & oSite.RSAConfigurationPath
```

Specifying:

```
oSite.RSAConfigurationPath = "C:\RSAConfig\SDConf.rec"
```

EFT v6.3 and later

- Property **SecretsModuleBackupClientID** As String

```
HRESULT SecretsModuleBackupClientID([out, retval] BSTR* pVal);
HRESULT SecretsModuleBackupClientID([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleBackupClientSecret** As String

```
HRESULT SecretsModuleBackupClientSecret([out, retval] BSTR* pVal);
HRESULT SecretsModuleBackupClientSecret([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleBackupEnable** As Boolean

```
HRESULT SecretsModuleBackupEnable([out, retval] VARIANT_BOOL* pVal);
HRESULT SecretsModuleBackupEnable([in] VARIANT_BOOL newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleBackupType** As [SecretsModuleType](#)

```
HRESULT SecretsModuleBackupType([in] SecretsModuleType newVal);
```

```
HRESULT SecretsModuleBackupType([out, retval]
SecretsModuleType* pVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleBackupURL** As String

```
HRESULT SecretsModuleBackupURL([in] BSTR newVal);
HRESULT SecretsModuleBackupURL([out, retval] BSTR* pVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleClientID** As String

```
HRESULT SecretsModuleClientID([out, retval] BSTR *pVal);
HRESULT SecretsModuleClientID([in] BSTR newVal);
```

EFT v8.0.1 and later

- Property **SecretsModuleClientSecret** As String

```
HRESULT SecretsModuleClientSecret([out, retval] BSTR *pVal);
HRESULT SecretsModuleClientSecret([in] BSTR newVal);
```

EFT v8.0.1 and later

- Property **SecretsModuleProxyHostname** As String

```
HRESULT SecretsModuleProxyHostname([out, retval] BSTR*
pVal);
HRESULT SecretsModuleProxyHostname([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleProxyPassword** As String

```
HRESULT SecretsModuleProxyPassword([out, retval] BSTR*
pVal);
HRESULT SecretsModuleProxyPassword([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleProxyPort** As Long

```
HRESULT SecretsModuleProxyPort([in] long newVal);
HRESULT SecretsModuleProxyPort([out, retval] long* pVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleProxyUsername** As String

```
HRESULT SecretsModuleProxyUsername([out, retval] BSTR*
pVal);
HRESULT SecretsModuleProxyUsername([in] BSTR newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleUseProxy** As Boolean

```
HRESULT SecretsModuleUseProxy([in] VARIANT_BOOL newVal);
HRESULT SecretsModuleUseProxy([out, retval] VARIANT_BOOL*
pVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleType** As [SecretsModuleType](#)

```
HRESULT SecretsModuleType([out, retval] SecretsModuleType*
pVal);
HRESULT SecretsModuleType([in] SecretsModuleType newVal);
```

EFT v8.0.1 and later

- Property **SecretsModuleWriteToBackup** As Boolean

```
HRESULT SecretsModuleWriteToBackup([out, retval] VARIANT_
BOOL* pVal);
HRESULT SecretsModuleWriteToBackup([in] VARIANT_BOOL
newVal);
```

EFT v8.0.5 and later

- Property **SecretsModuleURL** As String

```
HRESULT SecretsModuleURL([out, retval] BSTR *pVal);
HRESULT SecretsModuleURL([in] BSTR newVal);
```

EFT vv8.0.1 and later

- Property **SendPortalPath** As String

```
HRESULT SendPortalPath([out, retval] BSTR *pVal);
HRESULT SendPortalPath([in] BSTR *pVal);
```

EFT v7.3.6 and later

- Sub **SetACLForAdvancedWorkflowsRoot**(pVal As Object)

See also [GetACLForAdvancedWorkflowsRoot](#).

```
HRESULT SetACLForAdvancedWorkflowsRoot([in] IDispatch*
pVal);
```

EFT v7.2 and later

- Sub **SetACLForConnectionProfilesRoot**(pVal As Object)

See also [GetACLForConnectProfilesRoot](#).

```
HRESULT SetACLForConnectionProfilesRoot([in] IDispatch*
pVal);
```

EFT v7.2 and later

- Sub **SetACLForCustomCommandsRoot**(pVal As Object)

See also [GetACLForCustomCommandsRoot](#).

```
HRESULT SetACLForCustomCommandsRoot([in] IDispatch* pVal);
```

EFT v7.2 and later

- Function **SetACLForEventRulesRoot**(pVal As Object)

See also [GetACLForEventRulesRoot](#).

```
HRESULT SetACLForEventRulesRoot([in] IDispatch* pVal);
```

EFT v7.2 and later

- Sub **SetAdvancedWorkflowParams**(lIndex As Long, pdisplParams As Object)

(See also [GetAdvancedWorkflowParams](#))

Given Advanced Workflow zero-based index, changes the Workflow's parameters

```
HRESULT SetAdvancedWorkflowParams(
[in] long lIndex, // zero-based index of the AW
[in] IDispatch* pdisplParams); // AW parameters // as
ICIAdvancedWorkflowParams*
```

Example (VBScript):

```
`Rename workflow
idxAW = site.GetAdvancedWorkflowIndex("My Workflow")
if idxAW < 1 Then
```

```
Set AWparams = site.GetAdvancedWorkflowParams(idxAW)
AWparams.Name = "My Workflow - new name"
site.SetAdvancedWorkflowParams idxAW, AWparams
End If
```

EFT v6.5 and later

- Sub [SetAllowCOMB](#)(newVal As Boolean)

Specifies whether users on the Site are allowed to use multi-part transfers (the COMB command) for FTP transfers.

```
HRESULT SetAllowCOMB([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
oSite.SetAllowCOMB True
```

See also: [GetAllowComb](#)

EFT v6.3 and later

- Sub [SetAllowFXP](#)(newVal As Boolean)

Specifies whether users on the Site are allowed to use site-to-site transfers (FXP) for FTP transfers.

```
HRESULT SetAllowFXP([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
oSite.SetAllowFXP True
```

See also: [GetAllowFXP](#)

EFT v6.3 and later

- Sub [SetAllowNoop](#)(newVal As Boolean)

Specifies whether users on the Site are allowed to use client anti-timeout measures (the NOOP command) for FTP transfers.

```
HRESULT SetAllowNoop([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
oSite.SetAllowNoop True
```

See also: [GetAllowNoop](#)

EFT v6.3 and later

- Sub [SetAllowXCRC](#)(newVal As Boolean)

Specifies whether users on the Site are allowed to use integrity checking (the XCRC command) for FTP transfers.

```
HRESULT SetAllowXCRC([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
oSite.SetAllowXCRC True
```

See also: [GetAllowXCRC](#)

EFT v6.3 and later

- Sub **SetAMParams**(newVal As Object)

For an example using this method, refer to the Windows Authentication script in [Script Examples](#).

Specifies authentication manager (AM) parameters.

```
HRESULT SetAMParams([in] IDispatch* newVal);
```

Example (VBScript)

```
Set oAMParams = oSite.GetAMParams()  
oAMParams.RefreshIntervalMinutes = 10  
oSite.SetAMParams(oAMParams)
```

See also: [GetAMParams](#)

EFT v6.3 and later

- Function [SetCertFilePath](#)(newVal As String) As Boolean

Changes the certificate file path and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetCertFilePath(  
[in] BSTR newVal,  
[out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetCertFilePath](#)

All versions

- Function [SetCheckClientCert](#)(newVal As Boolean) As Boolean

Enables or disables the requirement of certificates on an implicit SSL Site. If you must restart the Server for the change to take effect the SetCheckClientCert method returns a value of TRUE. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetCheckClientCert(
    [in] VARIANT_BOOL newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetCheckClientCert](#)

All versions

- Sub **SetClientDataField**(pdispField As Object)

Modify client data field object ([ICIClientDataField](#))

```
HRESULT SetClientDataField([in] IDispatch* pdispField);
```

Example (VBScript):

```
set email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsEnabled = Not email.IsEnabled
oSite.SetClientDataField(emailDataSettings)
```

See also: [GetClientDataField](#)

EFT 7.5 and later

- Sub [SetComplexPasswordSettings](#)(pIComplexPasswordSettings As Object)

Specifies complex password settings.

```
HRESULT SetComplexPasswordSettings(
    (In v6.3 - 6.3.3) [in] IDispatch*
    pIComplexPasswordSettings);
(In v6.3.8 and later) [in] ICIResetPasswordSettings*
    pIResetPasswordSettings);
```

Example (VBScript):

```
Set oPwdSettings = oSite.GetComplexPasswordSettings()
oPwdSettings.MinPasswordLength = 8
```

```
oSite.SetComplexPasswordSettings(oPwdSettings)
```

See also: [GetComplexPasswordSettings](#)

EFT v6.3 and later

- Sub **SetConnectionProfileParams**(lIndex As Long, pdispParams As Object)

See also [GetConnectionProfileParams](#).

```
HRESULT SetConnectionProfileParams([in] LONG lIndex, [in]
IDispatch* pdispParams);
```

See also: [GetConnectionProfileParams](#)

EFT v7.2 and later

- Sub **SetDefaultPGPKey**(bzKeyID As String, bzPassPhrase As String)

See also [DefaultPGPKeyID](#) and [DefaultPGPKeyPassphrase](#).

Specifies the default PGP Key and passphrase. Specify an empty string for the key to assign "no key."

```
HRESULT SetDefaultPGPKey(
[in] BSTR bzKeyID,
[in] BSTR bzPassphrase);
```

EFT v6.4 and later

- Function [SetFTPAccess](#)(newVal As Boolean) As Boolean

Enables or disables FTP access and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetFTPAccess(
[in] VARIANT_BOOL newVal,
[out, retval] VARIANT_BOOL *pNeedRestart);
```

True = Access enabled; False = Access disabled

See also: [GetFTPAccess](#)

All versions

- Function **SetGDPRSettings**() As Object

Modify GDPR settings ([ICIGDPRSettings](#)).

```
HRESULT SetGDPRSettings([in] IDispatch *pdispParams);
```

Example:

```
set gdpr = oSite.GetGDPRSettings
gdpr.MaterialScope = MaterialScope_InScope
oSite.SetGDPRSettings(gdpr)
```

See also [GetGDPRSettings](#)

EFT v8

- Function [SetHTTPAccess](#)(newVal As Boolean) As Boolean

Enables or disables HTTP access and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetHTTPAccess(
[in] VARIANT_BOOL newVal,
[out, retval] VARIANT_BOOL *pNeedRestart);
```

True = Access enabled; False = Access disabled

See also: [GetHTTPAccess](#)

EFT v4.3.4 and later

- Function [SetHTTPPort](#)(newVal As Long) As Boolean

Changes the HTTP port number of the Site and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetHTTPPort(
[in] long newVal,
[out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetHTTPPort](#)

EFT v4.3.4 and later

- Function [SetHTTPSAccess](#)(newVal As Boolean) As Boolean

Enables or disables HTTPS access and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetHTTPSAccess(
    [in] VARIANT_BOOL newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetHTTPSAccess](#)

EFT v4.3.4 and later

- Function [SetHTTPSPort](#)(newVal As Long) As Boolean

Changes the HTTPS port number of the Site and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetHTTPSPort(
    [in] long newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetHTTPSPort](#)

EFT v4.3.4 and later

- Function [SetIP](#)(newVal As Long) As Boolean

Changes the index of Listen IP and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetIP(
    [in] long newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetIP](#)

All versions

- Function [SetKeyFilePath](#)(newVal As String) As Boolean

Changes the Site's SSL private key file path and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetKeyFilePath(
    [in] BSTR newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetKeyFilePath](#)

All versions

- Function [SetPassPhrase](#)(newVal As String) As Boolean

Changes the Site's SSL private key passphrase and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetPassPhrase(
    [in] BSTR newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetPassPhrase](#)

All versions

- Sub **SetPermission**(pPermission As [Permission](#), [bRemoveOtherPermissions As Boolean])

Refer to [ICIPermission - VFS Permission Interface](#) for the available permission properties.

Adds a new permission or changes an existing one.

```
HRESULT SetPermission(
    [in] IPermission *pPermission,
    [in, optional] VARIANT_BOOL bRemoveOtherPermissions);
```

Example (VBScript):

If you want to set permissions on a folder for a user and remove all other permissions, pass TRUE to the second parameter of setPermission(). This will REMOVE all other permissions from a folder (that is, break the inheritance).

For example,

```
set perm = Site.GetBlankPermission( "/usr/foo", "foo" )
perm.FileDelete = True
perm.FileUpload = True
perm.FileDownload = False
Site.SetPermission( perm, True )
```

See also: [GetBlankPermission](#), [RemovePermission](#), [RemovePermissionGroup](#), [RenamePermissionGroup](#)

All versions

- Function [SetPort](#)(newVal As Long) As Boolean

Changes the port number and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetPort([in] long newVal,[out, retval] VARIANT_BOOL
*pNeedRestart);
```

See also: [GetPort](#)

All versions

- Sub **SetPrivacyPolicy**(pdispParams As Object)

Modifies the Privacy Policy settings ([ICIPrivacyPolicy](#))

```
HRESULT SetPrivacyPolicy([in] IDispatch *pdispParams);
```

Example (VBScript):

```
set tos = oSite.GetPrivacyPolicy
tos.Mode = GDPRPrivacyPolicyMode_Disabled
oSite.SetPrivacyPolicy(tos)
```

See also: [GetPrivacyPolicy](#)

EFT v8

- Sub [SetResetPasswordSettings](#)(pIResetPasswordSettings As Object)

Use [ICISite::GetResetPasswordSettings](#) and [ICISite::SetResetPasswordSettings](#) to retrieve and set the password settings for a Site.

Specifies password reset settings.

```
HRESULT SetResetPasswordSettings (
(In v6.3 - 6.3.3)          [in] ICIResetPasswordSettings*
pIResetPasswordSettings);
(In v6.3.8 and later)    [in] IDispatch*
pIResetPasswordSettings);
```

Example (VBScript):

```
Set oPwdSettings = oSite.GetResetPasswordSettings()
PwdSettings.MaxPasswordAgeDays = 8
oSite.SetResetPasswordSettings(oPwdSettings)
```

See also: [GetResetPasswordSettings](#)

EFT v6.3 and later

- Function [SetRootFolder](#)(newVal As String) As Boolean

Changes the root folder of the site and returns TRUE if the site has to be restarted.
(Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetRootFolder(
    [in] BSTR newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetRootFolder](#)

All versions

- Property **SSHKeys** As Variant

Retrieves SSH keys on a Site.

```
HRESULT SSHKeys([out, retval] VARIANT* pVal);
```

See also: [GetSSHKeyForSFTP](#), [SetSSHKeyForSFTP](#), [Site_SFTP_Settings](#)

EFT v8

- Property **SSLCipherList** As String

Used to provide list of ciphers.

```
HRESULT SSLCipherList([out, retval] BSTR *pVal);
HRESULT SSLCipherList([in] BSTR newVal);
```

EFT v8.0.5

- Sub **SetSSHKeyForSFTP**([in] Long keyIndex)

Specifies Site's SSH key by key's index in SSHKeys array.

```
HRESULT SetSSHKeyForSFTP([in] LONG keyIndex);
```

Example (VBScript):

```
oSite.SetSSHKeyForSFTP
    keyIndex, "password"
```

See also: [GetSSHKeyForSFTP](#), [Site_SFTP_Settings](#)

EFT v8

- Property **SSLVersionMask** As Int

Specify the SSL versions allowed on the Site.

```
HRESULT SSLVersionMask([out, retval] int *pVal);
HRESULT SSLVersionMask([in] int newVal);
```

EFT v8.0.5

- Function [SetSSHKeyFilePath](#)(newVal As String) As Boolean

Changes the SSH Certificate file path and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetSSHKeyFilePath(
[in] BSTR newVal,
[out, retval] VARIANT_BOOL *pNeedRestart);
```

Example (VBScript):

```
Dim bNeedRestart: bNeedRestart = oSite.SetSSHKeyFilePath
("C:\Keys\Key.pvk")
If bNeedRestart Then
Call oSite.Stop()
Call oSite.Start()
End If
```

See also: [Site_SFTP_Settings](#)

EFT 4.3.4 - 7.4.x (deprecated in EFT v8)

- Function [SetSSLAuth](#)(newVal As Boolean) As Boolean

Enables or disables SSL authentication and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetSSLAuth(
[in] VARIANT_BOOL newVal,
[out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetSSLAuth](#)

All versions

- Sub **SetSSLCertificate**(bzCertFilePath As String, bzKeyFilePath As String, bzPassPhrase As String)

Specifies the Site's SSL certificate file path, key file path, and passphrase.


```
HRESULT SetSSLCertificate(
    [in] BSTR bzCertFilePath,
    [in] BSTR bzKeyFilePath,
    [in] BSTR bzPassPhrase);
```

Example (VBScript):

```
Site.SetSSLCertificate "C:\MySite.crt", "C:\ MySite.key",
    "My-Site_passwd"
```

See also: [GetSSLCertificate](#)

EFT v6.1 and later

- Function [SetSSLImp](#)(newVal As Boolean) As Boolean

Enables or disables SSLImp and returns TRUE if the Site has to be restarted. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SetSSLImp(
    [in] VARIANT_BOOL newVal,
    [out, retval] VARIANT_BOOL *pNeedRestart);
```

See also: [GetSSLImp](#)

All versions

- Sub [SetTermsOfService](#)(pdispParams As Object)

Modifies the Terms of Service settings ([ICITermsOfService](#)).

```
HRESULT SetTermsOfService([in] IDispatch *pdispParams);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.Mode = GDPRTermsOfServiceMode_Disabled
oSite.SetTermsOfService(tos)
```

See also: [GetTermsOfService](#)

EFT v8

- Sub [SetWorkspace](#)(bstrGUID As String, pdispWorkspace As Object)

Sets the Workspace with the given GUID

```
HRESULT SetWorkspace([in] BSTR bstrGUID, IDispatch*
pdisplWorkspace);
```

See also: [GetWorkspace](#)

EFT v7.1 and later

(Refer to [ICIWorkspace](#) for examples.)

- Sub SetWorkspaceMaxDuration(TimeUnits As TimeUnits, val As DWord)

```
HRESULT SetWorkspaceMaxDuration(
[in] TimeUnits timeUnits,
[in] ULONG val);
```

See also [TimeUnits](#) enum [GetWorkspaceMaxDuration](#).

EFT v7.3.3 and later

- Sub SetWorkspacesOAIMaxDuration(TimeUnits As TimeUnits, val As DWord)

```
HRESULT SetWorkspacesOAIMaxDuration(
[in] TimeUnits timeUnits,
[in] ULONG val);
```

See also [TimeUnits](#) enum and [GetWorkspacesOAIMaxDuration](#).

EFT v7.3.3 and later

- Sub SetWorkspacesOAIRetainFilesFor

```
SetWorkspacesOAIRetainFilesFor(
[in] TimeUnits timeUnits,
[in] ULONG val);
```

See also [TimeUnits](#) enum and [GetWorkspacesOAIRetainFilesFor](#).

EFT v8

- Property **SFTPEncoding** as string

Refer to [TransferEncodingType](#) in the Enum Reference.

Retrieves or specifies the transfer encoding mode for SFTP. TransferEncodingUTF8 and TransferEncodingASCII are the only valid values for this property.

```
HRESULT SFTPEncoding([out, retval] TransferEncodingType*
pVal);
HRESULT SFTPEncoding([in] TransferEncodingType newVal);
```

Example (VBScript):

Retrieving:

```
If Site.SFTPEncoding = 1 Then
  'UTF-8 mode
End If
```

Specifying:

```
'set UTF-8 mode for SFTP
Site.SFTPEncoding = 1f
```

EFT v6.5 and later

- Property [SFTPEncodingAlgorithms](#) As String

Retrieves the allowed ciphers. You must restart the Site for changes to take effect.

```
HRESULT SFTPEncodingAlgorithms([out, retval] BSTR* pVal);
HRESULT SFTPEncodingAlgorithms([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP ciphers: " & oSite.SFTPEncodingAlgorithms
```

Specifying:

```
oSite.SFTPEncodingAlgorithms = "twofish256-cbc,twofish-
cbc,twofish128-cbc,blowfish-cbc"
```

EFT v6.3 and later

- Property [SFTPIdentificationComment](#) As String

Retrieves the SFTP comments. You must restart the Site for changes to take effect.
(Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SFTPIdentificationComment([out, retval] BSTR* pVal);
HRESULT SFTPIdentificationComment([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP comments: " & oSite.SFTPIdentificationComment
```

Specifying:

```
oSite.SFTPIdentificationComment = "GlobalSCAPE"
```

EFT v6.3 and later

- Property **SFTPKexAlgorithms** As String

```
HRESULT SFTPKexAlgorithms([out, retval] BSTR* pVal);
HRESULT SFTPKexAlgorithms([in] BSTR newVal);
```

EFT v7.4 and later

- Property [SFTPKeyFile](#) As String

Retrieves the path to the SFTP key. You must restart the Site for changes to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SFTPKeyFile([out, retval] BSTR* pVal);
HRESULT SFTPKeyFile([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP private key file is: " &
oSite.SFTPKeyFile
```

Specifying:

```
oSite.SFTPKeyFile = "C:\SFTPPriveteKeys\key1"
```

EFT v6.3 - 7.4.x (deprecated in EFT v8)

- Property [SFTPKeyPassphrase](#) As String

Retrieves the SFTP key passphrase. You must restart the Site for changes to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SFTPKeyPassphrase([out, retval] BSTR* pVal);
HRESULT SFTPKeyPassphrase([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP private key passphrase is: "
      & oSite.SFTPKeyPassphrase
```

Specifying:

```
oSite.SFTPKeyPassphrase = "new_passphrase"
```

EFT v6.3 and later

- Property [SFTPMacAlgorithms](#) As String

Retrieves the list of allowed MAC algorithms. You must restart the Site for changes to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SFTPMacAlgorithms([out, retval] BSTR* pVal);
HRESULT SFTPMacAlgorithms([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP MAC algorithphms: " & oSite.SFTPMacAlgorithms
```

Specifying:

```
oSite.SFTPMacAlgorithms = "hmac-sha1,hmac-md5,hmac-sha1-96,hmac-md5-96"
```

EFT v6.3 and later

- Property [SFTPPort](#) As Long

Retrieves or specifies the SFTP port number. You must restart the Site for the change to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SFTPPort([out, retval] LONG* pVal);
HRESULT SFTPPort([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP port: " & CStr(oSite.SFTPPort)
```

Specifying:

```
oSite.SFTPPort = 40
```

EFT v6.3 and later

- Property [SFTPPublicKeyBlob](#) As String (read only)

Retrieves the SFTP public key blob.

```
HRESULT SFTPPublicKeyBlob([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "SFTP public key blob: " & oSite.SFTPPublicKeyBlob
```

EFT v6.3 and later

- Property [SFTPSoftwareVersion](#) As String

Retrieves the SFTP software version. You must restart the Site for changes to take effect. (Refer to [Stop](#) and [Start](#) below.)

```
HRESULT SFTPSoftwareVersion([out, retval] BSTR* pVal);
HRESULT SFTPSoftwareVersion([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "SFTP software: " & oSite.SFTPSoftwareVersion
```

Specifying:

```
oSite.SFTPSoftwareVersion = "1.82_sshlib"
```

EFT v6.3 and later

- Property **SMSPProviderProfile** As Object

```
HRESULT SmsProviderProfile(
    [out, retval] IDispatch** ppProfile);
HRESULT SmsProviderProfile(
    [in] IDispatch* pProfile);
```

EFT v8.0.4 and later

- Property **SSHKeys** As Variant (read-only)

```
HRESULT SSHKeys([out, retval] VARIANT* pVal);
```

Retrieve a list of site's SSH Keys. Its value is a variant containing the array of [ICISSHKeyInfo](#).

Example (VBScript):

```
For Each key In oSite.SSHKeys
  MsgBox key.Name " key "
Next
```

EFT v8 and later

- Sub [Start\(\)](#)

Starts the Site.

```
HRESULT Start();
```

All versions

- Sub [Stop\(\)](#)

Stops the Site.

```
HRESULT Stop();
```

All versions

- Property **SuppressForgotPasswordMode** As [SuppressForgotPasswordMode](#)

```
HRESULT SuppressForgotPasswordMode(
  [out, retval] SuppressForgotPasswordMode *pVal);
HRESULT SuppressForgotPasswordMode(
  [in] SuppressForgotPasswordMode newVal);
```

EFT v7.4.11 and later

- Method **TestContentIntegrityControlProfile**

Test specified CIC profile

```
HRESULT TestContentIntegrityControlProfile(
  [in]
      IDispatch* pProfile, [out, retval] VARIANT_BOOL* pVal);
```

EFT v8 and later

- Property [TwoFactorAuthenticationTwoFactorAuthenticationMode](#)

Retrieves or specifies the two-factor authentication mode.

```
HRESULT TwoFactorAuthentication(
    [out, retval] TwoFactorAuthenticationMode* pVal);
HRESULT TwoFactorAuthentication(
    [in] TwoFactorAuthenticationMode newVal);
```

EFT v6.3 and later

- Sub **UpdateUploadForm**([in] IDispatch* pUploadForm, [in] long formIndex)

Used to upload an Upload Form.

```
HRESULT UpdateUploadForm(
    [in] IDispatch* pUploadForm,
    [in] long formIndex);
```

EFT v8.0.4 and later

- Property **UploadFormsList** As Variant (read only)

Provides a list of upload forms defined on the Site.

```
HRESULT UploadFormsList([out, retval] VARIANT *pVal);
```

EFT v8.0.4 and later

- Sub **UnbanIP**(bstrMask As String) (read only)

Removes denied IP mask from the Autoban list for the Site.

(Does not remove manually banned IPs. Use [RemoveIPAccessRule](#) instead.)

```
HRESULT UnbanIP([in] BSTR bstrMask //mask to remove);
```

Example (VBScript):

```
oSite.UnbanIP("1.2.3.4")
```

- Property **UnderPCIDSS** As Boolean (read only)

Returns VARIANT_TRUE if the Site is under PCI DSS policy. Used to query the Site to determine if it is a strict security/PCI DSS-enabled Site.

```
HRESULT UnderPCIDSS([out, retval] VARIANT_BOOL * pVal);
```


Example (VBScript):

```
MsgBox "Under PCIDSS: " & CStr(oSite.UnderPCIDSS)
```

EFT v6.3.1 and later

- Function **UnlockUser**(bstrUserName As String) As Boolean

Retrieves whether a user account is locked and unlock the account.

```
HRESULT UnlockUser(
    [in] BSTR bstrUserName,
    [out, retval] VARIANT_BOOL *pSuccess);
```

TRUE=Unlocked; FALSE=Locked

Example (VBScript):

```
if False = oSite.UnlockUser("User1") then
    MsgBox "Cannot unlock user."
End if
```

EFT v6.3 and later

- Method **UpdateContentIntegrityControlProfile**

```
HRESULT UpdateContentIntegrityControlProfile(
    [in] IDispatch* pProfile, [in] long profileIndex);
```

EFT v8

- Property [UseDynamicPGPLogFile](#) As Boolean

Retrieves or specifies whether to use dynamic log file.

```
HRESULT UseDynamicPGPLogFile(
    [out, retval] VARIANT_BOOL* pVal);
HRESULT UseDynamicPGPLogFile(
    [in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "PGP uses dynamic log file:" & CStr
(oSite.UseDynamicPGPLogFile)
```

Specifying:

```
oSite.UseDynamicPGPLogFile = true
```

EFT v6.3 and later

- Property [UserLimitMessage](#) As String

Retrieves or specifies user limit message.

```
HRESULT UserLimitMessage(
    [in] BSTR newVal);
[out, retval] BSTR *pVal);
```

All versions

- Property [UsersDefined](#) As Long (read only)

Retrieves the number of user accounts defined on the Site.

```
HRESULT UsersDefined(
    [out, retval] LONG* pVal)
```

Example (VBScript):

```
MsgBox "Users count: " & CStr(oSite.UsersDefined)
```

EFT v6.3 and later

- Function **ValidatePassword**(bstrUser As String, bstrPwd As String) As Boolean

Validates a user's password.

```
HRESULT ValidatePassword(
    [in] BSTR bstrUser,
    [in] BSTR bstrPwd,
    [out, retval] VARIANT_BOOL *bValid);
```

Example (VBScript):

```
If oSite.ValidatePassword( "test", "pass")
    Then
    WScript.Echo "Valid password"
End If
```

EFT v3.5.1 and later

- Property [VFSFilter](#) As String

Retrieves or specifies list of banned file types.

```
HRESULT VFSFilter([out, retval] BSTR *pVal);[in] BSTR
newVal);
```

All versions

- Property **WebSSOAttributeName** As String OAttributeName

```
HRESULT WebSSOAttributeName([in] BSTR newVal);
HRESULT WebSSOAttributeName([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WebSSOIdpEntityId** As String

```
HRESULT WebSSOIdpEntityId([in] BSTR newVal);
HRESULT WebSSOIdpEntityId([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WebSSOExtendUsernameLookupToAuthProvider** As Boolean

```
HRESULT WebSSOExtendUsernameLookupToAuthProvider(
[out, retval] VARIANT_BOOL* pVal);
HRESULT WebSSOExtendUsernameLookupToAuthProvider(
[in] VARIANT_BOOL newVal);
```

EFT v7.4.13 and later

- Property **WebSSOIdpPostUrl** As String

```
HRESULT WebSSOIdpPostUrl([in] BSTR newVal);
HRESULT WebSSOIdpPostUrl([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WebSSOIdpPublicKeyPath** As String

```
HRESULT WebSSOIdpPublicKeyPath([in] BSTR newVal);
HRESULT WebSSOIdpPublicKeyPath([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WebSSOJitUserProvisioning** As Boolean

```
HRESULT WebSSOJitUserProvisioning(
[out, retval] VARIANT_BOOL* pVal);
HRESULT WebSSOJitUserProvisioning(
[in] VARIANT_BOOL newVal);
```

EFT v7.4.13 and later

- Property **WebSSOJitUserProvisioningEmailLocation** As Boolean

```
HRESULT WebSSOJitUserProvisioningEmailLocation(
    [out, retval] long* pVal);
HRESULT WebSSOJitUserProvisioningEmailLocation(
    [in] long newVal);
```

EFT v8 and later

- Property **WebSSOJitUserProvisioningEmailAttribute** As Boolean

```
HRESULT WebSSOJitUserProvisioningEmailAttribute([out,
    retval] BSTR* pVal);
HRESULT WebSSOJitUserProvisioningEmailAttribute([in] BSTR
    newVal);
```

EFT v8 and later

- Property **WebSSOJitUserProvisioningSettingsTemplate** As String

```
HRESULT WebSSOJitUserProvisioningSettingsTemplate([out,
    retval] BSTR* pVal);
HRESULT WebSSOJitUserProvisioningSettingsTemplate([in] BSTR
    newVal);
```

EFT v7.4.13 and later

- Property **WebSSONameOrAttribute** As [WebSSONameOrAttribute](#)

```
HRESULT WebSSONameOrAttribute([in] WebSSONameOrAttribute
    newVal);
HRESULT WebSSONameOrAttribute([out, retval]
    WebSSONameOrAttribute *pVal);
```

EFT v7.3.3 and later

- Property **WebSSONameOrAttributeIdentifierFormat** As [WebSSONameOrAttributeIdentifierFormat](#)

```
HRESULT WebSSONameOrAttributeIdentifierFormat(
    [out, retval] WebSSONameOrAttributeIdentifierFormat *pVal);
HRESULT WebSSONameOrAttributeIdentifierFormat(
    [in] WebSSONameOrAttributeIdentifierFormat newVal);
```

EFT v7.3.3 and later

- Property **WebSSORedirectToSsoServiceBypassingLoginPage** As Boolean

```
HRESULT WebSSORedirectToSsoServiceBypassingLoginPage(
    [out, retval] VARIANT_BOOL* pVal);
HRESULT WebSSORedirectToSsoServiceBypassingLoginPage(
    [in] VARIANT_BOOL pVal);
```

EFT v8.0.4 and later

- Property **WebSSOSpEntityId** As String

```
HRESULT WebSSOSpEntityId([in] BSTR newVal);
HRESULT WebSSOSpEntityId([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WebSSOSpReservedPath** As String

```
HRESULT WebSSOSpReservedPath([in] BSTR newVal);
HRESULT WebSSOSpReservedPath([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WebSSOUserNameRegex** As String

```
HRESULT WebSSOUserNameRegex([in] BSTR newVal);
HRESULT WebSSOUserNameRegex([out, retval] BSTR* pVal);
```

EFT v7.3.3 and later

- Property **WorkspacesDomainsToForbidInviteeAddresses** As String

```
HRESULT WorkspacesDomainsToForbidInviteeAddresses(
    [out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesDomainsToForbidInviteeAddresses(
    [in] VARIANT_BOOL newVal);
```

EFT v8 and later

- Property **WorkspacesDomainsToLimitInviteeAddresses** As String

```
HRESULT WorkspacesDomainsToLimitInviteeAddresses(
    [out, retval] BSTR *pVal);
HRESULT WorkspacesDomainsToLimitInviteeAddresses(
    [in] BSTR newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesDropOffAllowRecipientAddresses** As Boolean

```
HRESULT WorkspacesDropOffAllowRecipientAddresses (
    [out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesDropOffAllowRecipientAddresses (
    [in] VARIANT_BOOL pVal);
```

EFT v7.4.2 and later

- Property **WorkspacesDropOffCaptchaType** As [CaptchaType](#)

```
HRESULT WorkspacesDropOffCaptchaType (
    [out, retval] CaptchaType *pVal);
HRESULT WorkspacesDropOffCaptchaType (
    [in] CaptchaType newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesDropOffDomainsToLimitRecipientAddresses** As String

```
HRESULT WorkspacesDropOffDomainsToLimitRecipientAddresses
    ([out, retval] BSTR *pVal);
HRESULT WorkspacesDropOffDomainsToLimitRecipientAddresses
    ([in] BSTR newVal);
```

EFT v7.4.2 and later

- Property **WorkspacesDropOffEnableDomainsToLimitRecipientAddresses** As String

```
HRESULT
WorkspacesDropOffEnableDomainsToLimitRecipientAddresses
    ([out, retval] BSTR *pVal);
HRESULT
WorkspacesDropOffEnableDomainsToLimitRecipientAddresses ([in]
    BSTR newVal);
```

EFT v7.4.9 and later

- Property **WorkspacesDropOffEnableRecaptcha** As Boolean

```
HRESULT WorkspacesDropOffEnableRecaptcha ([out, retval]
    VARIANT_BOOL* pVal);
HRESULT WorkspacesDropOffEnableRecaptcha ([in] VARIANT_BOOL
    pVal);
```

EFT v7.4.2 and later

- Property **WorkspacesDropOffLimitMessageSize** As Boolean

```
HRESULT WorkspacesDropOffLimitMessageSize([out, retval]
VARIANT_BOOL *pVal);
HRESULT WorkspacesDropOffLimitMessageSize([in] VARIANT_BOOL
pVal);
```

EFT v7.4.2 and later

- Property **WorkspacesDropOffRecaptchaSecretKey** As String

```
HRESULT WorkspacesDropOffRecaptchaSecretKey([out, retval]
BSTR *pVal);
HRESULT WorkspacesDropOffRecaptchaSecretKey([in] BSTR
newVal);
```

EFT v7.4.2 and later

- Property **WorkspacesDropOffRecaptchaSiteKey** As String

```
HRESULT WorkspacesDropOffRecaptchaSiteKey(
[out, retval] BSTR *pVal);
HRESULT WorkspacesDropOffRecaptchaSiteKey(
[in] BSTR newVal);
```

EFT v7.4.2 and later

- Property **WorkspacesDropOffSecureMessageBodyPolicy** As [WorkspacesSecureMessageBodyPolicy](#)

```
HRESULT WorkspacesDropOffSecureMessageBodyPolicy(
[out, retval] WorkspacesSecureMessageBodyPolicy *pVal);
HRESULT WorkspacesDropOffSecureMessageBodyPolicy(
[in] WorkspacesSecureMessageBodyPolicy pVal);
```

EFT v7.4.6 and later

- Property **WorkspacesEnableDomainsToForbidInviteeAddresses** As String

```
HRESULT WorkspacesEnableDomainsToForbidInviteeAddresses
([out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesEnableDomainsToForbidInviteeAddresses([in]
VARIANT_BOOL newVal);
```

EFT v8 and later

- Property **WorkspacesEnableDomainsToLimitInviteeAddresses** As String

```
HRESULT WorkspacesEnableDomainsToLimitInviteeAddresses(
    [out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesEnableDomainsToLimitInviteeAddresses(
    [in] VARIANT_BOOL newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesGAEnableParticipationlessAccountPolicy** As Boolean

```
HRESULT WorkspacesGAEnableParticipationlessAccountPolicy(
    [out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesGAEnableParticipationlessAccountPolicy(
    [in] VARIANT_BOOL newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesGAGrantHF** As Boolean
Give each Guest Account their own home folder.

```
HRESULT WorkspacesGAGrantHF([out, retval] VARIANT_BOOL
    *pVal);
HRESULT WorkspacesGAGrantHF([in] VARIANT_BOOL newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesGAParticipationlessAccountPolicy** As [WorkspacesParticipationlessAccountPolicy](#)

```
HRESULT WorkspacesGAParticipationlessAccountPolicy(
    [out, retval] WorkspacesParticipationlessAccountPolicy
    *pVal);
HRESULT WorkspacesGAParticipationlessAccountPolicy(
    [in] WorkspacesParticipationlessAccountPolicy newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesGAParticipationlessAccountPolicyDelayInDays** As DWord

```
HRESULT
WorkspacesGAParticipationlessAccountPolicyDelayInDays([out,
    retval] ULONG *pVal);
HRESULT
WorkspacesGAParticipationlessAccountPolicyDelayInDays([in]
    ULONG newVal);
```

EFT v7.4.7 and later

- Property **WorkspacesMfaType** As [MfaType](#)

```
HRESULT WorkspacesMfaType([out, retval] MfaType* pVal);
HRESULT WorkspacesMfaType([in] MfaType newVal);
```

EFT v8.0.4 and later

- Property **WorkspacesOAIDomainsToForbidInviteeAddresses** As String

```
HRESULT WorkspacesOAIDomainsToForbidInviteeAddresses([out,
retval] BSTR *pVal);
HRESULT WorkspacesOAIDomainsToForbidInviteeAddresses([in]
BSTR newVal);
```

EFT v8 and later

- Property **WorkspacesOAIDomainsToLimitInviteeAddresses** As String

```
HRESULT WorkspacesOAIDomainsToLimitInviteeAddresses([out,
retval] BSTR *pVal);
HRESULT WorkspacesOAIDomainsToLimitInviteeAddresses([in]
BSTR newVal);
```

EFT v7.4.9 and later

- Property **WorkspacesOAIEnableDomainsToForbidInviteeAddresses** As Boolean

```
HRESULT WorkspacesOAIEnableDomainsToForbidInviteeAddresses
([out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesOAIEnableDomainsToForbidInviteeAddresses
([in] VARIANT_BOOL newVal);
```

EFT v8 and later

- Property **WorkspacesOAIEnableDomainsToLimitInviteeAddresses** As Boolean

```
HRESULT WorkspacesOAIEnableDomainsToLimitInviteeAddresses
([out, retval] VARIANT_BOOL *pVal);
HRESULT WorkspacesOAIEnableDomainsToLimitInviteeAddresses
([in] VARIANT_BOOL newVal);
```

EFT v7.4.9 and later

- Property **WorkspacesOAIMfaType** As [MfaType](#)

```
HRESULT WorkspacesOAIMfaType([out, retval] MfaType* pVal);
HRESULT WorkspacesOAIMfaType([in] MfaType newVal);
```

EFT v8.0.4 and later

- Property **WorkspacesOAIPasscodeProtection** As [WorkspacesOAIPasscodeProtectionType](#)

```
HRESULT WorkspacesOAIPasscodeProtection([out, retval]
WorkspacesOAIPasscodeProtectionType* pVal);
HRESULT WorkspacesOAIPasscodeProtection([in]
WorkspacesOAIPasscodeProtectionType pVal);
```

EFT v8.0.2 and later

- Property **WorkspacesOAISharingMode** As [WorkspacesOAISharingModeType](#)

```
HRESULT WorkspacesOAISharingMode([out, retval]
WorkspacesOAISharingModeType* pVal);
HRESULT WorkspacesOAISharingMode([in]
WorkspacesOAISharingModeType pVal);
```

EFT v7.3.3 and later

- Property **WorkspacesOAISecureMessageBodyPolicy** As [WorkspacesSecureMessageBodyPolicy](#)

Specifies the policy use, AlwaysSecure, NeverSecure, UserChoice.

```
HRESULT WorkspacesOAISecureMessageBodyPolicy([out, retval]
WorkspacesSecureMessageBodyPolicy* pVal);
HRESULT WorkspacesOAISecureMessageBodyPolicy([in]
WorkspacesSecureMessageBodyPolicy newVal);
```

EFT v8.0.5

- Property **WSSeatsRemaining** As Long (read-only)

```
HRESULT WSSeatsRemaining([out, retval] LONG* pVal);
```

EFT v7.3.3 and later

- Property **WSSeatsUsed** As Long (read-only)

```
HRESULT WSSeatsUsed([out, retval] LONG* pVal);
```

EFT v7.3.3 and later

- Property [WTCSessionsActive](#) As Long (read only)

Prior to v7.3.2, this property retrieves the number of active WTC sessions in use.

In v7.3.2 and later, this property retrieves number of used WTC client access licences.

```
HRESULT WTCSessionsActive([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "WTC sessions count: " & CStr  
(oSite.WTCSessionsActive)
```

EFT v7.3 and later

- Property [WTCSessionsRemaining](#) As Long (read only)

Prior to v7.3.2, this property retrieves the number of WTC sessions available.

In v7.3.2 and later, this property retrieves the maximum number of WTC client access licences.

```
HRESULT WTCSessionsRemaining([out, retval] LONG* pVal);
```

Example (VBScript):

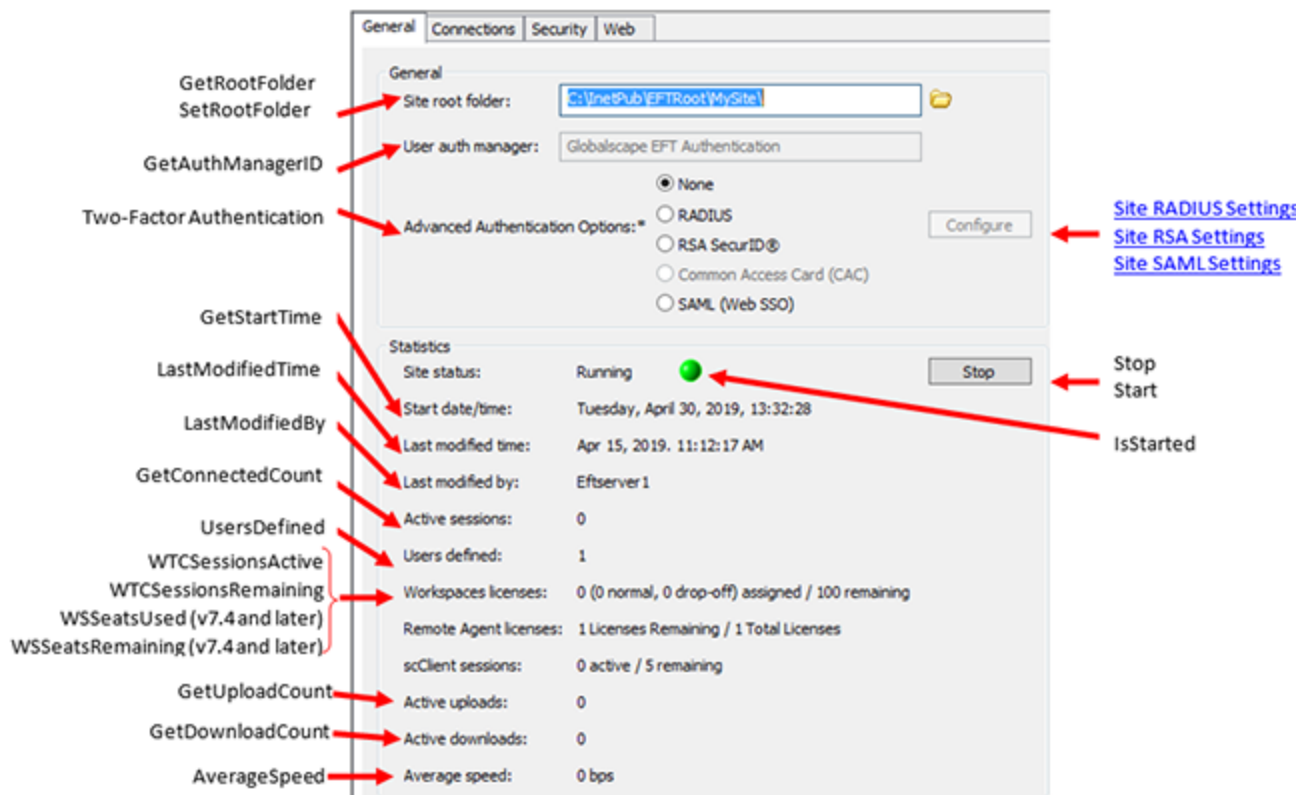
```
MsgBox "WTC sessions remaining: " & CStr  
(oSite.WTCSessionsRemaining)
```

EFT v7.3 and later

Dialog Box Equivalents

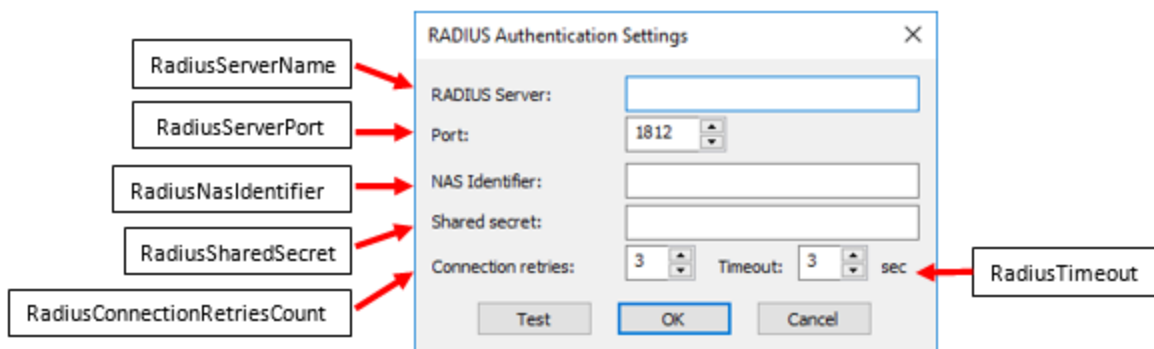
The ICISite interface properties and methods correlate to the following fields and controls in the EFT administration interface.

Site > General Tab

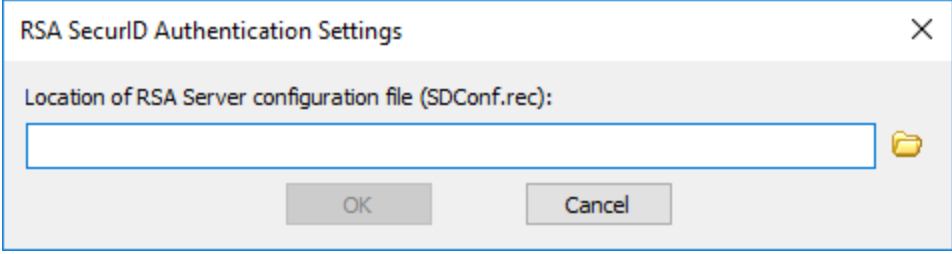


Site RADIUS Settings

The ICISite RADIUS authentication properties correlate to the following fields and controls in the EFT administration interface:



Site RSA Configuration Setting



A screenshot of a Windows-style dialog box titled "RSA SecurID Authentication Settings". The dialog has a close button (X) in the top right corner. Below the title bar, the text "Location of RSA Server configuration file (SDConf.rec):" is displayed. Underneath this text is a text input field. To the right of the input field is a yellow folder icon, indicating a file selection button. At the bottom of the dialog are two buttons: "OK" and "Cancel".

Site SAML (Web SSO) Settings

Web SSO SAML Configuration

Service Provider:

- Entity ID: (labeled WebSSOSpEntityId)
- Reserved Path: (labeled WebSSOSpReservedPath)
- ☐ Automatically redirect to the SSO IdP, bypassing the standard login page (labeled WebSSORedirectToSsoServiceBypassingLoginPage)

Identity Provider:

- Entity ID: (labeled WebSSOIDpEntityId)
- POST URL: (labeled WebSSOIDpPostUrl)
- Public Key: (labeled WebSSOIDpPublicKeyPath)

Username:

- Location in assertion: ☒ NameID ☐ Attribute (labeled WebSSONameorAttribute)
- Attribute name: (labeled WebSSOAttributeName)
- Identifier format: (labeled WebSSONameOrAttributeIdentifier)
- Parse the username using the regular expression: (labeled WebSSOUserNameRegex)
- ☒ Extend username lookup to authentication provider (labeled WebSSOExtendUsernameLookupToAuthProvider)
- ☐ Just in Time (JIT) provisioning (if not found) into: (labeled WebSSOJitUserProvisioning)

Email (JIT provisioning):

- Location in assertion: ☐ Ignore ☒ Username ☐ Attribute (labeled WebSSOJitUserProvisioningEmailLocation)
- Attribute Name: (labeled WebSSOJitUserProvisioningEmailAttribute)

Turn on Trace for SAML logger in logging.cfg

OK Cancel

Site ODBC Settings

The ODBCSettings property correlates to the Use this user attribute as logon name setting in the Windows Authentication Options dialog box. Other substrings appearing in the string depend on settings in the Windows Authentication Options dialog box. For example, if Create a virtual folder pointing to the user's home folder as defined in his/her AD profile is selected, the "CreateVirtualHF=1" substring is present in the ODBCSettings string; if Create a physical folder under the site root folder using the user's login name is selected, the "CreateVirtualHF=1" substring is not present in the ODBCSettings string.

Windows Authentication Options

Type: ☒ Active Directory ☐ Local System Accounts

Domain: ☒ Default ☐ Specify:

Group: ☒ Everyone ☐ Specify:

Advanced

Use this user attribute as logon name:

☒ Refresh user list automatically every minutes

Note: Not required. User list will update when user attempts to connect and is authenticated against the active directory server.

When creating home folders for newly added users:

☒ Create a virtual folder pointing to the user's home folder as defined in his/her AD profile

☐ Create a physical folder under the site root folder using the user's login name

Site > Connections Tab

General | **Connections** | Security | Web | Content Integrity Control

Listener Settings

Choose the protocol over which clients can connect to this site. For security best practices and for compliance with security mandates such as PCI DSS, choose only secure protocols.

Listening IP addresses:

☐ FTP ☐ FTPS (SSL / TLS) - Explicit mode ☐ FTPS (SSL / TLS) - Implicit mode ☐ SFTP (SSH2) ☐ HTTP ☒ HTTPS (SSL) ☐ AS2*

- Use yourdomain/as2 for AS2 account management
- Use yourdomain for inbound AS2 transactions

☐ Require additional factor for HTTP/S auth: *

☒ Allow Globalscape Mobile Transfer Client (MTC) over HTTPS ☐ Event Rule invoke over WS/SOAP (uses HTTPS port)

☐ Redirect all plaintext HTTP traffic to HTTPS* ☒ Enable HSTS ☐ Enable Proxy Protocol for all protocols enabled on this Site

SSL certificate settings: (Applies to HTTPS, FTPS, Web Services and AS2)

SSL security settings:

Network Usage and Security Settings

Connection limits:

IP access/ban list:

Denial of Service settings:

Listeners and Ports:

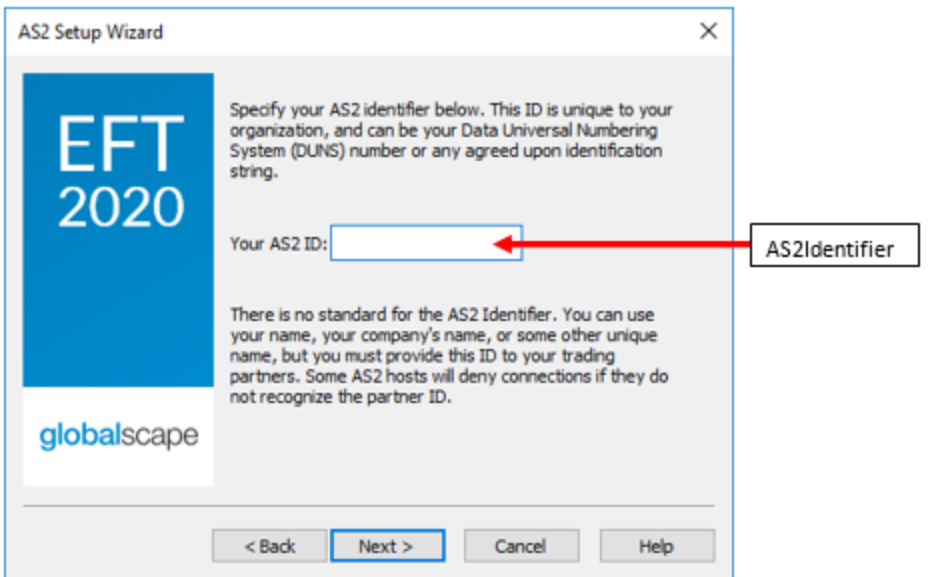
- 21: FTP
- 21: FTPS (SSL / TLS) - Explicit mode
- 990: FTPS (SSL / TLS) - Implicit mode
- 22: SFTP (SSH2)
- 80: HTTP
- 443: HTTPS (SSL)

Domain:

GetFTPAccess
SetFTPAccess
GetSSLAuth; SetSSLAuth
GetSSLImp; SetSSLImp
DisableSFTP
EnableSFTP
GetHTTPAccess
SetHTTPAccess
GetHTTPSAccess
SetHTTPSAccess
EnableAS2
EnableMfa
EnableMTC
EnableWebServices
RedirectHTTPtoHTTPS
EnableHSTS
EnableProxyProtocol

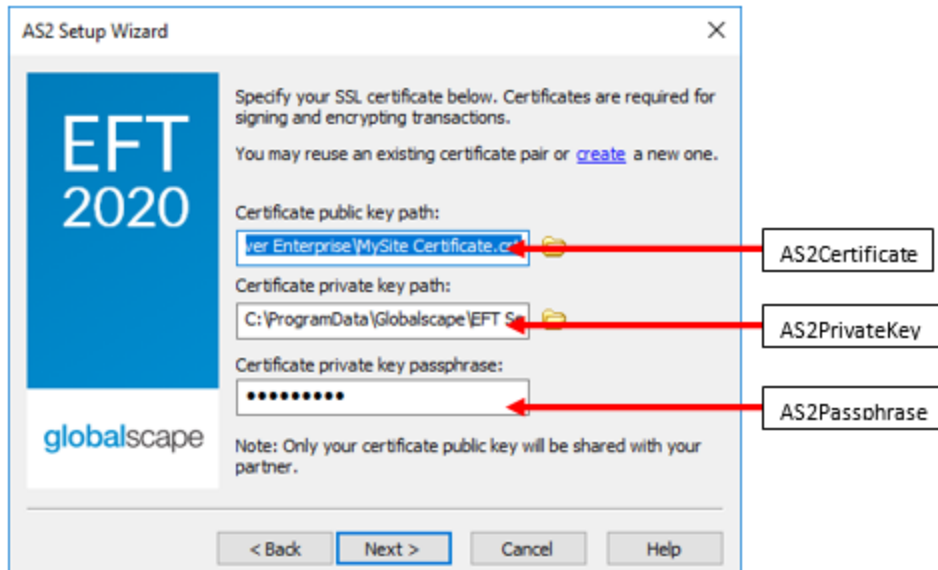
GetIP
SetIP
GetPort
SetPort
FTPSExplicitPort
FTPSImplicitPort
SFTPPort
HTTPDomain
GetHTTPPort; SetHTTPPort
GetHTTPSPort; SetHTTPSPort
SMSProviderProfile
MfaType

Site AS2 Setup Wizard



The first window of the AS2 Setup Wizard, titled "AS2 Setup Wizard", features the "EFT 2020" logo and the "globalscape" logo on the left. The main text area contains instructions: "Specify your AS2 identifier below. This ID is unique to your organization, and can be your Data Universal Numbering System (DUNS) number or any agreed upon identification string." Below this is a text input field labeled "Your AS2 ID:". A red arrow points from a label "AS2Identifier" to this field. Further instructions state: "There is no standard for the AS2 Identifier. You can use your name, your company's name, or some other unique name, but you must provide this ID to your trading partners. Some AS2 hosts will deny connections if they do not recognize the partner ID." At the bottom are four buttons: "< Back", "Next >", "Cancel", and "Help".

AS2Identifier

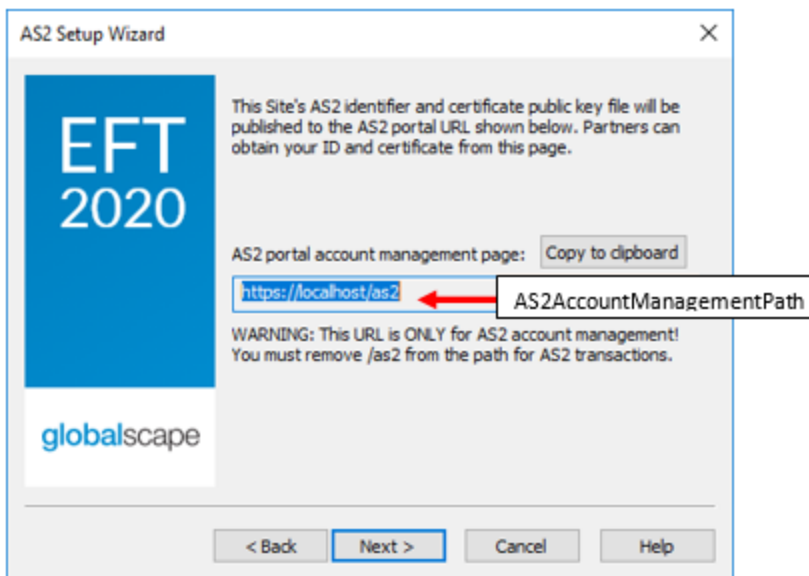


The second window of the AS2 Setup Wizard, also titled "AS2 Setup Wizard", contains the same logos. The main text area says: "Specify your SSL certificate below. Certificates are required for signing and encrypting transactions. You may reuse an existing certificate pair or [create](#) a new one." Below this are three fields: "Certificate public key path:" with a file explorer icon, "Certificate private key path:" with a folder icon, and "Certificate private key passphrase:" with a masked input field. Red arrows point from labels "AS2Certificate", "AS2PrivateKey", and "AS2Passphrase" to these respective fields. A note at the bottom states: "Note: Only your certificate public key will be shared with your partner." The same four buttons are at the bottom.

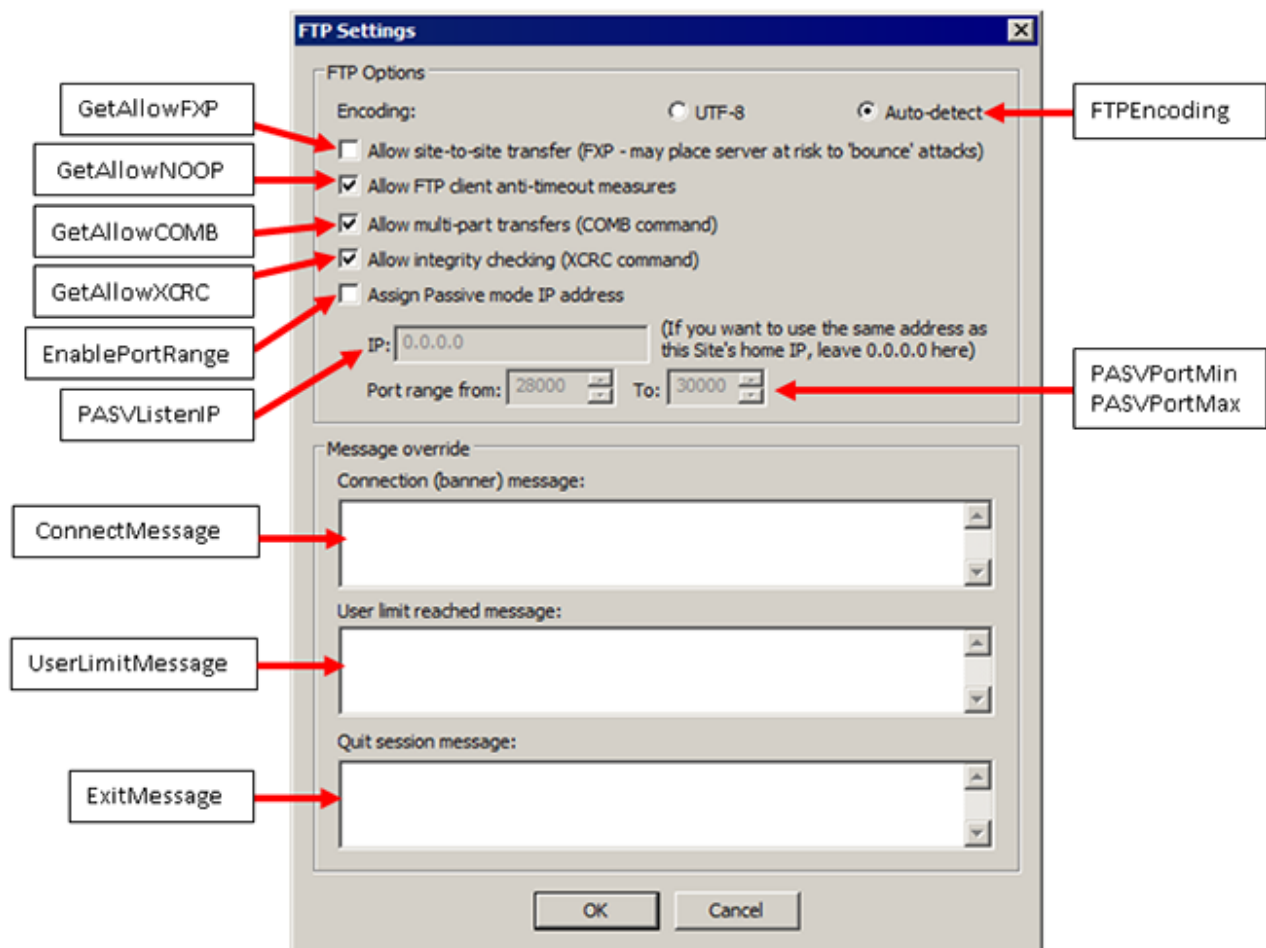
AS2Certificate

AS2PrivateKey

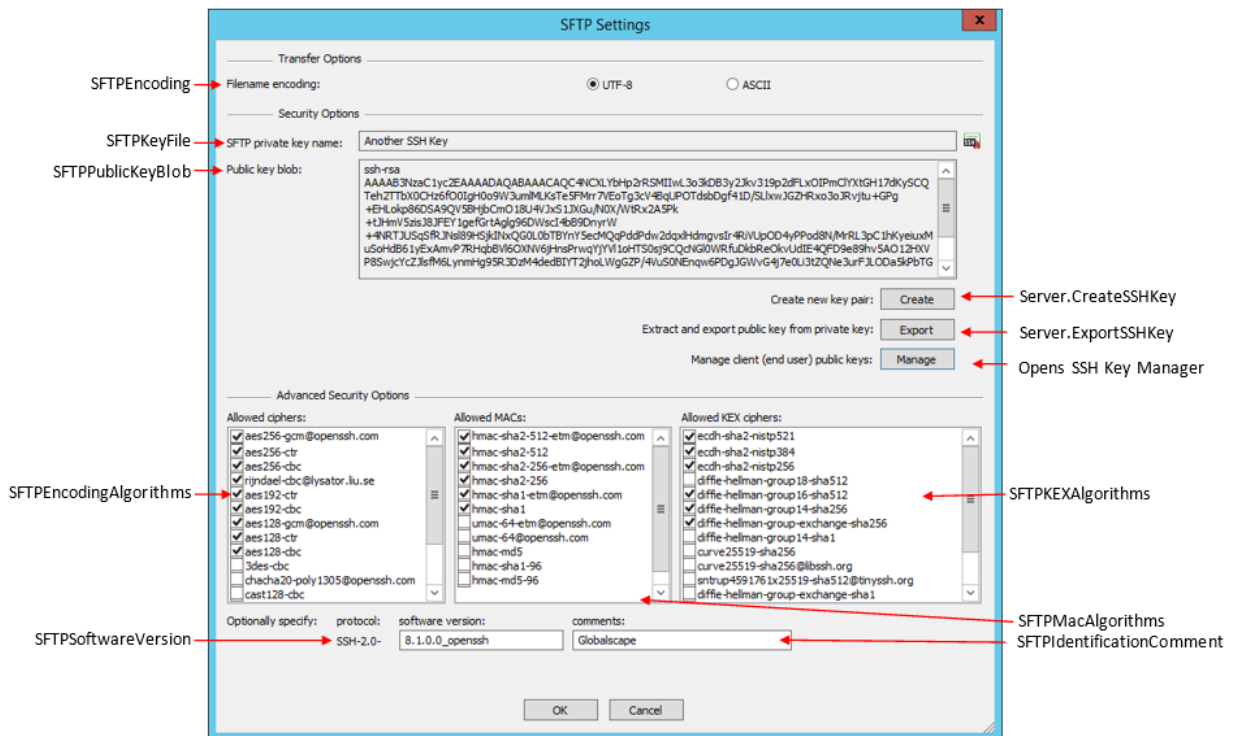
AS2Passphrase



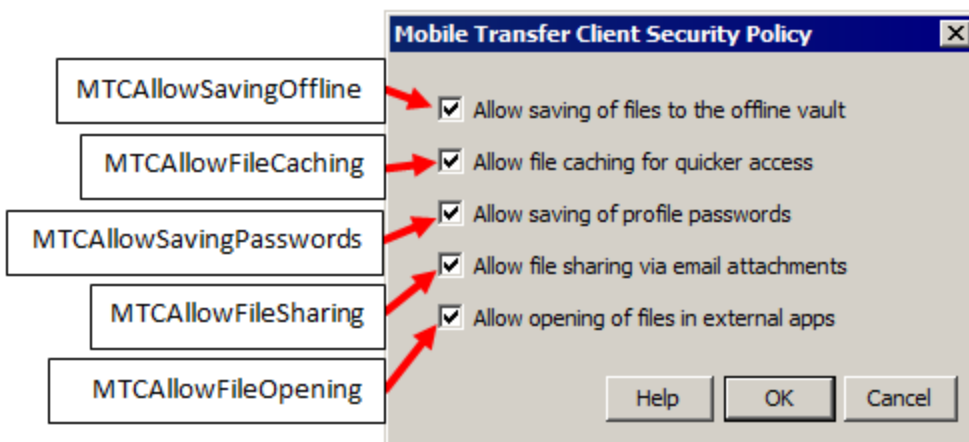
Site FTP Settings



Site SFTP Settings

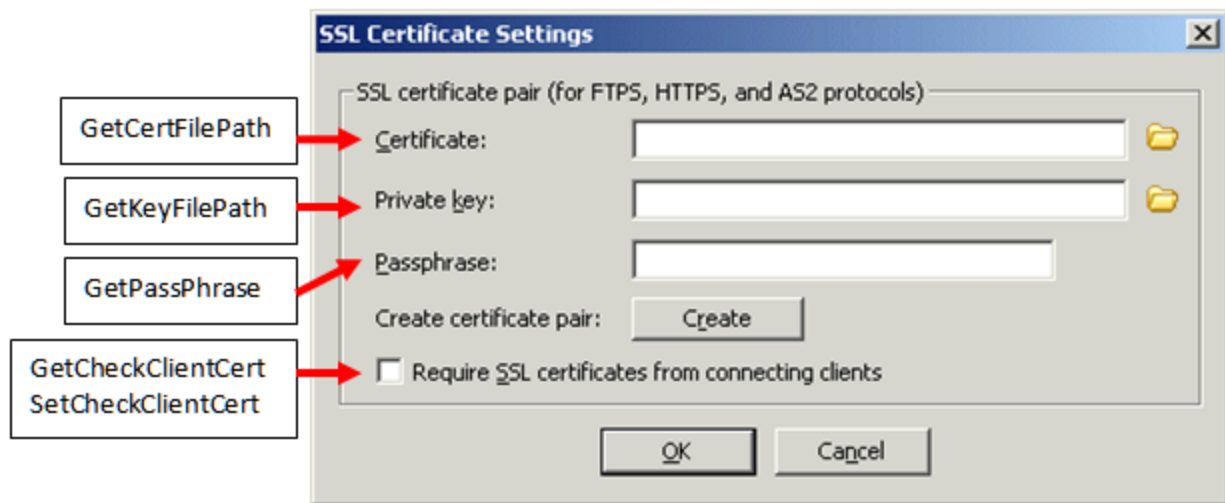


Mobile Transfer Client Security

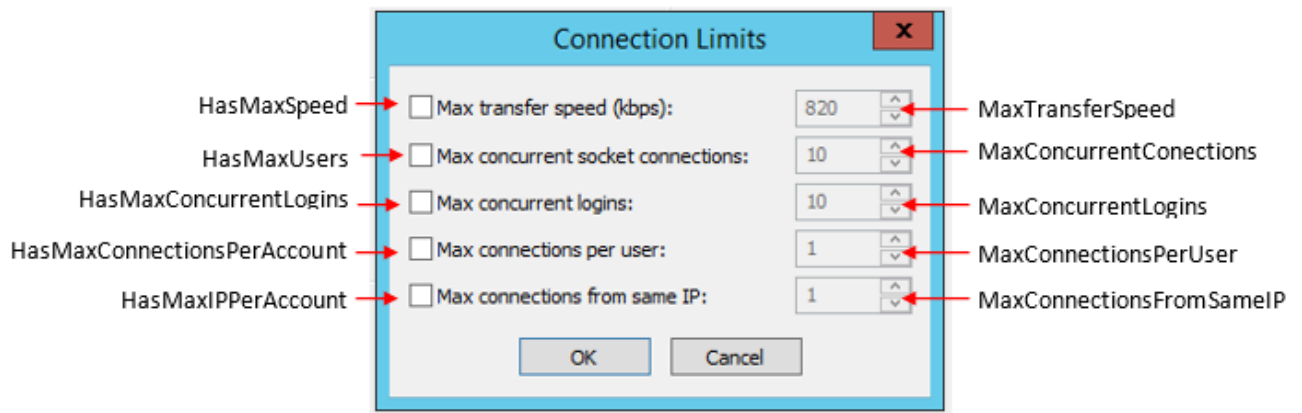


Site SSL Certificate Settings

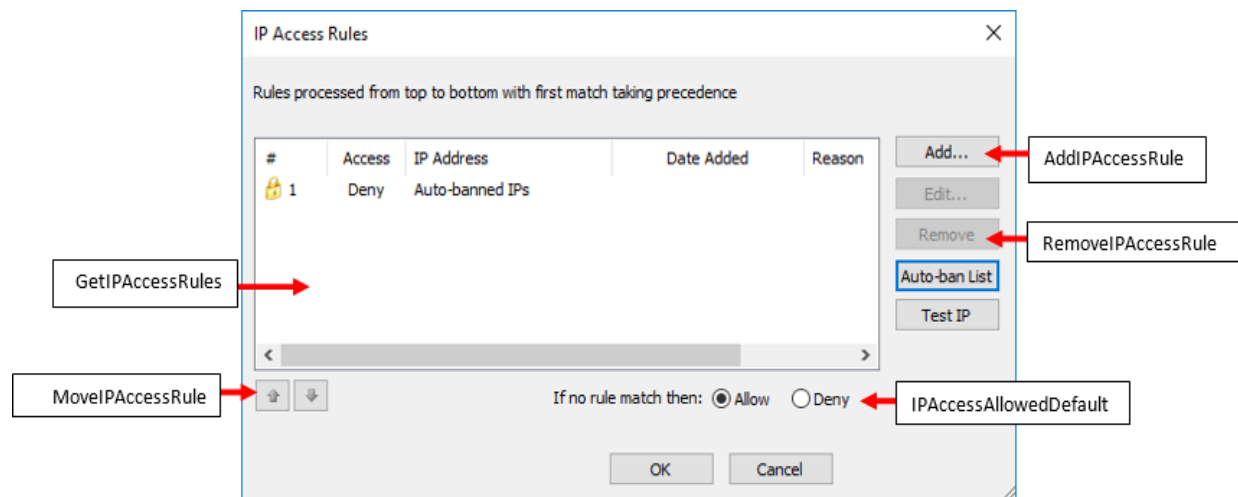
These parameters also can be viewed/set in the Administrator interface on the Connections tab of the Site.

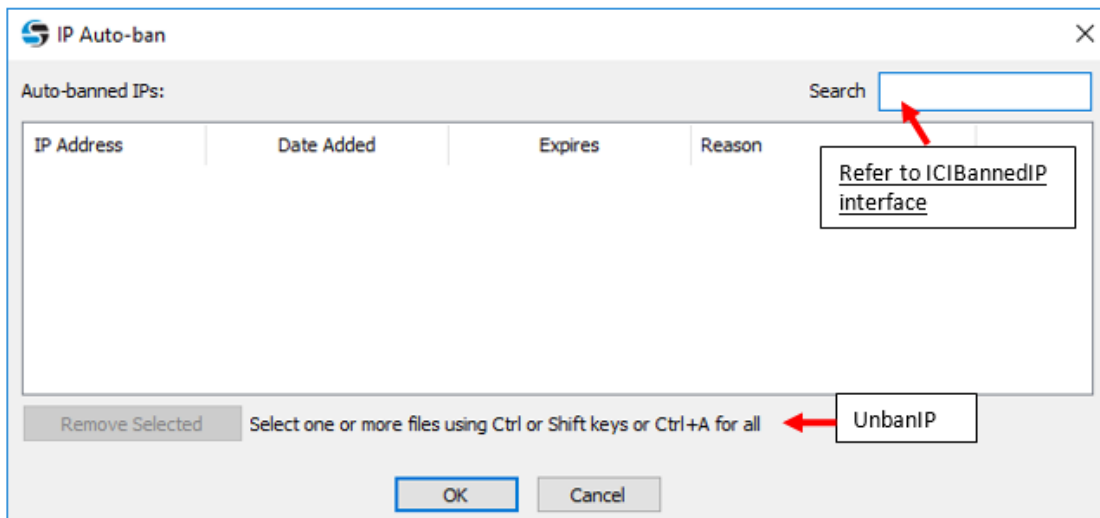


Site Connection Limits

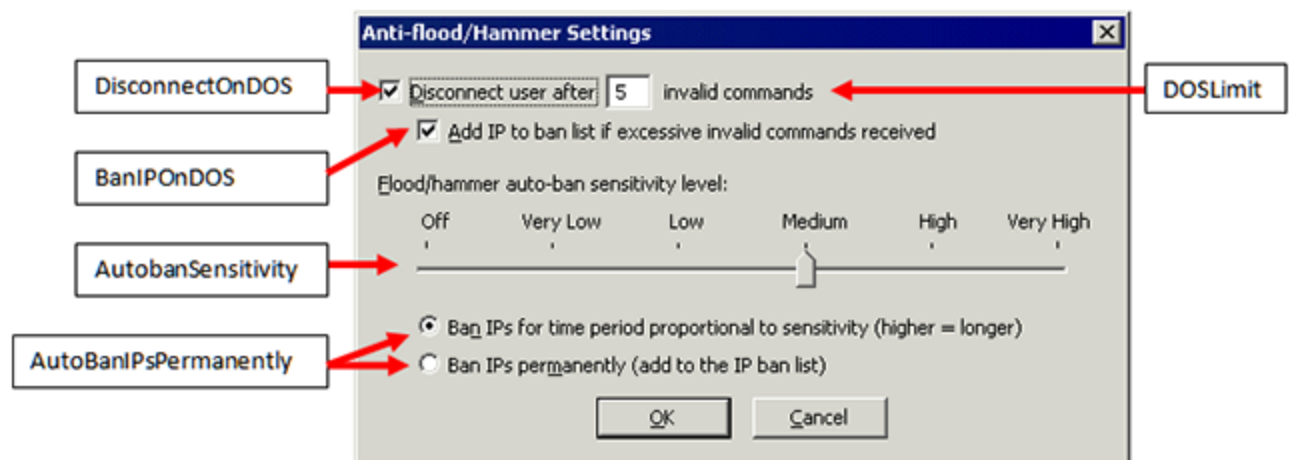


Site IP Access Rules

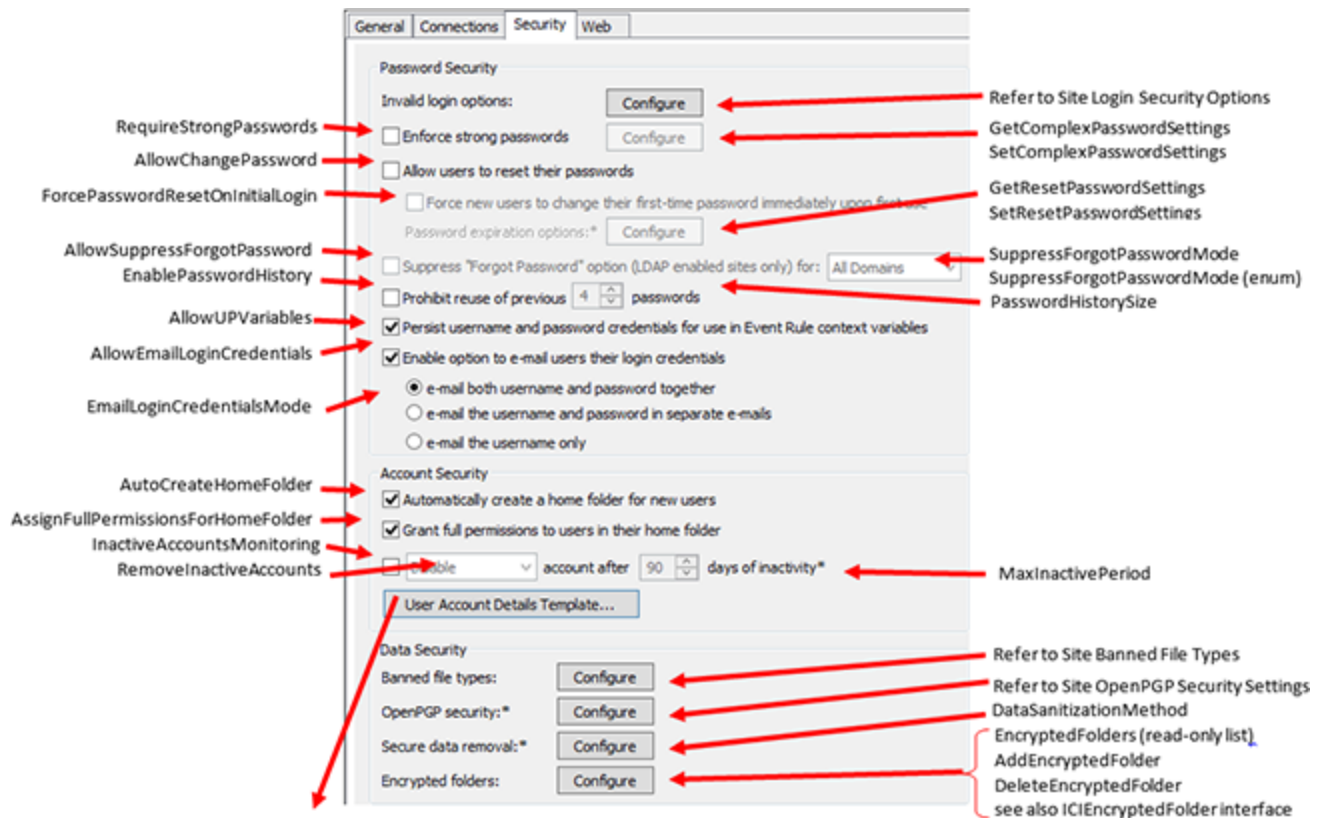




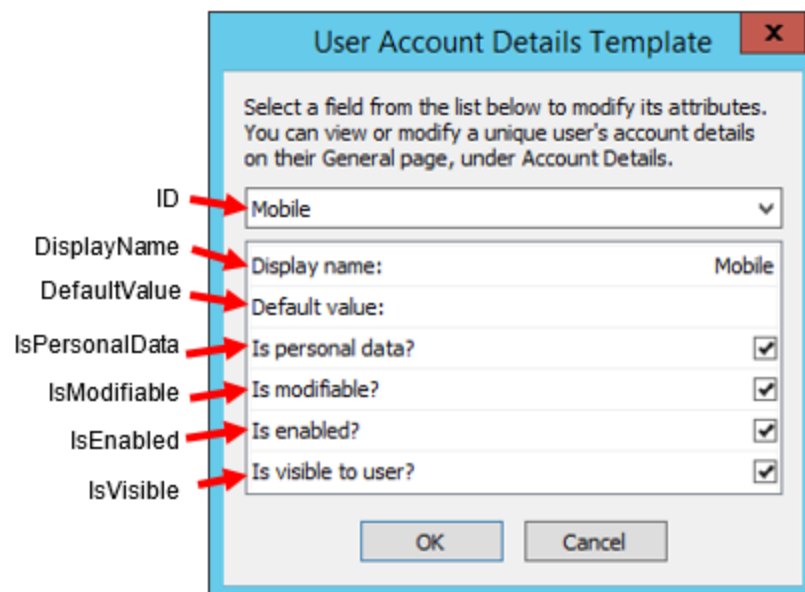
Site Anti-flood/Hammer Settings



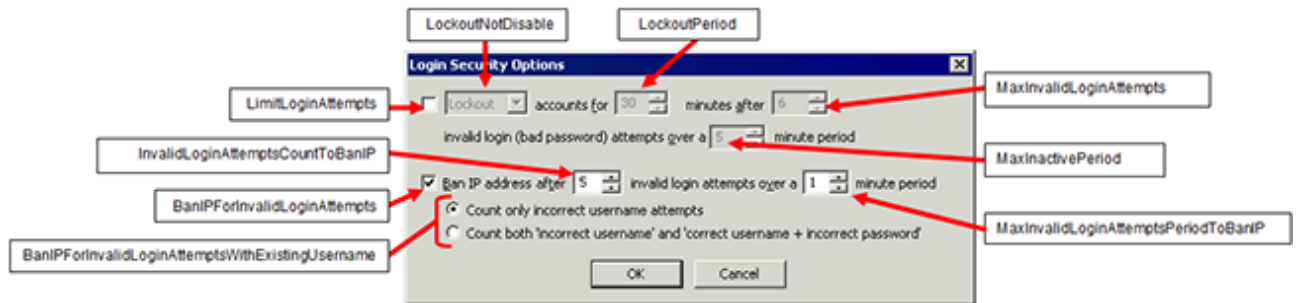
Site > Security Tab



User Account Details Template

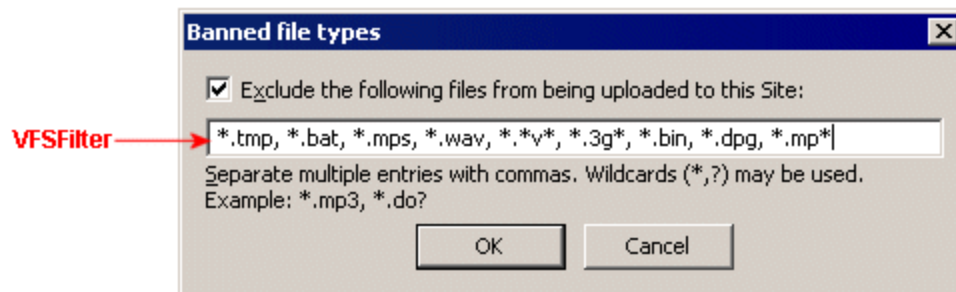


Site Invalid login (Login Security) options

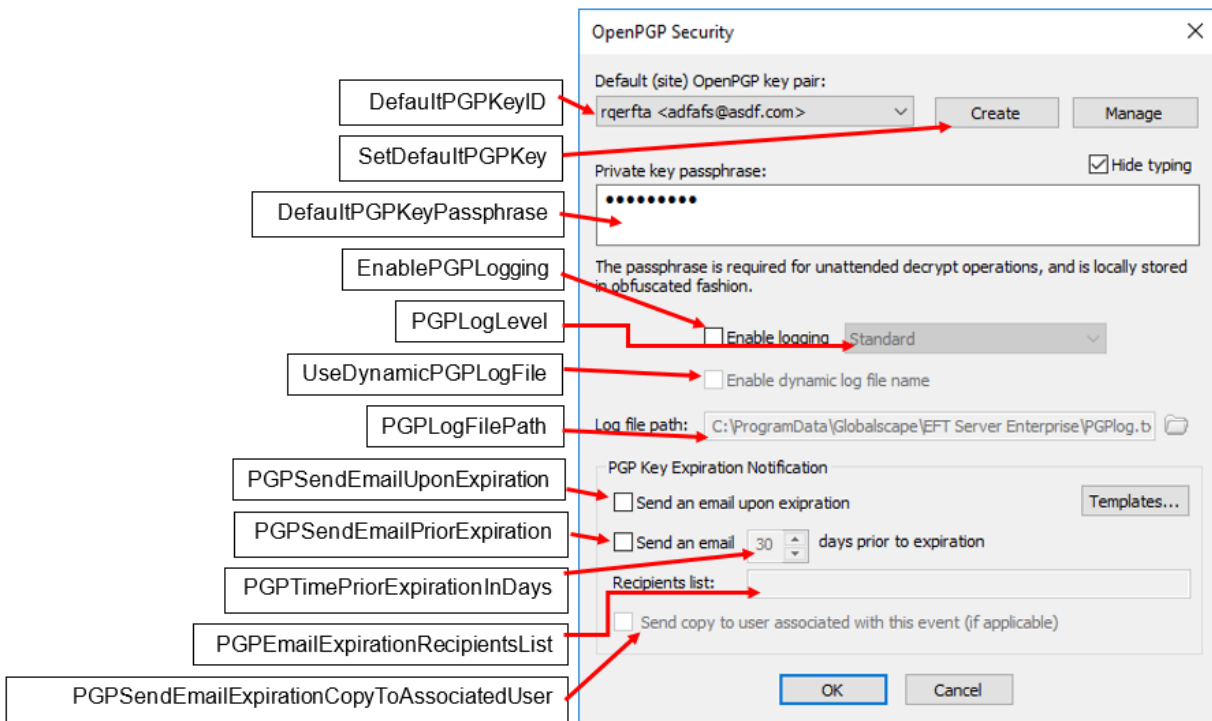


Site Banned File Types

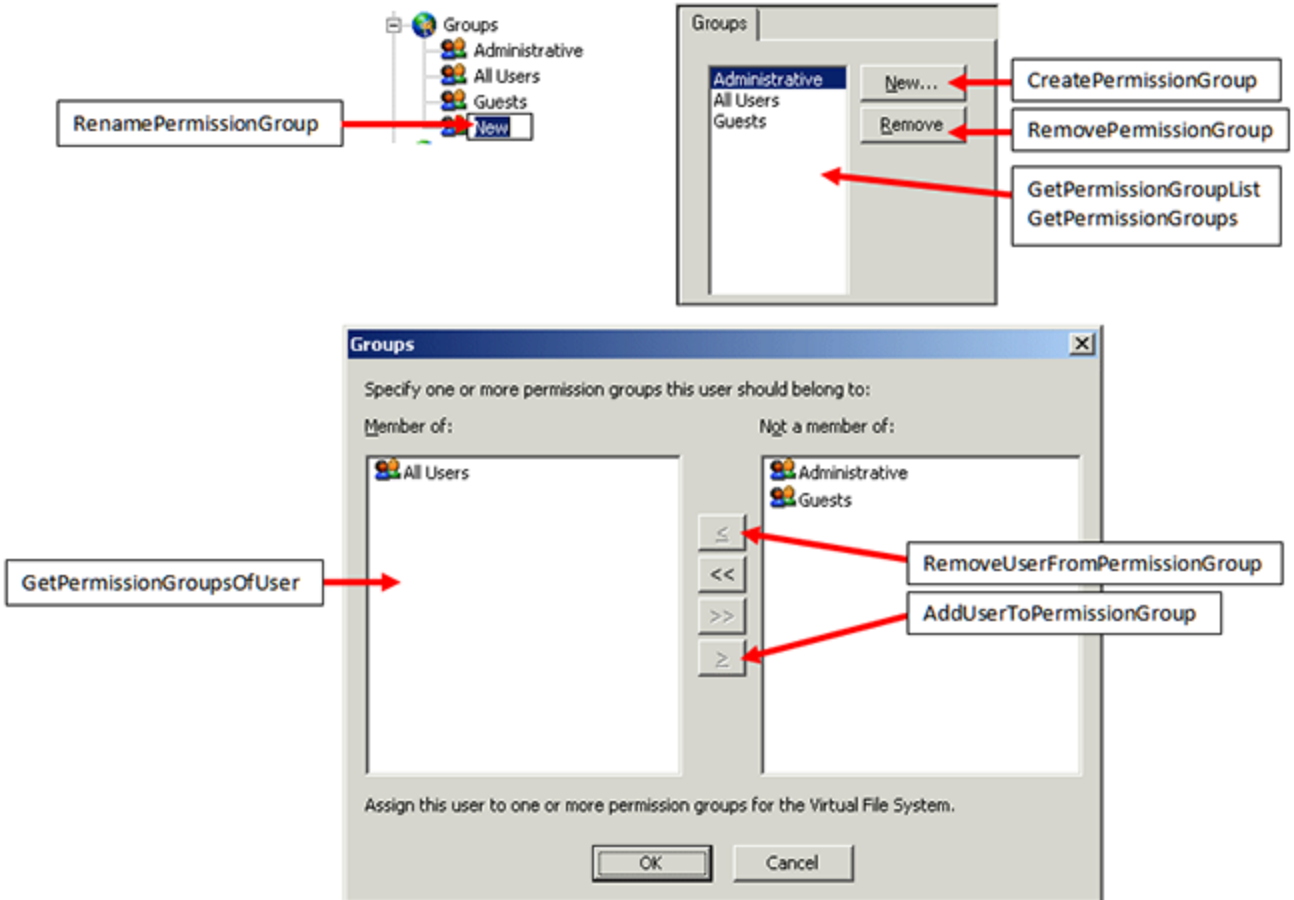
The Site data security settings can be viewed or set in the administration interface on the Site's Security tab.



Site OpenPGP Security Settings

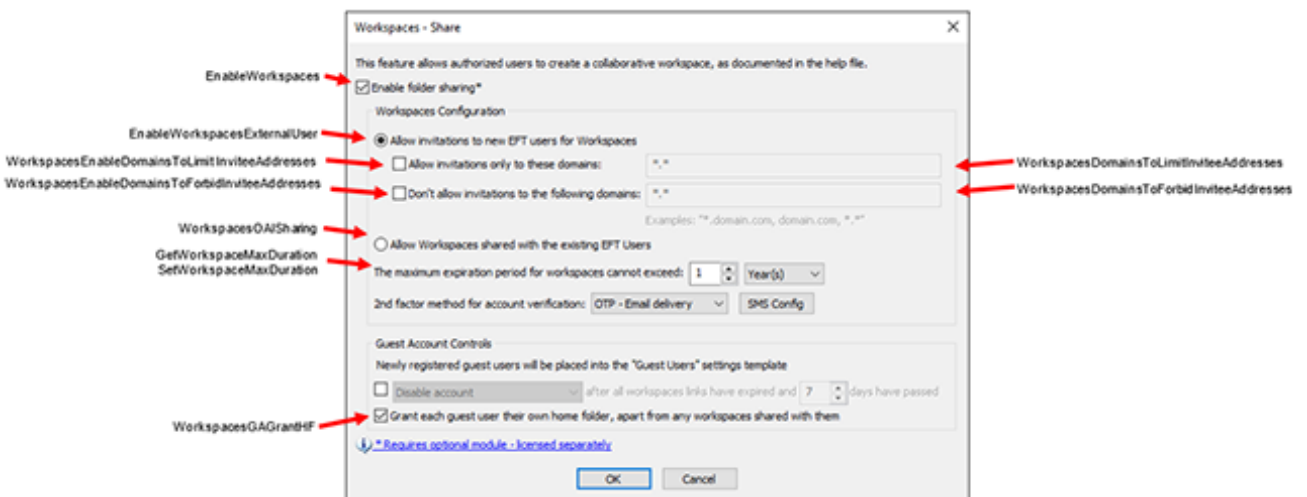


Site Groups

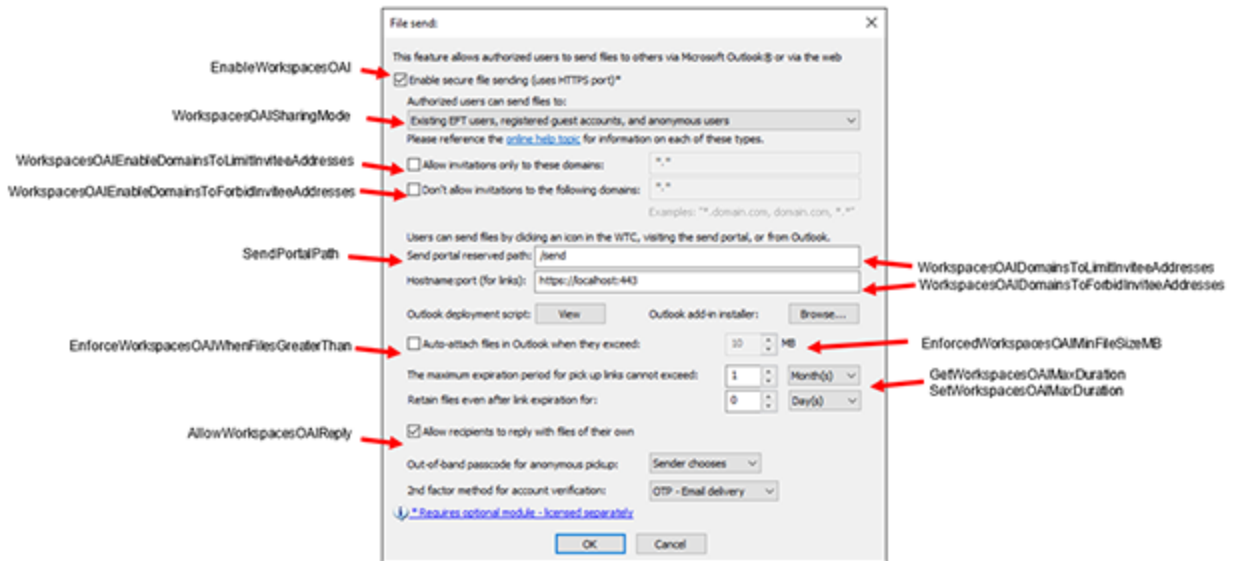


Site > Web tab (EFT v8 and later)

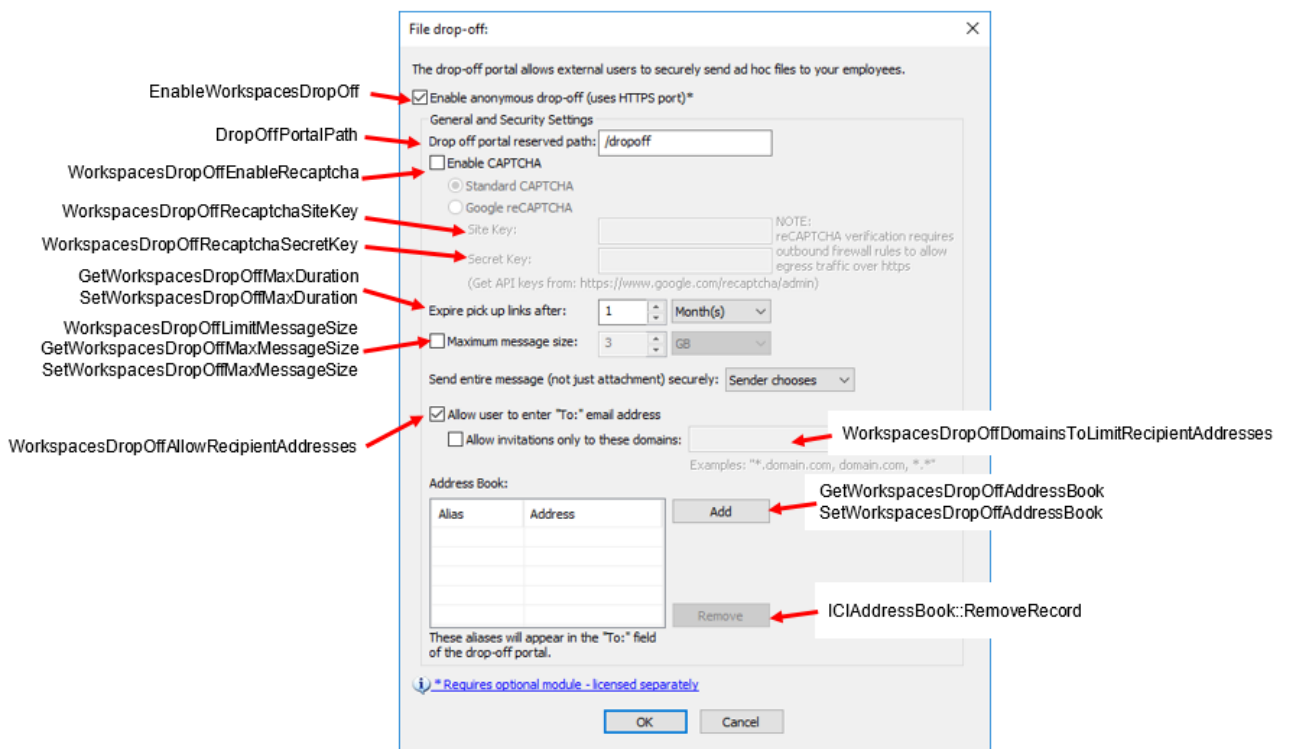
Folder sharing:



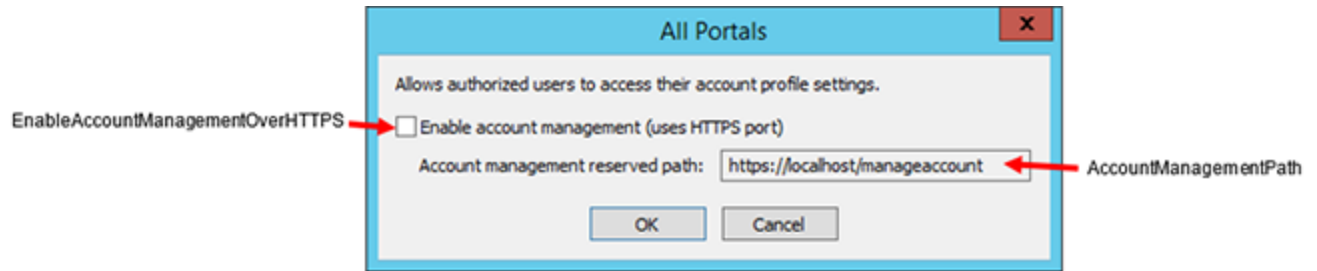
Send portal:



Drop-off portal:



Account Management:



For Privacy Policy, Terms of Service, and GDPR, refer to [GDPR and Privacy Settings](#).

ICISiteBackupInfo Interface

Use ICISiteBackupInfo interface to retrieve information about a Site that has been backed up.

```
interface ICISiteBackupInfo : IDispatch {
```

See also the [ICIBackupArchiveInfo::GetSiteBackupInfo](#), [ICIBackupArchiveInfo::SetSiteBackupInfo](#) methods.

Members (Listed alphabetically)

- Property **AuthManagerDataBase** As String

Retrieves or specifies the path to the authentication manager user database.

```
HRESULT AuthManagerDataBase([out, retval] BSTR* pVal);
HRESULT AuthManagerDataBase([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Site auth manager database: " &
oBackupSiteInfo.AuthManagerDataBase
```

Specifying:

```
oBackupSiteInfo.AuthManagerDataBase = "C:\Documents and
Settings\All Users\Application Data\GlobalSCAPE\EFT
Enterprise\MySite.aud"
```

EFT v6.3 to 7.3.3.20

- Property **AuthManagerID** As String

Retrieves or specifies the authentication manager ID.

```
HRESULT AuthManagerID([out, retval] BSTR* pVal);
HRESULT AuthManagerID([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Site auth manager: " & oBackupSiteInfo.AuthManagerID
```

Specifying:

```
oBackupSiteInfo.AuthManagerID = "LOCAL"
```

EFT v6.3 and later

- Property **DMZAddress** As String

Retrieves or specifies the DMZ Gateway IP address.

```
HRESULT DMZAddress([out,retval] BSTR* pVal);
HRESULT DMZAddress([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "DMZ IP: " & oBackupSiteInfo.DMZAddress
```

Specifying:

```
oBackupSiteInfo.DMZAddress = "192.168.0.88"
```

EFT v6.3 and later

- Property **DMZEnabled** As Boolean

Retrieves or specifies whether the DMZ Gateway is enabled.

```
HRESULT DMZEnabled([out,retval] VARIANT_BOOL* pVal);
HRESULT DMZEnabled([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled

Example (VBScript):

Retrieving:

```
MsgBox "DMZ enabled: " & CStr(oBackupSiteInfo.DMZEnabled)
```

Specifying:

```
oBackupSiteInfo.DMZEnabled = True
```

EFT v6.3 and later

- Property **DMZPort** As Long

Retrieves or specifies the DMZ Gateway port number.

```
HRESULT DMZPort([out,retval] LONG* pVal);
```

```
HRESULT DMZPort([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "DMZ port: " & oBackupSiteInfo.DMZPort
```

Specifying:

```
oBackupSiteInfo.DMZPort = 8888
```

EFT v6.3 and later

- Property **ListenIP** As String read-only

Retrieve the listener IP of the backed-up Site.

```
HRESULT ListenIP([out, retval] BSTR* pVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Old listen IP is: " & oBackupSiteInfo.ListenIP
```

EFT v6.3 and later

- Property **ListenIPs** As Variant

Retrieves or specifies the IP addresses on which the server listens.

```
HRESULT ListenIPs([out, retval] VARIANT* pVal);
HRESULT ListenIPs([in] VARIANT newVal);
```

EFT v6.4 and later

- Property **NewListenIP** As Variant

Retrieves or specifies the new listen IP address.

```
HRESULT NewListenIP([out, retval] BSTR* pVal);
HRESULT NewListenIP([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "New listen IP is: " & oBackupSiteInfo.NewListenIP
```

Specifying:

```
NewListenIP = "127.0.0.1"
```

EFT v6.3 and later

- Property **NewListenIPs** As String

Retrieves or specifies multiple new listen IP addresses.

```
HRESULT NewListenIPs([out, retval] VARIANT* pVal);
HRESULT NewListenIPs([in] VARIANT newVal);
```

EFT v6.4 and later

- Property **RestoreSite** As Boolean

Retrieves or specifies whether you need to restore the Site.

```
HRESULT RestoreSite([out, retval] VARIANT_BOOL* pVal);
HRESULT RestoreSite([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled

Example (VBScript):

Retrieving:

```
bRestore = oBackupSiteInfo.RestoreSite
```

Specifying:

```
oBackupSiteInfo.RestoreSite = True
```

EFT v6.3 and later

- Property **RestoreVFS** As Boolean

Retrieves or specifies whether to restore the VFS permissions and folders.

```
HRESULT RestoreVFS([out, retval] VARIANT_BOOL* pVal);
HRESULT RestoreVFS([in] VARIANT_BOOL newVal);
```

TRUE=Enabled; **FALSE**=Disabled**Example (VBScript):****Retrieving:**

```
bRestore = oBackupSiteInfo.RestoreVFS
```

Specifying:

```
oBackupSiteInfo.RestoreVFS = True
```

EFT v6.3 and later

- Property **RootFolder** As String

Retrieves or specifies the path to the root folder.

```
HRESULT RootFolder([out, retval] BSTR* pVal);
HRESULT RootFolder([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Site root folder is: " & oBackupSiteInfo.RootFolder
```

Specifying:

```
oBackupSiteInfo.RootFolder = "F:\ "
```

EFT v6.3 and later

- Property **SiteName** As String read-only

Retrieve the Site name.

```
HRESULT SiteName([out, retval] BSTR* pVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Site name is: " & oBackupSiteInfo.SiteName
```

EFT v6.3 and later

ICConnectedUser Interface

The ICConnectedUser interface is used to retrieve information about connected users. See also the [ICSite::ConnectedUsers](#) property.

```
interface ICConnectedUser : IDispatch
```

NOTE: "Function" indicates that the member is a method; "Property" indicates that it is a property.

Members

- Property **ConnectedAt** As Date read-only

Retrieves the date/time the user connected.

```
HRESULT ConnectedAt([out,retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Connected at: " & oCU.ConnectedAt
```

EFT v6.3 and later

- Property **ConnectionType** As String read-only

Retrieves the type of connection (e.g., FTP, HTTP, HTTPS, SFTP).

```
HRESULT ConnectionType([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Connection type: " & oCU.ConnectionType
```

EFT v6.3 and later

- Property **DataConnection** As String read-only

Retrieves the data connection state ("No data connection.", "RETRieving file", "STORing file", "RETRieving folder listing", and "N/A").

```
HRESULT DataConnection([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Data connection state: " & oCU.DataConnection
```

EFT v6.3 and later

- Property **FileName** As String read-only

Retrieves the filename currently downloading/uploading.

```
HRESULT FileName([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "File name:" & CStr(oCU.FileName)
```

EFT v6.3 and later

- Property **FileSize** As Signed Double Long read-only

Retrieves the file size of the file currently downloading.

```
HRESULT FileSize([out,retval] LONGLONG* pVal);
```

Example (VBScript):

```
MsgBox "File size: " & CStr(oCU.FileSize)
```

EFT v6.3 and later

- Property **ID** As Long read-only

Retrieves the connection ID.

```
HRESULT ID([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "ID: " & oCU.ID
```

EFT v6.3 and later

- Property **IP** As String read-only

Retrieves the user's connecting IP address.

```
HRESULT IP([out,retval] BSTR* pVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "IP: " & oCU.IP
```

EFT v6.3 and later

- Property **Log** As String

Retrieves the name of the transfer log.

```
HRESULT Log([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Log: " & oCU.Log
```

EFT v6.3 and later

- Property **Login** As String read-only

Retrieves the username.

```
HRESULT Login([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "User: " & oCU.Login
```

EFT v6.3 and later

- Property **Structure** As String read-only

Retrieves data connection structure ("File" or "N/A").

```
HRESULT Structure([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Structure: " & oCU.Structure
```

EFT v6.3 and later

- Property **TransferMode** As String read-only

Retrieves the transfer mode. (Possible values are "stream" or "N/A")

```
HRESULT TransferMode([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Structure: " & oCU.Structure
```

EFT v6.3 and later

- Property **TransferredSize** As Signed Double Long read-only

Retrieves the transferred size in bytes.

```
HRESULT TransferredSize([out,retval] LONGLONG* pVal);
```

Example (VBScript):

```
MsgBox "Transferred size: " & CStr(oCU.TransferredSize)
```

EFT v6.3 and later

- Property **TransferTime** As Long read-only

Retrieves the transfer time in seconds.

```
HRESULT TransferTime([out,retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Transfer time: " & CStr(oCU.TransferTime)
```

EFT v6.3 and later

- Property **type** As String read-only

Retrieves the mode of connection ("ASCII", "IMAGE", "EBCDIC"). Only for FTP, FTPS explicit, FTPS implicit connections; for others it returns "N/A."

```
HRESULT Type([out,retval] BSTR* pVal);
```

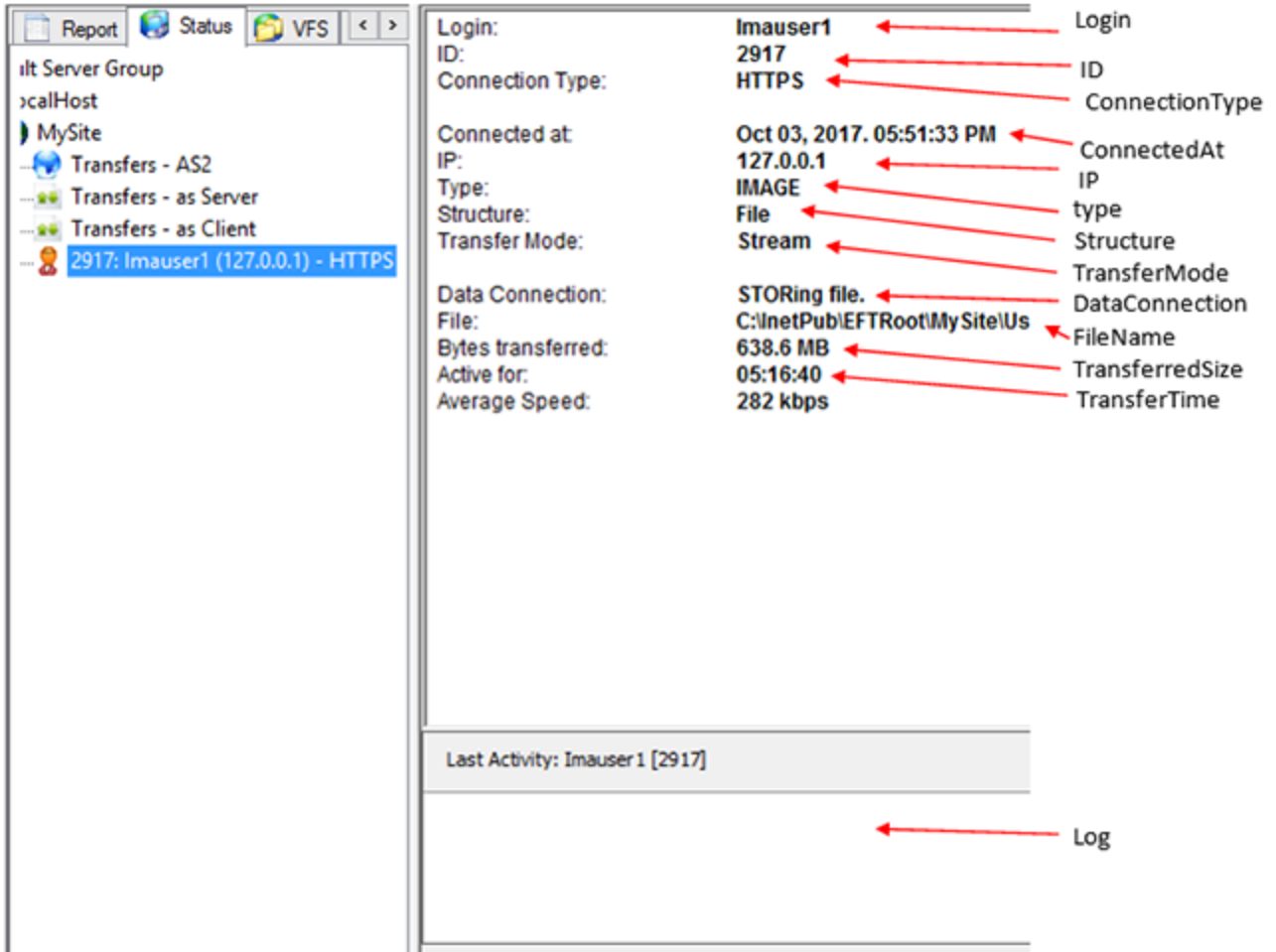
Example (VBScript):

```
MsgBox "Type: " & oCU.Type
```

EFT v6.3 and later

Dialog Box Equivalents

The ICICConnectedUser interface members correlate to the following fields and controls in the EFT administration interface. (EFT Enterprise v6.3 is shown in illustration; the properties are the same in later versions.)



ICIEncrypted Folder

The Encrypted Folder interface is used to specify a folder in which files are encrypted.

- Property **EncryptionKey** As String

Allows specifying which encryption key will be used to encrypt folder. This property can take following values:

If value is empty, then site key will be used to encrypt folder.

Example (VBScript): Specifying:

```
Set encryptedFolder = WScript.CreateObject
("SFTPCOMInterface.CIEncryptedFolder")
encryptedFolder.Path =
"C:\InetPub\EFTRoot\MySite\EncryptedFolder\"
encryptedFolder.EncryptionKey = ""

site.AddEncryptedFolder(encryptedFolder)
```

EFT v8.0.6

- Property **KeyAlias** As String

Allows specifying optional encryption key alias. Only used if the encryption key is not empty.

If value is an encryption key string, then this key will be used encrypt folder.

Example (VBScript): Specifying:

```
Set encryptedFolder = WScript.CreateObject
("SFTPCOMInterface.CIEncryptedFolder")
encryptedFolder.Path =
"C:\InetPub\EFTRoot\MySite\EncryptedFolder\"
encryptedFolder.EncryptionKey =
"634774224c47619954783532e130c35f467ce06464e86b133c44c16637
d83c3ba"
encryptedFolder.KeyAlias = "Greg's custom key"
WScript.Echo "New encryption key: " &
encryptedFolder.EncryptionKeysite.AddEncryptedFolder
(encryptedFolder)
```

Eft v8.0.6

- Property **Path** As String

```
HRESULT Path([out, retval] BSTR *pVal);
HRESULT Path([in] BSTR newVal);
```

EFT v7.4.13

ICICertInfo Interface

Certificates are digital identification documents that allow both servers and clients to authenticate each other. Server certificates contain information about your company and the organization that issued the certificate (such as Verisign or Thawte) while client

certificates contain information about the user and the organization that signed the certificate. (Dialog box equivalents are shown below the table.)

```
interface ICICertInfo : IDispatch
```

Members

- Property **Description** As String read-only

Retrieves the description of a certificate.

```
HRESULT Description([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **ID** As Long read-only

Retrieves the ID of a certificate.

```
HRESULT ID([out, retval] long *pVal);
```

EFT v6.0 and later

- Property **IssuerCName** As String read-only

Retrieves the issuer's common name on a certificate.

```
HRESULT IssuerCName([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **IssuerCountry** As String read-only

Retrieves the country of a certificate's issuer.

```
HRESULT IssuerCountry([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **IssuerOneLine** As String read-only

Retrieves all of the Issuer information (organization + unit + common name + country) concatenated in one line.

```
HRESULT IssuerOneLine([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **IssuerOrg** As String read-only

Retrieves the organization that issued the certificate.

```
HRESULT IssuerOrg([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **IssuerUnit** As String read-only

Retrieves the unit of a certificate's issuer.

```
HRESULT IssuerUnit([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **NotAfter** As String read-only

Retrieves the certificate's expiration date.

```
HRESULT NotAfter([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **NotBefore** As String read-only

Retrieves the first date the certificate became valid.

```
HRESULT NotBefore([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **SubjectCName** As String read-only

Retrieves the certificate subject's common name.

```
HRESULT SubjectCName([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **SubjectCountry** As String read-only

Retrieves the country of a certificate's subject.

```
HRESULT SubjectCountry([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **SubjectOneLine** As String read-only

Retrieves all of the Subject info (organization + unit + common name + country) concatenated in one line.

```
HRESULT SubjectOneLine([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **SubjectOrg** As String read-only

Retrieves the organization that is the subject of a certificate.

```
HRESULT SubjectOrg([out, retval] BSTR *pVal);
```

EFT v6.0 and later

- Property **SubjectUnit** As String read-only

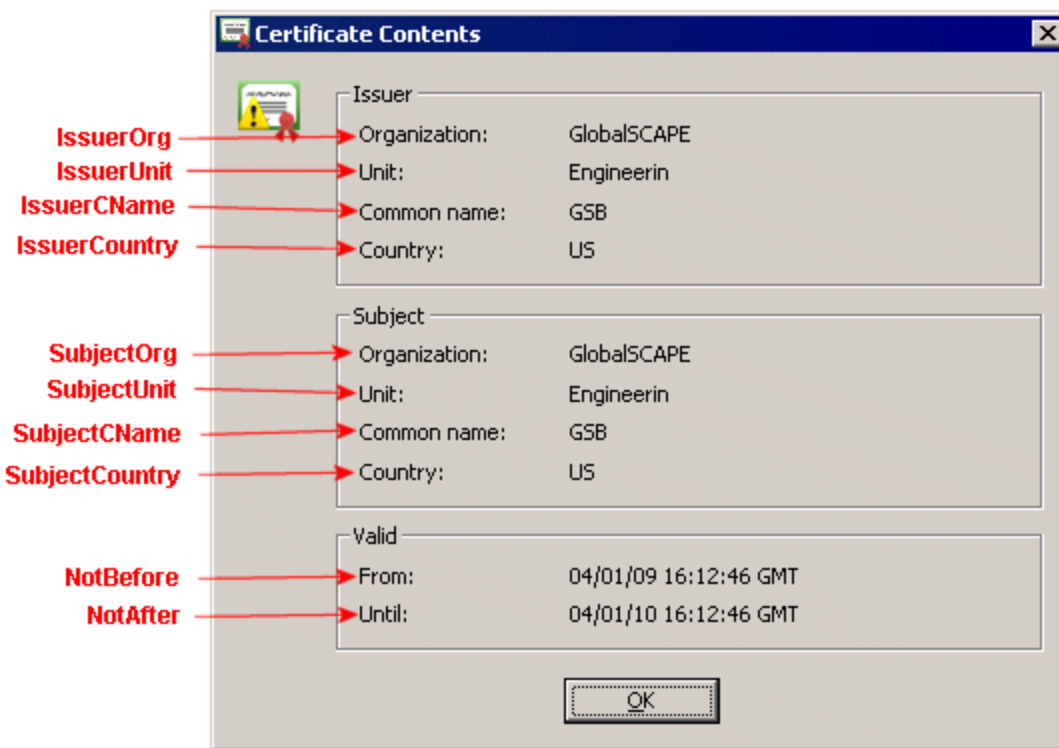
Retrieves a certificate subject's unit.

```
HRESULT SubjectUnit([out, retval] BSTR *pVal);
```

EFT v6.0 and later

Dialog Box Equivalents

The dialog box below is labeled with some of the properties that are available via the API.



ICIClientDataField Interface

The ICIClientDataField interface is used in the User Account Details Template on the Site > Security tab, which is used to specify, for privacy purposes, whether a field is enabled, visible, or is considered personal data, in addition to the field's display name and default value.

Members

- Property **DefaultValue** As String

Retrieves or modifies a specified data field's default value,

```
HRESULT DefaultValue([out, retval] BSTR *pVal);
HRESULT DefaultValue([in] BSTR val);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.DefaultValue = "Default"
```

EFT v8 and later

- Property **DisplayName** As String

Retrieves or modifies a specified data field's display name.

```
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([in] BSTR val);
```

Example (VBScript):

```
set email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.DisplayName = "Name"
```

EFT v8 and later

- Property ID As [ClientDataField](#) read-only

Retrieves the specified data field's ID.

```
HRESULT ID([out, retval] ClientDataField *pVal);
```

Example (VBScript):

```
set email = oSite.GetClientDataField(ClientDataFieldId_
Email)
wscript.echo email.ID
```

EFT v8 and later

- Property **IsEnabled** As Boolean

Retrieves or modifies the specified data field's "is enabled" attribute.

```
HRESULT IsEnabled([out, retval] VARIANT_BOOL *pVal);
HRESULT IsEnabled([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsEnabled = True
```

EFT v8 and later

- Property **IsModifiable** As Boolean

Retrieves or modifies data field's "is modifiable" attribute.

```
HRESULT IsModifiable([out, retval] VARIANT_BOOL *pVal);
HRESULT IsModifiable([in] VARIANT_BOOL newVal);
```

Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsModifiable = True
```

EFT v8 and later

- Property **IsPersonalData** As Boolean

Retrieves or modifies a specified data field's personal data attribute.

```
HRESULT IsPersonalData([out, retval] VARIANT_BOOL *pVal);
HRESULT IsPersonalData([in] VARIANT_BOOL newVal);
```


Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsPersonalData = True
```

EFT v8 and later

- Property **IsVisible** As Boolean

Retrieves modifies data field's "is visible" attribute.

```
HRESULT IsVisible([out, retval] VARIANT_BOOL *pVal);
HRESULT IsVisible([in] VARIANT_BOOL newVal);
```

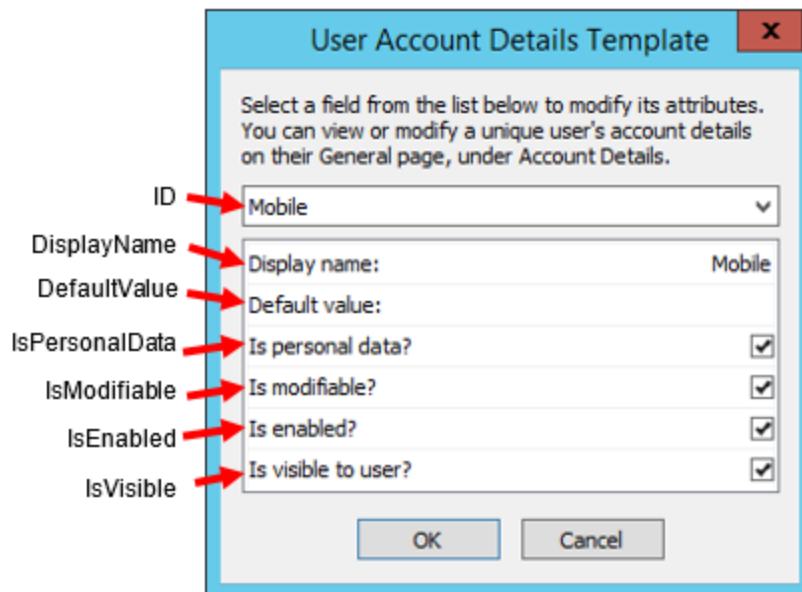
Example (VBScript):

```
get email = oSite.GetClientDataField(ClientDataFieldId_
Email)
email.IsEnabled = True
```

EFT v8 and later

Dialog Box Equivalents

Site > User Account Details Template



ICComplexPasswordSettings Interface

The IComplexPasswordSettings interface is used to specify complex password settings, such as minimum password length.

```
interface IComplexPasswordSettings : IDispatch{
```

See also:

- [Difference Between VARIANT_BOOL and SFTPAAdvBool](#)
- Scroll down to see the dialog box equivalents.
- [ICIServer::GetComplexPasswordSettings](#), [ICIServer::SetComplexPasswordSettings](#); [ICISite::GetComplexPasswordSettings](#), [ICISite::SetComplexPasswordSettings](#); [ICIClientSettings::GetComplexPasswordSettings](#), and [ICIClientSettings::SetComplexPasswordSettings](#) methods

Members

- Property **MaxRepeatingCharacters** As Long

Retrieves or specifies the maximum number of repeating characters allowed.

```
HRESULT MaxRepeatingCharacters([out, retval] LONG* pVal);
HRESULT MaxRepeatingCharacters([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Max repeating characters: " & CStr
(oCPS.MaxRepeatingCharacters)
```

Specifying:

```
oCPS.MaxRepeatingCharacters = 3
```

EFT v6.3 and later

- Property **MaxUsernameCharactersInPassword** As Long

Retrieves or specifies the maximum number of username characters allowed in passwords.

```
HRESULT MaxUsernameCharactersInPassword([out, retval] LONG*
pVal);
HRESULT MaxUsernameCharactersInPassword([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Max username characters: " & CStr
(oCPS.MaxUsernameCharactersInPassword)
```

Specifying:

```
oCPS.MaxUsernameCharactersInPassword = 4
```

EFT v6.3 and later

- Property **MinCharCategoriesInPassword** As Long

Retrieves or specifies the minimum number of character categories in passwords.

```
HRESULT MinCharCategoriesInPassword([out, retval] LONG*
pVal);
HRESULT MinCharCategoriesInPassword([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Minimal character categories in passwod is: " &
CStr(oCPS.MinCharCategoriesInPassword)
```

Specifying:

```
oCPS.MinCharCategoriesInPassword = 3
```

EFT v6.3 and later

- Property **MinPasswordLength** As Long

Retrieves or specifies the minimum password length.

```
HRESULT MinPasswordLength([out, retval] LONG* pVal);
HRESULT MinPasswordLength([in] LONG newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Minimal password length is: " & CStr
(oCPS.MinPasswordLength)
```

Specifying:

```
oCPS.MinPasswordLength = 8
```

EFT v6.3 and later

- Property **PasswordDictionaryPath** As String

Retrieves or specifies the path to the dictionary file.

```
HRESULT PasswordDictionaryPath([out, retval] BSTR* pVal);
HRESULT PasswordDictionaryPath([in] BSTR newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Dictionary path is: " & oCPS.PasswordDictionaryPath
```

Specifying:

```
oCPS.PasswordDictionaryPath = "C:\dictionary.txt"
```

EFT v6.3 and later

- Property **ProhibitDictionaryWordsInPassword** As Boolean

Retrieves or specifies whether dictionary words are prohibited in passwords.

```
HRESULT ProhibitDictionaryWordsInPassword([out, retval]
VARIANT_BOOL* pVal);
HRESULT ProhibitDictionaryWordsInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Prohibit dictionary is enable: " & CStr
(oCPS.ProhibitDictionaryWordsInPassword)
```

Specifying:

```
oCPS.ProhibitDictionaryWordsInPassword = True
```

EFT v6.3 and later

- Property **ProhibitReverseDictionaryWordsInPassword** As Boolean

Retrieves or specifies whether reverse dictionary words are prohibited in passwords.

```
HRESULT ProhibitReverseDictionaryWordsInPassword([out,
retVal] VARIANT_BOOL* pVal);
HRESULT ProhibitReverseDictionaryWordsInPassword([in]
VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Prohibit reverse dictionary is enable: " &
CStr(oCPS.ProhibitReverseDictionaryWordsInPassword)
```

Specifying:

```
oCPS.ProhibitReverseDictionaryWordsInPassword = True
```

EFT v6.3 and later

- Property **RequireDigitsInPassword** As Boolean

Retrieves or specifies whether digits are required in passwords.

```
HRESULT RequireDigitsInPassword([out, retVal] VARIANT_BOOL*
pVal);
HRESULT RequireDigitsInPassword([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Digits is required: " & CStr
(oCPS.RequireDigitsInPassword)
```

Specifying:

```
oCPS.RequireDigitsInPassword = False
```

EFT v6.3 and later

- Property **RequireLowerCaseInPassword** As Boolean

Retrieves or specifies whether lowercase is required in passwords.

```
HRESULT RequireLowerCaseInPassword([out, retval] VARIANT_BOOL* pVal);
HRESULT RequireLowerCaseInPassword([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Lower case is required: " & CStr
(oCPS.RequireLowerCaseInPassword)
```

Specifying:

```
oCPS.RequireLowerCaseInPassword = False
```

EFT v6.3 and later

- Property **RequireNonAlphaNumericInPassword** As Boolean

Retrieves or specifies whether non-alphanumeric characters are required in passwords.

```
HRESULT RequireNonAlphaNumericInPassword([out, retval]
VARIANT_BOOL* pVal);
HRESULT RequireNonAlphaNumericInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Non-alphanumeric is required: " & CStr
(oCPS.RequireNonAlphaNumericInPassword)
```

Specifying:

```
oCPS.RequireNonAlphaNumericInPassword = True
```

EFT v6.3 and later

- Property **RequireUnicodeInPassword** As Boolean

```
HRESULT RequireUnicodeInPassword([out, retval] VARIANT_BOOL*
pVal);
HRESULT RequireUnicodeInPassword([in] VARIANT_BOOL newVal);
```

EFT v7.4 and later

- Property **RequireUpperCaseInPassword** As Boolean

Retrieves or specifies whether uppercase is required in passwords.

```
HRESULT RequireUpperCaseInPassword([out, retval] VARIANT_
BOOL* pVal);
HRESULT RequireUpperCaseInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Upper case is required: " & CStr
(oCPS.RequireUpperCaseInPassword)
```

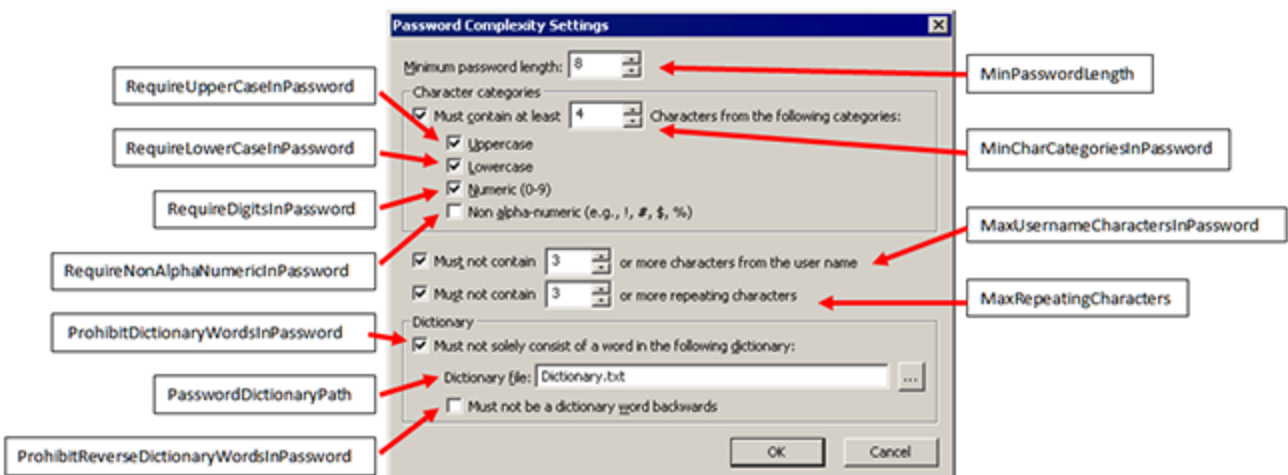
Specifying:

```
oCPS.RequireUpperCaseInPassword = False
```

EFT v6.3 and later

Dialog Box Equivalents

The ICIComplexPasswordSettings interface members correlate to the following fields and controls in the EFT administration interface. (Select the "Enforce strong passwords" check box.)



ICIPermission Interface

The VFS permissions can be viewed/set in the administration interface on the VFS tab. The ICIPermission interface represents the permissions for folders. Obtain it by using the GetFolderPermissions method of the [ICISite](#) interface. You can also use the GetBlankPermission method to start with no permissions.

NOTE: Changing properties in the IPermission interface has no effect until you call the ICISite::SetPermission method. Refer to the [ICISite::SetPermission](#) method for an example.

```
interface IPermission : IDispatch
```

Example:

```
aPerm = s.GetFolderPermissions("\pub\")
Set p = s.GetBlankPermission("\pub\", "Administrative")
```

Properties

- Property **Client** As String read-only

Retrieves the client name.

```
HRESULT Client([out, retval] BSTR *pVal);
```

All versions

- Property **DirCreate** As Boolean

Retrieves or specifies whether the account has "create directory" permission.

```
HRESULT DirCreate([out, retval] VARIANT_BOOL *pVal);
HRESULT DirCreate([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **DirDelete** As Boolean

Retrieves or specifies whether the account has "delete directory" permission.

```
HRESULT DirDelete([out, retval] VARIANT_BOOL *pVal);
HRESULT DirDelete([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **DirList** As Boolean

Retrieves or specifies whether the account has "directory list" permission.

```
HRESULT DirList([out, retval] VARIANT_BOOL *pVal);
HRESULT DirList([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **DirShowHidden** As Boolean

Retrieves or specifies whether the account has "show hidden directory" permission.

```
HRESULT DirShowHidden([out, retval] VARIANT_BOOL *pVal);
HRESULT DirShowHidden([in] VARIANT_BOOL newVal);
```

True = Displayed; **False** = Hidden

You must allow folder listing (**DirList**) before you can choose to hide or display hidden files.

All versions

- Property **DirShowInList** As Boolean

Retrieves or specifies whether the account has "show directory in list" permission.

```
HRESULT DirShowInList([out, retval] VARIANT_BOOL *pVal);
HRESULT DirShowInList([in] VARIANT_BOOL newVal);
```

True = Displayed; **False** = Hidden

You must allow folder listing (**DirList**) before you can choose to hide or display read-only files.

All versions

- Property **DirShowReadOnly** As Boolean

Retrieves or specifies whether the account has "show read only" directory permission.

```
HRESULT DirShowReadOnly([out, retval] VARIANT_BOOL *pVal);
HRESULT DirShowReadOnly([in] VARIANT_BOOL newVal);
```

True = Displayed; **False** = Hidden

You must allow folder listing (**DirList**) before you can choose to hide or display read-only files.

All versions

- Property **FileAppend** As Boolean

Retrieves or specifies whether the account has "append file" permission.

```
HRESULT FileAppend([out, retval] VARIANT_BOOL *pVal);
HRESULT FileAppend([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **FileDelete** As Boolean

Retrieves or specifies whether the account has "delete file" permission.

```
HRESULT FileDelete([out, retval] VARIANT_BOOL *pVal);
HRESULT FileDelete([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **FileDownload** As Boolean

Retrieves or specifies whether the account has "download file" permission.

```
HRESULT FileDownload([out, retval] VARIANT_BOOL *pVal);
HRESULT FileDownload([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **FileRename** As Boolean

Retrieves or specifies whether the account has "rename file" permission.

```
HRESULT FileRename([out, retval] VARIANT_BOOL *pVal);
HRESULT FileRename([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **FileUpload** As Boolean

Retrieves or specifies whether the account has "upload file" permission.

```
HRESULT FileUpload([out, retval] VARIANT_BOOL *pVal);
HRESULT FileUpload([in] VARIANT_BOOL newVal);
```

True = Allowed; **False** = Prohibited

All versions

- Property **Folder** As String read-only

Retrieves the folder name.

```
HRESULT Folder([out, retval] BSTR *pVal);
```

All versions

- Property **InheritedFrom** As String read-only

Retrieves Settings Template or Site inherited from. If this permissions isn't inherited, then **InheritedFrom** = Folder.

```
HRESULT InheritedFrom([out, retval] BSTR *pVal);
```

Example (VBScript):

```
MsgBox "Folder: " & oPerm.Folder & " inherited permission  
from: " & oPerm.InheritedFrom
```

EFT v6.3 and later

- Property **IsGroup** As Boolean read-only

Retrieves TRUE if this is a group permission.

```
HRESULT IsGroup([out, retval] VARIANT_BOOL* pVal);
```

Example (VBScript):

```
MsgBox "Is permission group: " & CStr(oPerm.IsGroup)
```

EFT v6.3 and later

- Property **IsInherited** As Boolean read-only

Retrieves TRUE if it is inherited.

```
HRESULT IsInherited([out, retval] VARIANT_BOOL* pVal);
```

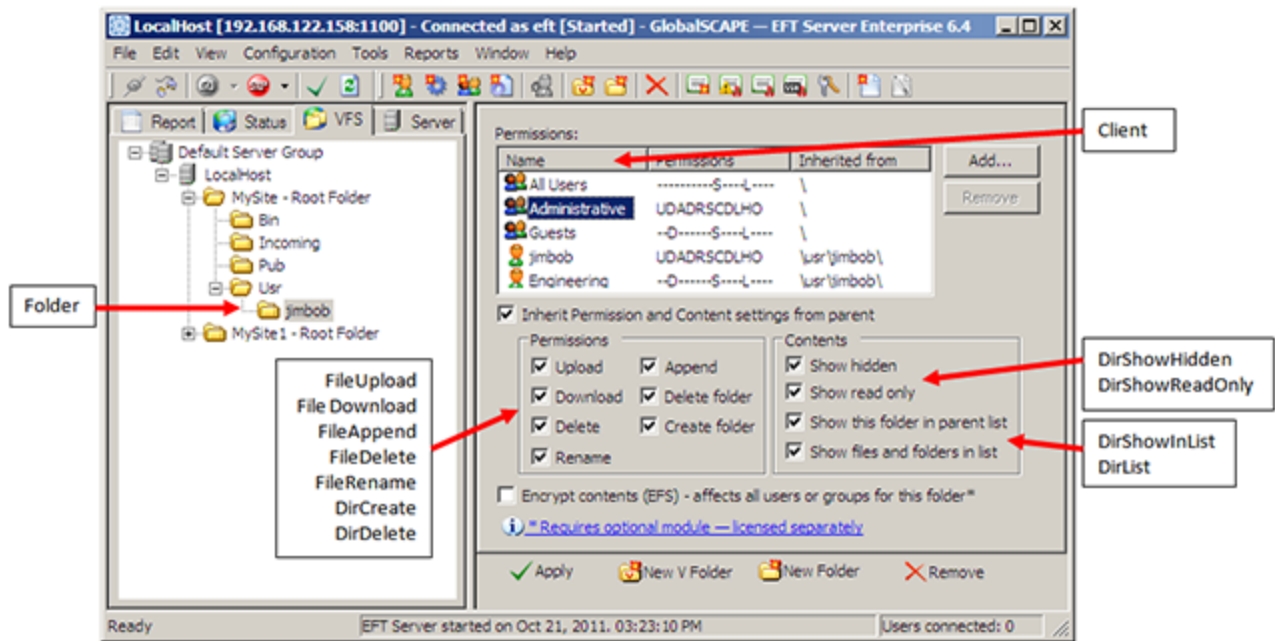
Example (VBScript):

```
MsgBox "Is inherited: " & CStr(oPerm.IsInherited)
```

EFT v6.3 and later

Dialog Box Equivalents

The illustration below is labeled with the methods that are available in the Permission interface.



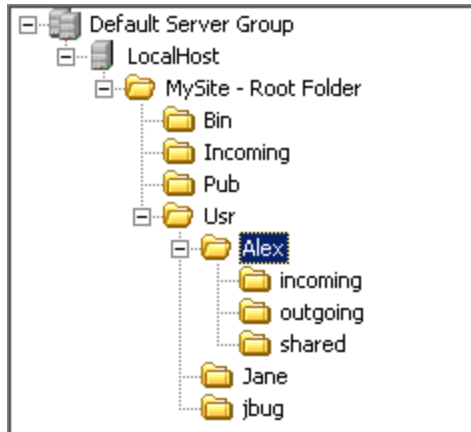
Difference between DirList and DirShowInList Permission

Within the VFS system, the DirList file permission and the DirShowInList folder permission can confuse users as to the intended operation of the server. At first glance, it would appear that DirList refers to the ability to see files in the directory and DirShowInList would refer to the ability to see folders in the directory, but this is incorrect.

The DirList file permission refers to the user's ability to retrieve a directory listing from the Server. If this option is enabled, the user is able to see a list of files in the directory. If it is disabled, no files or directories will be visible.

The DirShowInList permission is a bit more complex in that the permission corresponds to whether the directory will show up when a directory listing is retrieved in the parent directory. The interaction between inheritance and this setting can therefore lead to a bit of confusion.

Let's use the following folder structure as an example:



If you select the folder Alex and disable the DirShowInList permission for folder Jane, then when user Jane navigates to the /Usr/ directory and retrieves a directory listing, the folder Alex will NOT appear in her directory listing. To complicate matters, however, Jane still has access to the folder and if someone tells her to manually navigate to that folder she will still be able to perform actions within the Alex folder (provided she has appropriate permissions). In this scenario, by default, when user Jane retrieves a directory listing from within the folder Alex, she will only see a list of files in the folder; she will not see any subfolders in the folder. Why? Because the subfolders (incoming, outgoing, and shared) have all inherited the disabled DirShowInList setting!

Summary:

- The DirList permission applies to both files and folders in the current folder.
- The DirShowInList permission applies to the visibility of the current folder in its parent directory.
- Default inheritance rules will not allow the display of subfolders of a DirShowInList-disabled folder.

ICIResetPasswordSettings Interface

Use the ICIResetPasswordSettings interface to configure password reset details. (Use [ICISite::GetResetPasswordSettings](#) and [ICISite::SetResetPasswordSettings](#) to retrieve and set the password settings for a Site.)

```
interface ICIResetPasswordSettings : IDispatch {
```

See also see [ICISite::GetResetPasswordSettings](#), [ICISite::SetResetPasswordSettings](#), [ICIClientSettings::GetResetPasswordSettings](#), [ICIClientSettings::SetResetPasswordSettings](#).

Properties

- Property **DaysPriorPasswordExpirationToRemindUser** As Long

For an example using this property, refer to the GetCertInfo script in [Script Examples](#).

Retrieves or specifies the number of days prior to expiration to notify the user. If "0" is specified, no notification is sent. (Equivalent to the check box in the dialog box not being selected.)

```
HRESULT DaysPriorPasswordExpirationToRemindUser([out,retval]
LONG* pVal);
HRESULT DaysPriorPasswordExpirationToRemindUser([in] LONG
newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Remind before expiration: " & CStr
(oRPS.DaysPriorPasswordExpirationToRemindUser)
```

Specifying:

```
oRPS.DaysPriorPasswordExpirationToRemindUser = 3
```

EFT v6.3 and later

- Property **MaxPasswordAgeDays** As Long

For an example using this property, refer to the GetCertInfo script in [Script Examples](#).

Retrieves or specifies the number of days at which to expire the password. If "0" is specified, the password does not expire. (Equivalent to the check box in the dialog box not being selected.)

```
HRESULT MaxPasswordAgeDays([out,retval] LONG* pVal);
HRESULT MaxPasswordAgeDays([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Max password age is: " & CStr
(oRPS.MaxPasswordAgeDays)
```

Specifying:

```
oRPS.MaxPasswordAgeDays = 100
```

EFT v6.3 and later

- Property **SendEmailBeforeExpiration** As Boolean

For an example using this property, refer to the GetCertInfo script in [Script Examples](#).

Retrieves or specifies whether to send an email notification before the password expires.

```
HRESULT SendEmailBeforeExpiration([out,retval] VARIANT_BOOL*
pVal);
HRESULT SendEmailBeforeExpiration([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Send email before expiration: " & CStr
(oRPS.SendEmailBeforeExpiration)
```

Specifying:

```
oRPS.SendEmailBeforeExpiration = False
```

EFT v6.3 and later

- Property **SendEmailUponExpiration** As Boolean

For an example using this property, refer to the GetCertInfo script in [Script Examples](#).

Retrieves or specifies whether to send an email when the password expires.

```
HRESULT SendEmailUponExpiration([out,retval] VARIANT_BOOL*
pVal);
HRESULT SendEmailUponExpiration([in] VARIANT_BOOL newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Send email on expiration: " & CStr
(oRPS.SendEmailUponExpiration)
```

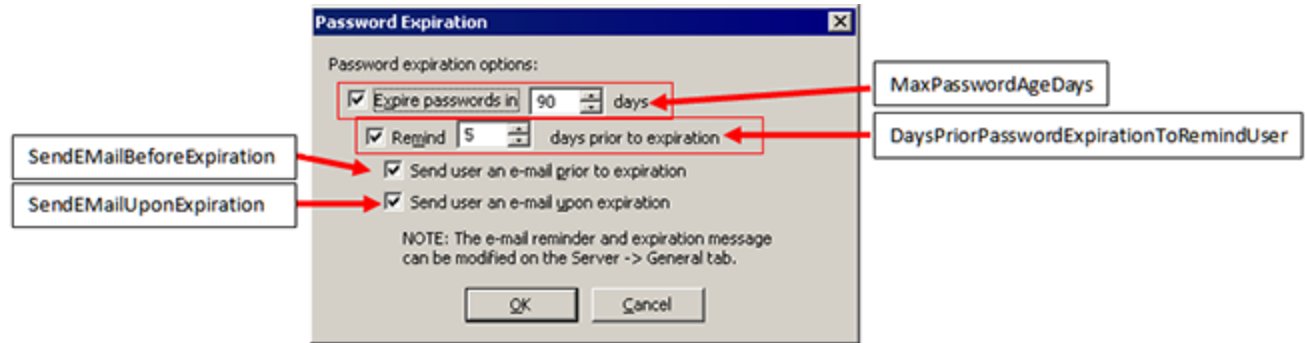
Specifying:

```
oRPS.SendEmailUponExpiration = False
```

EFT v6.3 and later

Dialog Box Equivalents

The ICIRestPasswordSettings interface properties correlate to the following fields and controls in the EFT administration interface:



SFTP-Related Objects

The following interfaces, properties, and methods are used for SFTP management in the COM API.

- [ICISSHKeyInfo Interface](#):
- [ICISite Interface](#):
 - DisableSFTP
 - EnableSFTP
 - GetSFTPAccess
 - SFTPEncoding
 - SFTPEncodingAlgorithms
 - SFTPIdentificationComment
 - SFTPKeyAlgorithms
 - SFTPKeyFile (deprecated in EFT v8)
 - SFTPKeyPassphrase
 - SFTPMacAlgorithms
 - SFTPPort
 - SFTPPublicKeyBlob
 - SFTPSoftwareVersion
- [ICIDMZGateway](#) interface:

- RouteSFTP
- SFTPPort
- [Enums:](#)
 - EventProperty > Client Properties > ClientSFTPAllowed
 - NetworkProtocol > ProtocolSFTP
 - TransferEncodingType (used in SFTPEncoding)

ICISshKeyInfo Interface

Use the ICISshKeyInfo interface to retrieve information about EFT's SSH public keys. The interface provides the set of read-only properties. (Refer also to the [ICIServer:AvailableSSHKeys](#) property.)

```
interface ICISshKeyInfo : IDispatch{
```

Members

- Property **Fingerprint** As String read-only

Retrieves the fingerprint of the SSH key.

```
HRESULT Fingerprint([out, retval] BSTR* pVal);
```

EFT v6.1 and later

- Property **HasPrivate** As Boolean read-only

Retrieves "true" if the key pair includes the private SSH key.

```
HRESULT HasPrivate ([out, retval] BSTR* pVal);
```

Example (VBScript):

```
For Each key In Site.SSHKeys
    If (True = key.HasPrivate) Then
        MsgBox key.Name & " Has private key"
    End If
Next
```

EFT v8 and later

- Property **ID** As Long read-only

Retrieves the ID of the SSH key.

```
HRESULT ID([out, retval] long* pVal);
```

EFT v6.1 and later

- Property **Name** As String read-only

Retrieves the name of the SSH key.

```
HRESULT Name([out, retval] BSTR* pVal);
```

EFT v6.1 and later

Examples (VBScript):

```
MsgBox "Key: " & key.Name & "(" & key.Fingerprint & ")"
Client.SetSSHKeyID key.ID
```

SSL-Related Objects

The following interfaces, properties, and methods are used for SSL management in the COM API.

- [ICISLVersion](#) and [ICISLVersions](#) interfaces to support the SSL library.
- In the [ICIServer](#) interface:
 - [AvailableSSLVersions](#) method (for use in the [ICISLVersions](#) interface).
 - [CipherList](#) property to retrieve or specify the cipher list
 - [CreateSSLCertificate](#) method to create and SSL certificate
 - [CreateSSLCertificateRemotely](#) method create and sign an SSL certificate
 - [PassPhrase](#) property to retrieve or specify an SSL certificate private key passphrase
 - [SetSSLCertificate](#) method specifies the SSL certificate for admin connections
 - [SignSSLCertificateRemotely](#) method to sign a certificate remotely
 - [SSLFIPSEnabled](#) property retrieves or specifies whether FIPS is enabled
 - [SSLVersionMask](#) property to retrieve or specify the SSL version mask
 - [UseSSLForAdministration](#) property enables or disables SSL for the admin socket connection
- In the [ICISite](#) interface:
 - [GetCertFilePath](#) retrieves the Site certificate file path
 - [GetCheckClientCert](#) method determines whether the server requires a certificate from users connecting via implicit SSL.
 - [GetKeyFilePath](#) retrieves the Site's SSL private key file path
 - [GetPassPhrase](#) retrieves the Site's SSL private key passphrase

- [GetSSLAuth](#) returns TRUE if SSL authentication is enabled
- [GetSSLImp](#) returns TRUE if SSL implicit is enabled
- [SetCertFilePath](#) certificate file path and returns TRUE if the Site has to be restarted
- [SetCheckClientCert](#) enables or disables the requirement of certificates on an implicit SSL Site
- [SetKeyFilePath](#) retrieves the Site's SSL private key file path
- [SetPassPhrase](#) retrieves the Site's SSL private key passphrase
- [SetSSLAuth](#) returns TRUE if SSL authentication is enabled
- [SetSSLImp](#) returns TRUE if SSL implicit is enabled
- In the [ICIClientSettings](#) interface:
 - [GetSSLAuthenticationType](#) to retrieve the SSL authentication type for the client
 - [GetSSLKeyID](#) to retrieve the SSL key ID for the client
 - [SetSSLAuthenticationType](#) to specify the SSL authentication type for the client
 - [SetSSLKeyID](#) to specify the SSL key ID for the client
- In the [ICILDAPAuthManagerSettings](#) interface:
 - [UseSSL](#) property retrieves or specifies whether to use SSL for LDAP authentication.
- In the Enums:
 - [CertificateFormat](#), used in the ICIServer [CreateSSLCertificate](#) method, is used to specify the certificate and private key format
 - In the [EventProperty](#) enum, ClientSSLAllowed specifies whether the user can connect using SSL.

Also refer to [Remotely Administering EFT Using the API](#) for information about the SSL DLLs required by the COM API.

User (Client)-Level Objects

This chapter contains Client-related COM objects.

ICIClientSettings Interface

The ICIClientSettings interface displays the user properties defined for a user in EFT.

```
interface ICIClientSettings : IDispatch
```

Use the ICISite interface [GetUserSettings](#) or [GetSettingsLevelSettings](#) methods to manage user account or Settings Template configuration.

See also [ClientDataField](#) enum.

Example:

```
Set oUser = oSite.GetUserSettings(aUsr(j))
Set oSettingsLevel = oSite.GetSettingsLevelSettings("Default
Settings")
```

Once you obtain a handle to the user or Settings Template configuration, you can Get or Set permissions for that user or Settings Template.

See also:

[Difference Between VARIANT_BOOL and SFTPAdvBool](#)

Scroll down to see the [dialog box equivalents](#); Refer to [Workspaces Settings](#) for details of Settings Template and user Workspaces settings.

[Example of using the Custom property](#)

For **SFTPAdvBool**, used in properties below:

- abFalse = 0
- abInherited = -2
- abTrue = 1

Jump to: [C](#) [E](#) [F](#) [G](#) [I](#) [L](#) [M](#) [P](#) [R](#) [S](#) [U](#)

- Property [AccountCreationTime](#) As Date

Retrieves date the user account was created.

```
HRESULT AccountCreationTime([out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Account creation time: " & CStr
(oClientSettings.AccountCreationTime)
```

EFT v6.3 and later

- Sub [AddIPAccessRule](#)(bstrMask As String, bAllow As Boolean, nRulePos As Long, [bstrReason As String = ""])

Adds allowed (true) or denied (false) IP mask for the client.

```
HRESULT AddIPAccessRule(
[in] BSTR bstrMask,
[in] VARIANT_BOOL bAllow,
[in] long nRulePos,
[in, defaultvalue("")]
    BSTR bstrReason);
```

VARIANT_BOOL equals TRUE(-1) when it is allowed and FALSE(0) when it is denied (defaultvalue and Reason added in EFT v8)

Example (C#):

```
client.AddIPAccessRule("1.1.1.2", true,
    0);
```

For an example using this method, refer to the ServerIPBanEntry script in [Script Examples](#).

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

EFT v5.1.1 and later

- Property [AS2InboundSettings](#) As [CIAS2InboundSettings](#)

Retrieves user account's AS2 inbound settings.

```
HRESULT AS2InboundSettings(
[out,retval] IDispatch** pSttg); (v6.3 - 6.3.3)
[out,retval] ICIAS2InboundSettings** pSttg);
```

Example (VBScript):

```
Set oAS2InboundSettings = oClientSettings.AS2InboundSettings
```

EFT v6.3 and later

- Property [AS2OutboundSettings](#) As [CIAS2OutboundSettings](#)

Retrieves user account's AS2 outbound settings.

```
HRESULT AS2OutboundSettings (
[out,retval] IDispatch** pSttg); (v6.3 - 6.3.3)
```

Example (VBScript):

```
Set oAS2OutboundSettings =
oClientSettings.AS2OutboundSettings
```

EFT v6.3 and later

- Function [AS2TestOutboundProfile\(\)](#) As [AS2OutboundProfileStatus](#)

Tests user account's AS2 outbound profile and returns status.

```
HRESULT AS2TestOutboundProfile (
[out,retval] AS2OutboundProfileStatus* pVal);
```

Example (VBScript):

```
if AS2StatusOk = oClientSettings.AS2TestOutboundProfile then
MsgBox "Success."
else
MsgBox "Some going wrong."
end if
```

EFT v6.3 and later

- Property [Comment](#) As String

Retrieve comments for a user account:

```
HRESULT Comment([out, retval] BSTR *pVal);
HRESULT Comment([in] BSTR newVal);
```

All versions

- Property **CompanyName** As String

Retrieve or modify Company Name for a user account:

```
HRESULT CompanyName ([out, retval] BSTR *pVal);
HRESULT CompanyName ([in] BSTR newVal);
```

Example (VBScript):

```
Wscript.echo oUser.CompanyName
oUser.CompanyName = "Company"
```

EFT v8 and later

- Property [Custom1](#) As String; Property [Custom2](#) As String; Property [Custom3](#) As String
(See the [example](#).)

Retrieve the custom fields of a user account.

```
HRESULT Custom1([out, retval] BSTR *pVal);
HRESULT Custom1([in] BSTR newVal);
```

EFT v5.0.1 and later

- Property **Duns** As String

Retrieve or modify a user account' DUNS.

```
HRESULT Duns([out, retval] BSTR *pVal);
HRESULT Duns([in] BSTR newVal);
```

Example (VBScript):

```
Wscript.echo oUser.Duns
oUser.Duns = "123456"
```

EFT v8 and later

- Property [Email](#) As String

Retrieve a user's email address.

```
HRESULT Email(out, retval] BSTR *pVal);
HRESULT Email([in] BSTR newVal);
```

All versions

- Property [Fax](#) As String

Retrieve a user's fax number.

```
HRESULT Fax([out, retval] BSTR *pVal);
HRESULT Fax([in] BSTR newVal);
```

All versions

- Sub [ForcePasswordChange\(\)](#)

Forces the user to change the account password on the next login.

```
HRESULT ForcePasswordChange();
```

EFT v6.0 and later

- Property [FullName](#) As String

Retrieve a user's full name.

```
HRESULT FullName([out, retval] BSTR *pVal);
HRESULT FullName([in] BSTR newVal);
```

All versions

- Property **Mobile** As String

Retrieve or modify mobile number for a user account:

```
HRESULT Mobile ([out, retval] BSTR *pVal);
HRESULT Mobile ([in] BSTR newVal);
```

Example (VBScript):

```
Wscript.echo oUser.Mobile
oUser.Mobile = "123456"
```

EFT v8 and later

- Function [GetAllowCOMB](#)(plInherited) As Boolean
Sub [SetAllowCOMB](#)(val As [SFTPAdvBool](#))

Retrieve/Set whether the COMB command is allowed.

```
HRESULT GetAllowCOMB(
    [out] VARIANT *pInherited,
    [out,retval] VARIANT_BOOL *pVal);
HRESULT SetAllowCOMB(SFTPAdvBool val);
```


VARIANT_BOOL equals TRUE(-1) when it is allowed and FALSE(0) when it is not allowed.

Examples (VBScript):

```
bEnabled = oClientSettings.GetAllowCOMB(bInherited)
MsgBox "Enabled: " & CStr(bEnabled) & " Inherited: " &
CStr(bInherited) oClientSettings.SetAllowCOMB abTrue
```

(For information about using the COMB command with EFT, refer to "Allowing Multipart Transfers (COMB Command)" in the *EFT Installation, Administration, and User Guide*.)

EFT v6.3 and later

- Function [GetAllowFXP](#)(pInherited) As Boolean
Sub [SetAllowFXP](#)(val As [SFTPAdvBool](#))

Retrieve/Specify whether site-to-site transfers are allowed.

Example (VBScript):

```
HRESULT GetAllowFXP([out] VARIANT *pInherited,[out,retval]
VARIANT_BOOL *pVal);
oClientSettings.SetAllowCOMB abFalse HRESULT SetAllowFXP
(SFTPAdvBool val);
```

VARIANT_BOOL equals TRUE(-1) when it is allowed and FALSE(0) when it is not allowed.

EFT v6.3 and later

- Function [GetAllowMODEZ](#)([pInherited]) As Boolean
Sub [SetAllowMODEZ](#)(val As [SFTPAdvBool](#))

Retrieve/Specify whether **Allow Mode Z** for the client is ON or OFF.

```
HRESULT GetAllowMODEZ([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

You can also pass in an optional BOOLEAN variable as a parameter to this function; when the function returns, this optional parameter contains the INHERITED status of this feature.

```
HRESULT SetAllowMODEZ([in] SFTPAdvBool val);
```

All versions

- Function [GetAllowNoop](#)([pInherited]) As Boolean
Sub [SetAllowNoop](#)(val As [SFTPAdvBool](#))

Retrieve/Specify whether a user can send the NOOP command to keep a connection open. Use the same method to prohibit the user from sending the NOOP command.

```
HRESULT GetAllowNoop(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetAllowNoop([in] SFTPAdvBool val);
```

All versions

- Function [GetAllowXCRC](#)([pInherited]) As Boolean
Sub [SetAllowXCRC](#)(val As [SFTPAdvBool](#))

Retrieve/specify whether a user is allowed to send the XCRC command to confirm transfers. Use the same method to prohibit the user from sending the XCRC command.

```
HRESULT GetAllowXCRC([out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False (0) = Disabled

```
HRESULT SetAllowXCRC([in] SFTPAdvBool val);
```

All versions

- Function **GetAllowWorkspaces** As Boolean
Function **SetAllowWorkspaces** As Boolean

Retrieves/specifies whether user account or Settings Template is allowed to create Workspaces.

```
HRESULT GetAllowWorkspaces([out, optional] VARIANT
    *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
HRESULT SetAllowWorkspaces([in] SFTPAdvBool val);
```

EFT v7.3.3

- Function **GetAllowWorkspacesOAI** As Boolean
Function **SetAllowWorkspacesOAI** As Boolean

Retrieves/specifies whether user account or Settings Template is allowed to share files in Workspaces or Outlook

```
HRESULT GetAllowWorkspacesOAI([out, optional] VARIANT
    *pInherited, [out, retval] VARIANT_BOOL *pVal);
HRESULT SetAllowWorkspacesOAI([in] SFTPAdvBool val);
```

EFT v7.3.3

- Function [GetAppletEnabled](#)([pInherited]) As Boolean
Sub [SetAppletEnabled](#)(val As [SFTPAdvBool](#))

Retrieve/specify EFT Web Transfer Client applet access setting for a user or Settings Template.

```
HRESULT GetAppletEnabled([out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetAppletEnabled([in] SFTPAdvBool val);
```

Example, setting the state:

```
slSettings = SetSettingsLevelSettings("Default Settings")
bEnabled = slSettings.SetAppletEnabled( bInherited )
```

Example for v6.1 and later:

```
If Client.AppletEnabled(isInherited) And isInherited
    Then
    MsgBox "WTC is allowed for client because it is allowed for
    Settings Template"
End If
```

For an example using this method, refer to the GetCertInfo script in [Script Examples](#).

EFT v3.5.1 and later

- Function [GetAS2Inbound](#)([pInherited]) As Boolean
Sub [SetAS2Inbound](#)(val As [SFTPAdvBool](#))

Retrieve/specify whether AS2 inbound transactions are enabled for the user or Settings Template.

```
HRESULT GetAS2Inbound([out, optional] VARIANT* pInherited,
[out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetAS2Inbound([in] SFTPAdvBool val);
```

Example (VBScript):

```
bEnabled = oClientSettings.GetAS2Inbound(bInherited)
MsgBox "Enabled: " & CStr(bEnabled) & " Inherited: " &
CStr(bInherited)
oClientSettings.SetAS2Inbound abInherited
```

EFT v6.3 and later

- Function [GetAS2Outbound](#)([pInherited]) As Boolean
Sub [SetAS2Outbound](#)(val As [SFTPAdvBool](#))

Retrieve/specify whether AS2 outbound transactions are enabled for the user or Settings Template.

```
HRESULT GetAS2Outbound([out, optional] VARIANT* pInherited,
[out,
        retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetAS2Outbound([in] SFTPAdvBool val);
```

Example (VBScript):

```
bEnabled = oClientSettings.GetAS2Outbound(bInherited)
MsgBox "Enabled: " & CStr(bEnabled) & " Inherited: " &
CStr(bInherited)
oClientSettings.SetAS2Outbound abTrue
```

EFT v6.3 and later

- Function [GetChangePwd](#)([pInherited]) As Boolean
Sub [SetChangePwd](#)(val As [SFTPAdvBool](#))

Allow or determine if users can change their passwords. Use the same method to prohibit the user from changing passwords.

```
HRESULT GetChangePwd([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetChangePwd([in] SFTPAdvBool val);
```

All versions

- Function [GetClearFTP](#)([pInherited]) As Boolean
Sub [SetClearFTP](#)(val As [SFTPAdvBool](#))

Determine or allow if a user account can make connections using insecure FTP.

```
HRESULT GetClearFTP([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetClearFTP([in] SFTPAdvBool val);
```

All versions

- Function [GetClearHTTP](#)([pInherited]) As Boolean
Sub [SetClearHTTP](#)(val As [SFTPAdvBool](#))

Allow or determine if a user account can make connections using insecure HTTP.

```
HRESULT GetClearHTTP([out, optional] VARIANT *pInherited,
[out,
    retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetClearHTTP([in] SFTPAdvBool val);
```

EFT v4.3.4 and EFT v5.1 and later

- Function [GetComplexPasswordSettings](#)() As [CIComplexPasswordSettings](#)
Sub [SetComplexPasswordSettings](#)(pIComplexPasswordSettings As [CIComplexPasswordSettings](#))

Retrieve/specify complex password settings for a user or Settings Template.

```
HRESULT GetComplexPasswordSettings(
[out, retval] IDispatch** ppIComplexPasswordSettings)
```

```
HRESULT SetComplexPasswordSettings(
    [in] IDispatch* pIComplexPasswordSettings)
```

Examples (VBScript):

```
Set oComplexPwdSettings =
oClientSettings.GetComplexPasswordSettings();
oClientSettings.SetComplexPasswordSettings
oComplexPwdSettings;
```

EFT v6.3 and later

- Function [GetDescription\(\)](#) As String
Sub [SetDescription](#)(bstrVal As String)

Retrieve or provide the description for a user or User Settings Template.

```
HRESULT GetDescription([out, retval] BSTR *prop);
HRESULT SetDescription([in] BSTR bstrVal);
```

All versions

- Function [GetDownloadsPerSession](#)([pInherited]) As Long
Sub [SetDownloadsPerSession](#)(val As Long)

Determine/specify the maximum number of downloads a user is permitted per session.

```
HRESULT GetDownloadsPerSession(
    [out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetDownloadsPerSession([in] long val);
```

All versions

- Function [GetEnableAccount](#)([pInherited]) As Boolean
Sub [SetEnableAccount](#)(val As [SFTPAdvBool](#))

Determine/specify if a user account or Settings Template is enabled.

```
HRESULT GetEnableAccount(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetEnableAccount([in] SFTPAAdvBool val);
```

You need to obtain a handle to the specific user or Settings Template from ICISite's GetUserSettings method or ICISite's GetSettingsLevelSettings method before you can perform an action upon that user or Settings Template.

Below is a sample code snippet using PHP that demonstrates this technique:

```
<?php

// first create server object
$Server = new COM("SFTPCOMInterface.CIServer") or die("Unable to
    instantiate Server");

// connect to server
$Server->Connect("localhost",1000,"admin","admin");

// get handle to list of sites
$Sites = $Server->Sites();

// chose your site. On most one-site systems this will be "0"
$MySite = $Sites->Item(0);

// Pull the settings for the user that you want.
$Settings = $MySite->GetUserSettings("juser");

// enable or disable or inherit.
// Enable is 1. Disable is 0. Inherit is -2
// Notice that you apply this to the settings for the user that you just
pulled
$Settings->SetEnableAccount(1);

// Be sure to apply the settings or else nothing will really change
$Server->ApplyChanges();

// close the connection.
$Server->Close();
?>
```

All versions

- Function [GetEnableDiskQuota](#)([plnherited]) As Boolean
Sub [SetEnableDiskQuota](#)(val As [SFTPAAdvBool](#))

Determine if a user account's home folder space is restricted or limit the amount of space a user can use in a home folder.

```
HRESULT GetEnableDiskQuota(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetEnableDiskQuota([in] SFTPAdvBool val);
```

All versions

- Function [GetEnablePasswordHistory](#)([pInherited]) As Boolean
Sub [SetEnablePasswordHistory](#)(val As [SFTPAdvBool](#))

Determine/specify if password reuse is allowed.

```
HRESULT GetEnablePasswordHistory(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetEnablePasswordHistory(SFTPAdvBool val);
```

Example (VBScript):

```
bEnable = oClientSettings.GetEnablePasswordHistory
(bInherited)
oClientSettings.SetEnablePasswordHistory abTrue
```

EFT v6.3 and later

- Function [GetEnableTimeOut](#)([pInherited]) As Boolean
Sub [SetEnableTimeOut](#)(val As [SFTPAdvBool](#))

Determine if or enable a time out after a user connection is inactive too long.

```
HRESULT GetEnableTimeOut(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetEnableTimeOut([in] SFTPAdvBool val);
```

All versions

- Function [GetExpirationDate](#)(dDate) As Boolean
Sub [SetExpirationDate](#)(dDate, bEnable As Boolean)

Determine/specify whether an expiration date is set for a particular user account. (Set the expiration date with [SetExpirationDate](#).)

```
HRESULT GetExpirationDate(
    [out]VARIANT *dDate,
    [out, retval] VARIANT_BOOL *pVal);
HRESULT SetExpirationDate(
    [in] VARIANT *dDate,
    [in]VARIANT_BOOL bEnable);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Examples:

PHP 5 example (uses a VARIANT rather than a STRING to set the expiration date):

```
$expireddate = "05/15/2005";
$enableexpiration = 1;
$vtdate = new VARIANT($expireddate, VT_DATE);
$UserSettings->SetExpirationDate($expireddate,
    $enableexpiration);
```

VBScript):

```
sUser = "q"
sExpirationDate = "Never"
If Site.GetUserSettings(sUser).GetExpirationDate(
    expirationDate)
    Then
sExpirationDate = CStr(expirationDate)
End If
MsgBox "Account " & sUser & " expiration date: " &
sExpirationDate
Set client_settings = Site.GetUserSettings("testuser")
client_settings.SetExpirationDate
CDate(FormatDateTime("15/05/2005")), True
```

All versions

- Function [GetExpirationDateAsVariant](#)() As Variant
Determine the expiration date for a particular user account, set with [SetExpirationDate](#).

```
HRESULT GetExpirationDateAsVariant([out, retval] VARIANT
*dDate);
```

Example:

```
Dim strUser: strUser = "test"
set oUserSettings = oSite.GetUserSettings(strUser)
dtAccExpDate = oUserSettings.GetExpirationDateAsVariant()
WScript.Echo ("dtAccExpDate = " & dtAccExpDate)
```

EFT v4.3.4 and later

- Function [GetForcePasswordResetOnInitialLogin](#)([pInherited]) As Boolean
Sub [SetForcePasswordResetOnInitialLogin](#)(val As Boolean)

Retrieve or specify whether the option to force the user to change their password on next login is on. If the option allowing this client to change their password is not inherited (i.e., is set to value other than `abInherited`), this option override the Site's **ForcePasswordResetOnInitialLogin** value. If the option is ON, it does not mean the user will necessarily be forced to change the password on next login, because the user's password might be already changed since the admin had switched the option from OFF to ON. To unconditionally force the user to change password on next login, use [ICIClientSettings::ForcePasswordChange](#) method.

```
HRESULT GetForcePasswordResetOnInitialLogin(
[out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
HRESULT SetForcePasswordResetOnInitialLogin(
[in] VARIANT_BOOL val);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):**Specifying:**

```
client.SetChangePwd abTrue 'override "Allow change password"
option first
client.SetForcePasswordResetOnInitialLogin False
```

Retrieving:

```
If client.GetForcePasswordResetOnInitialLogin(isInherited)
Then
If isInherited Then
```

```
MsgBox "The option to reset password is turned on from site"
End If
End If
```

EFT v6.2 and later

- Function **GetForcePasswordResetOnNextLogin()** As Boolean
Sub **SetForcePasswordResetOnNextLogin**(val As SFTPAdvBool)

```
HRESULT GetForcePasswordResetOnNextLogin(
[out, retval] VARIANT_BOOL *pVal);
HRESULT SetForcePasswordResetOnNextLogin(
[in] SFTPAdvBool val);
```

See also [Script Examples](#).

EFT v7.4 and later

- Function [GetFTPS](#)([pInherited]) As Boolean
Sub [SetFTPS](#)(val As [SFTPAdvBool](#))

Retrieve/enable/disable whether access is enabled for a client.

```
HRESULT GetFTPS (
[out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetFTPS([in] SFTPAdvBool val);
```

EFT v6.0 and later

- Function **GetGDPRPrivacyPolicyConsentStatus()** As [GDPRPrivacyPolicyConsentStatus](#)
Sub **SetGDPRPrivacyPolicyConsentStatus**(newVal As [GDPRPrivacyPolicyConsentStatus](#))

Retrieve/specify the Privacy Policy consent status for a client.

```
HRESULT GetGDPRPrivacyPolicyConsentStatus(
[out, retval] GDPRPrivacyPolicyConsentStatus *pVal);
HRESULT SetGDPRPrivacyPolicyConsentStatus(
[in] GDPRPrivacyPolicyConsentStatus newVal);
```

Example (VBScript):

```
Wscript.echo oUser.GetGDPRPrivacyPolicyConsentStatus()
Wscript.echo oUser.SetGDPRPrivacyPolicyConsentStatus
(PPConsent_Granted_explicit)
```

EFT v8 and later

- Function **GetGDPRSubjectStatus()** As [GDPRSubjectStatus](#)
Sub **SetGDPRSubjectStatus**(newVal As [GDPRSubjectStatus](#))

Retrieve/specify GDPR Subject status for a client.

```
HRESULT GetGDPRSubjectStatus(
[out, retval] GDPRSubjectStatus *pVal);
HRESULT SetGDPRSubjectStatus(
[in] GDPRSubjectStatus newVal);
```

Examples (VBScript):

```
Wscript.echo
    oUser.GetGDPRSubjectStatus()
Wscript.echo
    oUser.SetGDPRSubjectStatus(Subject_Yes)
```

EFT v8 and later

- Function **GetGDPRTermsOfServiceConsentStatus()** As [GDPRTermsOfServiceConsentStatus](#)
Sub **SetGDPRTermsOfServiceConsentStatus**(newVal As [GDPRTermsOfServiceConsentStatus](#))

Retrieve/specify the Terms of Service consent status for a client.

```
HRESULT GetGDPRTermsOfServiceConsentStatus(
[out, retval] GDPRTermsOfServiceConsentStatus *pVal);
HRESULT SetGDPRTermsOfServiceConsentStatus(
[in] GDPRTermsOfServiceConsentStatus newVal);
```

Example (VBScript):

```
Wscript.echo oUser. GetGDPRTermsOfServiceConsentStatus()
Wscript.echo oUser. SetGDPRTermsOfServiceConsentStatus
(ToSConsent_Disagreed)
```

EFT v8 and later

- Function [GetHasDownloadsPerSession](#)([pInherited]) As Boolean
Sub [SetHasDownloadsPerSession](#)(val As [SFTPAdvBool](#))

Determine or limit number of files a user is allowed to download per session.

```
HRESULT GetHasDownloadsPerSession(  
    [out, optional] VARIANT *pInherited,  
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetHasDownloadsPerSession([in] SFTPAdvBool val);
```

All versions

- Function [GetHasMaxDownloadSize](#)([pInherited]) As Boolean
Sub [SetHasDownloadsPerSession](#)(val As [SFTPAdvBool](#))

Determine or limit maximum file size a user is allowed to download.

```
HRESULT GetHasMaxDownloadSize([out, optional] VARIANT  
    *pInherited, [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetHasMaxDownloadSize([in] SFTPAdvBool val);
```

All versions

- Function [GetHasMaxIPs](#)([pInherited]) As Boolean
Sub [SetHasMaxIPs](#)(val As [SFTPAdvBool](#))

Determine if or limit user or Settings Template maximum number of allowed concurrent connections from same IP.

```
HRESULT GetHasMaxIPs([out, optional] VARIANT *pInherited,  
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Unlimited; False(0) = Limited

```
HRESULT SetHasMaxIPs([in] SFTPAdvBool val);
```

All versions

- Function [GetHasMaxSpeed](#)([pInherited]) As Boolean
Sub [SetHasMaxSpeed](#)(val As [SFTPAdvBool](#))

Determine if or limit user account maximum allowed transfer speed.

```
HRESULT GetHasMaxSpeed([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Maximum; False(0) = Not the maximum

```
HRESULT SetHasMaxSpeed([in] SFTPAdvBool val);
```

All versions

- Function [GetHasMaxUploadSize](#)([pInherited]) As Boolean
Sub [SetHasMaxUploadSize](#)(val As [SFTPAdvBool](#))

Determine or limit maximum file size a user is allowed to upload.

```
HRESULT GetHasMaxUploadSize([out, optional] VARIANT
    *pInherited,[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Maximum; False(0) = Not the maximum

```
HRESULT SetHasMaxUploadSize([in] SFTPAdvBool val);
```

All versions

- Function [GetHasMaxUsers](#)([pInherited]) As Boolean
Sub [SetHasMaxUsers](#)(val As [SFTPAdvBool](#))

Determine if or limit maximum number of allowed concurrent connections for a user.

```
HRESULT GetHasMaxUsers([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Maximum; False(0) = Not maximum

```
HRESULT SetHasMaxUsers([in] SFTPAdvBool val);
```

All versions

- Function [GetHasPwdRetries](#)([pInherited]) As Boolean
Sub [SetHasPwdRetries](#)(val As [SFTPAdvBool](#))

Determine or specify if a user account will be disabled after trying too many bad passwords in a row. Use the same method to allow an unlimited number of bad passwords.

```
HRESULT GetHasPwdRetries([out, optional] VARIANT
                        *pInherited,[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Maximum; False(0) = Not maximum

```
HRESULT SetHasPwdRetries([in] SFTPAdvBool val);
```

NOTE: GetHasPwdRetries and SetHasPwdRetries are related to an old "disable account after X login attempts" feature, which became obsolete when the PCI-enhanced feature "disable/lockout account after X login attempts in X minute period" was introduced. These methods were left in the .dll for compatibility with old scripts.

All versions

- Function [GetHasUploadsPerSession](#)([pInherited]) As Boolean
Sub [SetHasUploadsPerSession](#)(val As [SFTPAdvBool](#))

Determine if or limit maximum number of files a user is allowed to upload per session.

```
HRESULT GetHasUploadsPerSession([out, optional] VARIANT
                        *pInherited,[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Maximum; False(0) = Not maximum

```
HRESULT SetHasUploadsPerSession([in] SFTPAdvBool
                        val);
```

All versions

- Function [GetHomeDir](#)([pInherited]) As Boolean
Sub [SetHomeDir](#)(val As [SFTPAdvBool](#))

Determine/specify if the user or User Settings Template can have a separate home folder.

```
HRESULT GetHomeDir([out, optional] VARIANT *pInherited,[out,
                        retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetHomeDir([in] SFTPAdvBool val);
```

Examples (VBScript):

```
If client.GetHomeDir Then
MsgBox "Client has separate home folder"
End If
client.SetHomeDir abInherited
'use home folder from Settings Template
```

All versions

- Function [GetHomeDirIsRoot](#)([pInherited]) As Boolean
Sub [SetHomeDirIsRoot](#)(val As [SFTPAdvBool](#))

Determine or specify a user's default root folder.

```
HRESULT GetHomeDirIsRoot([out, optional] VARIANT
*pInherited,[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetHomeDirIsRoot([in] SFTPAdvBool val);
```

All versions

- Function [GetHomeDirString](#)() As String
Sub [SetHomeDirString](#)(bstrVal As String)

Retrieve or specify the path to a user or User Settings Template home folder.

```
HRESULT GetHomeDirString([out, retval] BSTR *prop);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetHomeDirString([in] BSTR bstrVal);
```

All versions

- Function [GetHTTPS](#)([pInherited]) As Boolean
Sub [SetHTTPS](#)(val As [SFTPAdvBool](#))

Retrieve or specify whether access is enabled for a client.

```
HRESULT GetHTTPS([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled


```
HRESULT SetHTTPS([in] SFTPAAdvBool val);
```

EFT v6.0 and later

- Function [GetInactiveAccountsMonitoring](#)([pInherited]) As Boolean
Sub [SetInactiveAccountsMonitoring](#)(val As [SFTPAAdvBool](#))

Retrieve or specify whether inactive accounts are disabled/removed after days of inactivity.

```
HRESULT GetInactiveAccountsMonitoring([out, optional]  
VARIANT* pInherited,[out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetInactiveAccountsMonitoring(SFTPAAdvBool val);
```

Example (C#):

```
//Set inactivity monitoring  
userSettings.RemoveInactiveAccounts = true;  
userSettings.MaxInactivePeriod = 30;  
userSettings.SetInactiveAccountsMonitoring  
(SFTPAAdvBool.abTrue);
```

Example (VBScript):

```
bEnable = oClientSettings.GetInactiveAccountsMonitoring  
(bInherited)  
oClientSettings.SetInactiveAccountsMonitoring abFalse
```

EFT v6.3 and later

- Function [GetIncorrectPasswordAttempts](#)() As Long

Retrieve how many times a user has tried to connect with incorrect log in information.

```
HRESULT GetIncorrectPasswordAttempts([out, retval] long  
*pVal);
```

All versions

- Function [GetIPAccessRules](#)() As Variant

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)[ICIIPAccessRule Interface](#)[ICIManualIPAccessRule Interface](#)[ICIServer - Server Interface](#) : [GetIPAccessRules](#)[ICISite - Single-Site Interface](#) : [GetIPAccessRules](#)

Retrieves an array of client/template IP Access rules.

```
HRESULT GetIPAccessRules([out, retval] VARIANT *aMasks);
```

Example (VBScript):

The following script would retrieve a list of all IP addresses that have been banned automatically for a user or Settings template.

```
rules = oClientSettings.GetIPAccessRules()
For Each key In rules
  If key.type = 0 Then
    WScript.echo "AutoBan"
  End if
  If key.type = 1 Then
    WScript.echo "Manual " + key.address
  End if
Next
```

EFT v6.3 and later

- Function [GetLimitAccessByIP](#)([pInherited]) As Boolean
Sub [SetLimitAccessByIP](#)(val As [SFTPAdvBool](#))

Determine/specify if the IP Access Restriction List is enabled on the Settings Template or user account.

```
HRESULT GetLimitAccessByIP(
  [out, optional] VARIANT *pInherited,
  [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetLimitAccessByIP(
  [in] SFTPAdvBool val);
```

Example (VBScript):

```
If client.GetLimitAccessByIP Then
MsgBox "client's Limit Access by IP is on"
End If
client.SetLimitAccessByIP abInherit
```

Example (VBScript); assigns IP address 1.1.1.1 to a client as an allowed IP address:

```
client.AddIPAccessRule "1.1.1.1", True 'Assign 'Allowed 'IP
```

EFT v6.1 and later

- Function [GetLimitLoginAttempts](#)([pInherited]) As Boolean
Sub [SetLimitLoginAttempts](#)(val As [SFTPAAdvBool](#))

Retrieve/specify whether recognize whether client will be penalized if he repeatedly tries to login with incorrect password.

```
HRESULT GetLimitLoginAttempts(
[out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetLimitLoginAttempts(
[in] SFTPAAdvBool val);
```

EFT v5.1.1 and later

- Function [GetLockoutNotDisable](#)([pInherited]) As Boolean
Sub [SetLockoutNotDisable](#)(val As Boolean)

Determine or specify if account lockout is enabled for a user.

```
HRESULT GetLockoutNotDisable([out, optional] VARIANT
*pInherited, [out, retval] VARIANT_BOOL *pVal);
HRESULT SetLockoutNotDisable([in] VARIANT_BOOL val);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
client.SetLockoutNotDisable False
```

EFT v5.1.1 and later

- Function [GetLoginMsg\(\)](#) As Long
Sub [SetLoginMsg](#)(val As Long)

Determine or specify if the server uses the default log in message, adds a custom string to the default message, replaces the default message with a custom string, or if the server uses no login message.

```
HRESULT GetLoginMsg([out, retval] long *pVal);
HRESULT SetLoginMsg([in] long val);
```

Values

- **0** = Use default
- **1** = Add to default
- **2** = Replace default
- **3** = None

Example (VBScript):

```
If client.GetLoginMsg = 3 Then
MsgBox "No login banner will be shown for the client"
End If
```

What is the difference between a connection banner and a login message?

EFT displays a connection banner immediately after a client connects via FTP, before requesting the login credentials. After the client provides the login credentials and the credentials are authenticated, EFT displays a login banner.

The default login banner is "Login OK, proceed." At the Settings Template level, you can specify the following alternatives: use default ("Login OK, proceed.") banner, append something to default ("Login OK, proceed.") banner, use its own banner, and show no banner at all. At the user account level, you can specify the following alternatives: use default (the one formed at the Settings Template level) banner, append something to default (the one formed at Settings Template level) banner, use its own banner, and show no banner at all.

All versions

- Function [GetLoginMsgString\(\)](#) As String
Sub [SetLoginMsgString](#)(bstrVal As String)

Retrieve or create a login message.

```
HRESULT GetLoginMsgString([out, retval] BSTR *prop);
```

All versions

- Function [GetMaxDownloadSize](#)([pInherited]) As Long
Sub [SetMaxDownloadSize](#)(val As Long)

Determine/specify the maximum file size, in kilobytes, a user is allowed to download.

```
HRESULT GetMaxDownloadSize([out, optional] VARIANT
    *pInherited,[out, retval] long *pVal);
HRESULT SetMaxDownloadSize([in] long val);
```

All versions

- Function [GetMaxIPs](#)([pInherited]) As Long
Sub [SetMaxIPs](#)(val As Long)

Determine or specify the number of concurrent connections allowed from the same IP address.

```
HRESULT GetMaxIPs([out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetMaxIPs([in] long val);
```

All versions

- Function [GetMaxSpace](#)([pInherited]) As Long
Sub [SetMaxSpace](#)(val As Long)

Determine/specify the amount of space a user can use in a home folder.

```
HRESULT GetMaxSpace([out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetMaxSpace([in] long val);
```

All versions

- Function [GetMaxSpeed](#)([pInherited]) As Long
Sub [SetMaxSpeed](#)(val As Long)

Retrieve/specify, in kilobytes per second, the maximum transfer speed allowed for an account or Settings Template.

```
HRESULT GetMaxSpeed([out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetMaxSpeed([in] long val);
```

All versions

- Function [GetMaxUploadSize](#)([pInherited]) As Long
Sub [SetMaxUploadSize](#)(val As Long)

Retrieve/specify, in kilobytes, the maximum file size a user is allowed to upload.

```
HRESULT GetMaxUploadSize([out, optional] VARIANT
    *pInherited,[out, retval] long *pVal);
HRESULT SetMaxUploadSize([in] long val);
```

All versions

- Function [GetMaxUsers](#)([pInherited]) As Long
Sub [SetMaxUsers](#)(val As Long)

Retrieve/retrieve the maximum number of concurrent connections allowed for an account or Settings Template.

```
HRESULT GetMaxUsers([out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetMaxUsers([in] long val);
```

All versions

- Function [GetPwdRetries](#)([pInherited]) As Long
Sub [SetPwdRetries](#)(val As Long)

Retrieve/specify the number of times a user can try to connect with an incorrect password.

```
HRESULT GetPwdRetries([out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetPwdRetries([in] long val);
```

All versions

- Function [GetRequireMfa](#)([pInherited]) As Boolean and Sub [SetRequireMfa](#)(newVal As SFTPAAdvBool)

Used to retrieve or specify whether MFA is required on a Settings Template.

```
HRESULT GetRequireMfa(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
HRESULT SetRequireMfa([in] SFTPAAdvBool newVal);
```

EFT v8.0.4 and later

- Function [GetRequireStrongPasswords](#)([pInherited]) As Boolean and Sub [SetRequireStrongPasswords](#)(val As [SFTPAAdvBool](#))

Retrieve/specify whether strong (complex) passwords are required.

```
HRESULT GetRequireStrongPasswords(
    [out, optional] VARIANT* pInherited,
    [out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetRequireStrongPasswords(
    [in] SFTPAAdvBool val);
```

Examples (VBScript):

```
bEnabled = oClientSettings.GetRequireStrongPasswords
(bInherited)
MsgBox "Enabled: " & CStr(bEnabled) & " Inherited: " & CStr
(bInherited)
oClientSettings.SetRequireStrongPasswords abFalse
```

EFT v6.3 and later

- Function [GetResetPasswordSettings](#)() As CResetPasswordSettings Sub [SetResetPasswordSettings](#)(pResetPasswordSettings As [CResetPasswordSettings](#))

Retrieve/specify password reset settings.

```
HRESULT GetResetPasswordSettings([out, retval] IDispatch**
    ppIResetPasswordSettings); (v6.3 - 6.3.3)
HRESULT SetResetPasswordSettings([in] IDispatch*
    pIResetPasswordSettings); (v6.3 - 6.3.3)
```

Examples (VBScript):

```
Set oResetPwdSettings =
oClientSettings.GetResetPasswordSettings()
oClientSettings.SetResetPasswordSettings oResetPwdSettings
```

Example (C#):

```
//Change reset password settings
CResetPasswordSettings
resetPswdSettings = userSettings.GetResetPasswordSettings();
```

```

resetPswdSettings.MaxPasswordAgeDays = 10;
resetPswdSettings.SendEmailBeforeExpiration = false;
resetPswdSettings.SendEmailUponExpiration = false;
resetPswdSettings.DaysPriorPasswordExpirationToRemindUser =
1;
userSettings.SetResetPasswordSettings(resetPswdSettings);

```

EFT v6.3 and later

- Function [GetSFTP](#)([pInherited]) As Boolean
Sub [SetSFTP](#)(val As [SFTPAdvBool](#))

Retrieve or specify whether access is enabled for a client. True = Access enabled; False = Access disabled.

```

HRESULT GetSFTP([out, optional] VARIANT *pInherited,
[out, retval] VARIANT_BOOL *pVal);

```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```

HRESULT SetSFTP([in] SFTPAdvBool val);

```

Examples:

```

Dim strUser:strUser = "test"
set oUserSettings = oSite.GetUserSettings(strUser)
bIsFTPEnabled = oUserSettings.GetSFTP()
WScript.Echo ("bIsFTPEnabled=" & bIsFTPEnabled)

```

```

Dim strUser:strUser = "test"
set oUserSettings = oSite.GetUserSettings(strUser)
oUserSettings.SetSFTP abTrue

```

All versions

- Function [GetSftpAuthenticationType](#)([pInherited]) As String
Sub [SetSftpAuthenticationType](#)(val As String)

Retrieve/specify the SFTP authentication type for the client.

```

HRESULT GetSftpAuthenticationType(
[out, optional] VARIANT *pInherited,

```



```
[out, retval] BSTR *pVal);
HRESULT SetSftpAuthenticationType([in] BSTR val);
```

Values

- inherited
- password
- key
- both
- either (v6.3 and later)

Examples:

```
Dim strUser:strUser = "test"
Set oUserSettings = oSite.GetUserSettings(strUser)
Dim strSFTPEnabled: strSFTPEnabled=false
oUserSettings.GetSFTPAuthenticationType(strSFTPEnabled)
Dim strType: strType =
oUserSettings.GetSFTPAuthenticationType()
WScript.Echo "SFTP Enabled: " & strSFTPEnabled
WScript.Echo "SFTP Password type: " & strType
```

```
Dim strUser:strUser = "test"
set oUserSettings = oSite.GetUserSettings(strUser)
oUserSettings.SetSFTPAuthenticationType ("password")
```

All versions

- Function [GetSSHKeyID](#)([pInherited]) As Long
Sub [SetSSHKeyID](#)(val As Long)

Retrieve/specify the SFTP (SSH) certificate ID for the client.

```
HRESULT GetSSHKeyID([out, optional] VARIANT *pInherited,
[out, retval] long *pVal);
HRESULT SetSSHKeyID([in] long val);
```

Examples:

```
For Each keyId In oClientSettings.GetSSHKeyIDs(bInherited)
MsgBox "Key id: " & CStr(keyId)
Next
```

```
Dim strUser:strUser = "test"
set oUserSettings = oSite.GetUserSettings(strUser)
oUserSettings.SetSSHKeyID(1)
```

All versions

- Function **GetSSHKeyIDs**([pInherited]) As Variant
Sub SetSSHKeyID(val As Long)

Retrieves/specifies the list of SSH Key IDs assigned to specified client.

```
HRESULT GetSSHKeyIDs([out, optional] VARIANT *pInherited,
[out, retval] VARIANT* pVal);
HRESULT SetSSHKeyIDs([in] VARIANT val);
```

Example (VBS):

Assign keys with ID = 1, 2, 3, 4, 5, 6, 7, and 8 to Test_user, then get all keys:

```
Set Client = Site.GetUserSettings("Test_user")
Client.SetSSHKeyIDs Array(1, 2, 3, 4, 5, 7, 8)
Dim allKeys
keys = Client.GetSSHKeyIDs
For Each keyID in Keys
    allKeys = allKeys & vbNewLine & CStr(keyID)
Next
MsgBox allKeys
```

Expected result is message box with 1, 2, 3, 4, 5, 6, 7, 8

Example (C#):

Assign keys in a list to a client:

```
userSettings.SetSSHKeyIDs(keyIds.ToArray());
```

See also [ExportSSHKey](#) (in the Server interface), SetSSHKeyIDs, [GetSSHKeyID](#) (above).

EFT v6.3 and later

- Function [GetSSLAuthenticationType](#)([pInherited]) As String
Sub [SetSSLAuthenticationType](#)(val As String)

Retrieve/specify authentication type for the client.

```
HRESULT GetSSLAuthenticationType([out, optional]
                                VARIANT *pInherited,[out, retval] BSTR *pVal);
```

```
HRESULT SetSSLAuthenticationType([in] BSTR val);
```

Examples

```
Dim strUser:strUser = "test"
Set oUserSettings = oSite.GetUserSettings(strUser)
Dim strSSLEnabled: strSSLEnabled=false
oUserSettings.GetSSLAuthenticationType(strSSLEnabled)
Dim strType: strType =
oUserSettings.GetSSLAuthenticationType()
WScript.Echo "SSL Enabled: " & strSSLEnabled
WScript.Echo "SSL Password type: " & strType
```

```
Dim strUser:strUser = "test"
Set oUserSettings = oSite.SetUserSettings(strUser)
Dim strSSLEnabled: strSSLEnabled=false
oUserSettings.SetSSLAuthenticationType(strSSLEnabled)
Dim strType: strType =
oUserSettings.SetSSLAuthenticationType()
WScript.Echo "SSL Enabled: " & strSSLEnabled
WScript.Echo "SSL Password type: " & strType
```

EFT v5.1 and later

- Function [GetSSLKeyID](#)([pInherited]) As Long
Sub SetSSLKeyID(val As Long)

Retrieve/specify the SSL key ID for the client.

```
HRESULT GetSSLKeyID(
[out, optional] VARIANT *pInherited,
[out, retval] long *pVal);
HRESULT SetSSLKeyID([in] long val);
```

Example:

```
Dim strUser:strUser = "test"
Set oUserSettings = oSite.GetUserSettings(strUser)
Dim strSSLEnabled: strSSLEnabled=false
oUserSettings.GetSSLKeyID(strSSLEnabled)
```

```
Dim strType: strType = oUserSettings.GetSSLKeyID()
WScript.Echo "SSL Enabled: " & strSSEnabled
WScript.Echo "SSL Password type: " & strType
```

EFT v5.1 and later

- Function [GetTimeOut](#)([pInherited]) As Long
Sub [SetTimeOut](#)(val As Long)

Retrieve/specify number of seconds EFT will allow a user's connection to be inactive before closing the connection.

```
HRESULT GetTimeOut(
    [out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
HRESULT SetTimeOut([in] long val);
```

All versions

- Function [GetTwoFactorAuthentication](#)([pInherited]) As Boolean
Sub [SetTwoFactorAuthentication](#)(val As [SFTPAdvBool](#))

Retrieve/specify whether two-factor authentication is enabled for the user or Settings Template.

```
HRESULT GetTwoFactorAuthentication(
    [out, optional] VARIANT* pInherited,
    [out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

```
HRESULT SetTwoFactorAuthentication([in] SFTPAdvBool val);
```

Examples (VBScript):

```
oClientSettings.GetTwoFactorAuthentication(bInherited)
oClientSettings.SetTwoFactorAuthentication abFalse
```

EFT v6.3 and later

- Function [GetUploadsPerSession](#)([pInherited]) As Long
Sub [SetUploadsPerSession](#)(val As Long)

Determine/specify the maximum number of uploads a user is permitted per session.

```
HRESULT GetUploadsPerSession(
    [out, optional] VARIANT *pInherited,
    [out, retval] long *pVal);
SetUploadsPerSession([in] long val);
```

All versions

- Function [GetUsedSpace\(\)](#) As Long

Determine the amount of disk space, in kilobytes, a user has used in a home folder.

```
HRESULT GetUsedSpace([out, retval] long *pVal);
```

Example (VBScript):

```
MsgBox "Client data occupy " & client.GetUsedSpace & " KB"
```

All versions

- Property **ID** As Long

Retrieve a user or Settings Template ID.

```
HRESULT ID([out,retval] LONG* pID);
```

Example:

```
MsgBox "Client id: " & CStr(oClientSettings.ID)
```

EFT v6.3 and later

- Property [InvalidAttemptsPeriod](#)(pDate) As Long

Specify or retrieve user invalid login attempts critical period.

```
HRESULT InvalidAttemptsPeriod([out, retval] long *pVal);
HRESULT InvalidAttemptsPeriod([in] long newVal);
```

EFT v6.1 and later

- Property [IPAccessAllowedDefault](#) As Boolean

Specify or retrieve whether user or Settings Template IP access settings allow or deny access by default (if no rule match).

```
HRESULT IPAccessAllowedDefault(
    [out, retval] VARIANT_BOOL *pVal);
```

```
HRESULT IPAccessAllowedDefault (
[in] VARIANT_BOOL newVal);
```

VARIANT_BOOL equals True(-1) = Allow; False(0) = Deny

EFT v5.1.1 and later

- Property **IsConnected** As Boolean read-only

Retrieve whether the client is connected. (Client only, not for user group.)

```
HRESULT IsConnected([out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Connected; False(0) = Not connected

Example (VBScript):

```
if True = oClientSettings.IsConnected then
MsgBox "Connected."
end if
```

EFT v6.3 and later

- Function [IsInactivityPeriodLimited](#)(pDate) As Boolean

Retrieve whether client account has limited maximum inactivity period and what the period end date is.

```
HRESULT IsInactivityPeriodLimited (
[out] VARIANT* pDate,
[out,retval] VARIANT_BOOL* pLimited);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
If Client.IsInactivityPeriodLimited(date) Then
MsgBox "Client will be sanctioned (removed or
disabled)" &_
"if it continues be inactive before " & CStr(date)
End If
```

EFT v6.3 and later

- Property [IsLocked](#) As Boolean

Retrieve whether account is locked.

```
HRESULT IsLocked(
    [out, retval] VARIANT_BOOL* pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
if False = oClientSettings.IsLocked then
    MsgBox "Not locked."
end if
```

EFT v6.3 and later

- Function [IsPasswordAgeLimited](#)(pDate) As Boolean

Retrieve whether the user account password will expire and, if so, the expiration date.

```
HRESULT IsPasswordAgeLimited(
    [out] VARIANT* pDate,
    [out,retval] VARIANT_BOOL* pLimited);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
If Client.IsPasswordAgeLimited(date) Then
    MsgBox "Client Password will expire " &
        CStr(date)
End If
```

EFT v6.1 and later

- Property [LastConnectionTime](#) As Date

Retrieve date and time user last connected to the server.

```
HRESULT LastConnectionTime(
    [out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Last connection time: " &
    CStr(oClientSettings.LastConnectionTime)
```

EFT v6.3 and later

- Property [LastModificationTime](#) As Date

Retrieve date and time user account was last modified.

```
HRESULT LastModificationTime(
    [out, retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Last modification time: " &
    CStr(oClientSettings.LastModificationTime)
```

EFT v6.3 and later

- Property [LastModifiedBy](#) As String

Retrieve administrator account name that last modified this user account.

```
HRESULT LastModifiedBy(
    [out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Last modified by: " & CStr
    (oClientSettings.LastModifiedBy)
```

EFT v6.3 and later

- Property [LockoutPeriod](#) As Long

Retrieve the length of time a user's account is locked out after invalid login attempts.

```
HRESULT LockoutPeriod([out, retval] long *pVal);
HRESULT LockoutPeriod([in] long newVal);
```

EFT v5.1.1 and later

- Property [MaxInactivePeriod](#) As Long

Retrieve the number of days of inactivity before account is disabled or removed.

```
HRESULT MaxInactivePeriod([out,retval] LONG* pVal);
HRESULT MaxInactivePeriod([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Inactive period is: " &
    CStr(oClientSettings.MaxInactivePeriod)
```


Specifying:

```
oClientSettings.MaxInactivePeriod = 90
```

EFT v6.3 and later

- Property [MaxInvalidLoginAttempts](#) As Long

Retrieve the maximum number of invalid login attempts a client can make to a Site.

```
HRESULT MaxInvalidLoginAttempts([out, retval] long *pVal);
HRESULT MaxInvalidLoginAttempts([in] long newVal);
```

EFT v5.1.1 and later

- Sub [MoveIPAccessRule](#)(nRulePosFrom As Long nRulePosTo As Long)

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Change allowed IP access rule position.

```
HRESULT MoveIPAccessRule(
[in] long nRulePosFrom,
[in] long nRulePosTo);
```

VARIANT_BOOL True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
oClientSettings.MoveIPAccessRule(1,2)
```

EFT v6.3 and later

- Property [Pager](#) As String

View a user's pager number.

```
HRESULT Pager([out, retval] BSTR *pVal);
HRESULT Pager([in] BSTR newVal);
```

EFT v5.0 and later

- Property PartnerID As String

Retrieve or modify PartnerID for a user account:

```
HRESULT PartnerID ([out, retval] BSTR *pVal);
HRESULT PartnerID ([in] BSTR newVal);
```

Example (VBScript):

```
Wscript.echo
    oUser.PartnerId
oUser. PartnerId
    = "PARTNERID"
```

EFT v8 and later

- Property PasswordHistorySize As Long

Retrieves number of previous passwords prohibited from reuse.

```
HRESULT PasswordHistorySize([out,retval] LONG* pVal);
HRESULT PasswordHistorySize(LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Password history size: " &
CStr(oClientSettings.PasswordHistorySize)
```

Specifying:

```
oClientSettings.PasswordHistorySize = 8
```

EFT v6.3 and later

- Property [Phone](#) As String

Retrieve a user's telephone number.

```
HRESULT Phone([out, retval] BSTR *pVal);
HRESULT Phone([in] BSTR newVal);
```

EFT v5.0 and later

- Property [RemoveInactiveAccounts](#) As Boolean

Specifies whether inactive accounts are removed.

```
HRESULT RemoveInactiveAccounts([out, retval] VARIANT_BOOL*
pVal);
HRESULT RemoveInactiveAccounts(VARIANT_BOOL newVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

Retrieving:

```
if False = oClientSettings.RemoveInactiveAccounts
    then
MsgBox "Disable"
else
MsgBox "Remove"
end if
```

Specifying:

```
oClientSettings.RemoveInactiveAccounts = True
```

EFT v6.3 and later

- Sub [RemoveIPAccessRule](#)(nRulePos As Long)

See also the following interfaces:

[ICIAutobanIPAccessRule Interface](#)

[ICIBannedIP Interface](#)

[ICIIPAccessRule Interface](#)

[ICIManualIPAccessRule Interface](#)

Remove the IP access rule.

```
HRESULT RemoveIPAccessRule([in] long nRulePos);
```

Example (VBScript):

```
oClientSettings.RemoveIPAccessRule(0)
```

EFT v5.1.1 and later

- Sub [ResetAllFieldAttributeDefaults](#)()

Reset user's client data fields attributes to default. (i.e., apply attributes from Site's Security tab > User Account Details Template dialog box.)

```
HRESULT ResetAllFieldAttributeDefaults();
```

Example (VBScript):

```
oUser. ResetAllFieldAttributeDefaults()
```

EFT v8 and later

- Sub ResetPassword()

Retrieve whether a user can reset the account password.

```
HRESULT ResetPassword([out,retval] VARIANT_BOOL* pbSuccess);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example (VBScript):

```
if False = oClientSettings.ResetPassword() then
MsgBox "Reset user password failed."
end if
```

EFT v6.3 and later

Dialog Box Equivalents

The ICIClientSettings interface methods and properties correlate to the following fields and controls in the EFT administration interface:

User Account Details forUser

User account details include the user's full name, phone, e-mail, fax, and pager, and can include custom fields and comments. You can view/set the user account details in the administration interface on the user's General tab > Account Details.

The dialog box below is labeled with some of the parameters that are available in the ICIClientSettings interface

User Account Details for NewUser

X

Full Name:

FullName

Company Name:

CompanyName

Description:

Email:

Email

DUNS:

Duns

Phone:

Phone

Mobile:

Mobile

Pager:

Pager

Fax:

Fax

Partner ID:

PartnerID

Custom 1:

Custom1

Custom 2:

Custom2

Custom 3:

Custom3

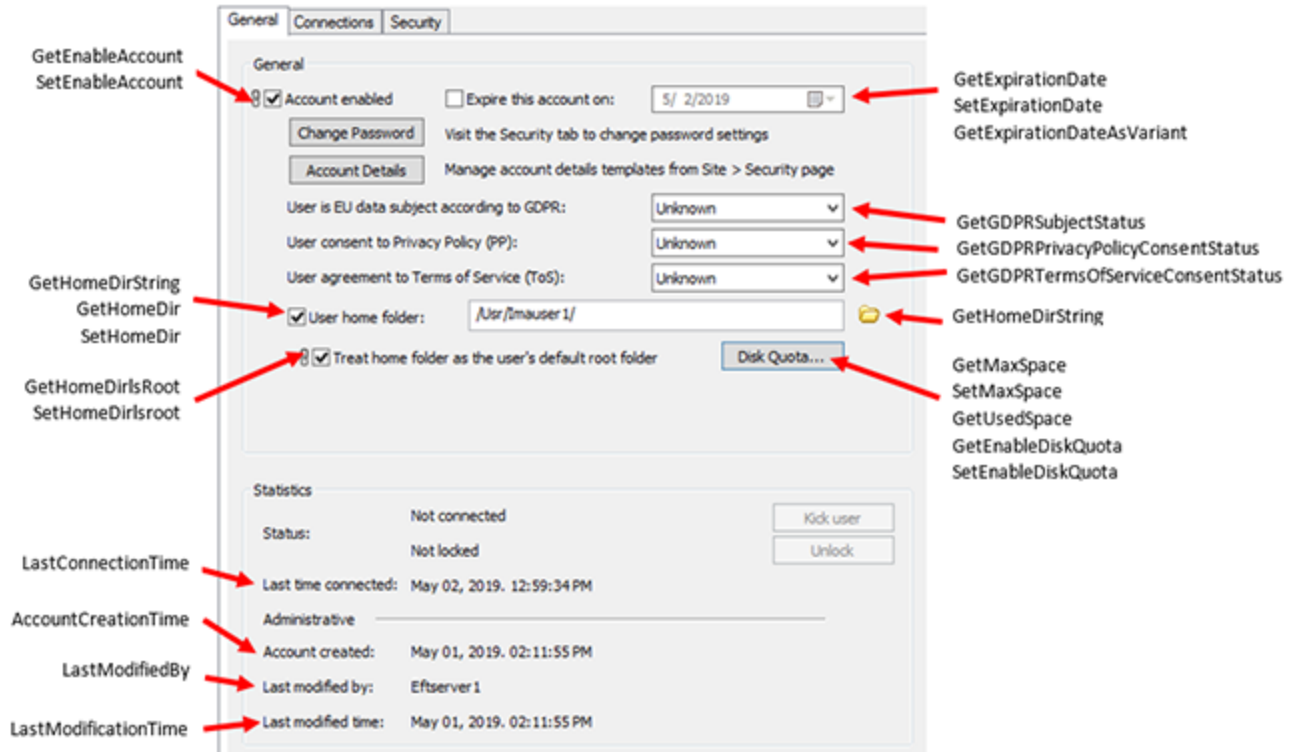
Comments:

Comment

OK

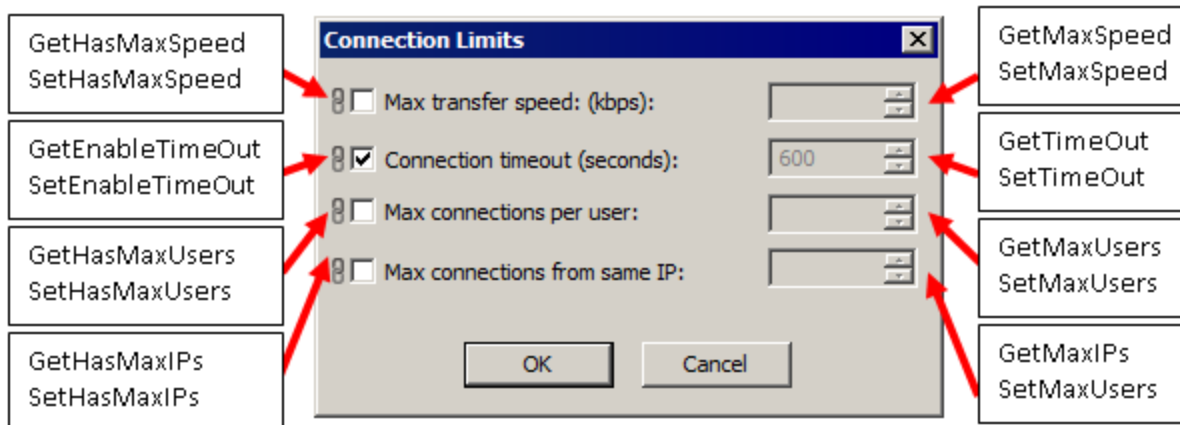
Cancel

User Account General Settings

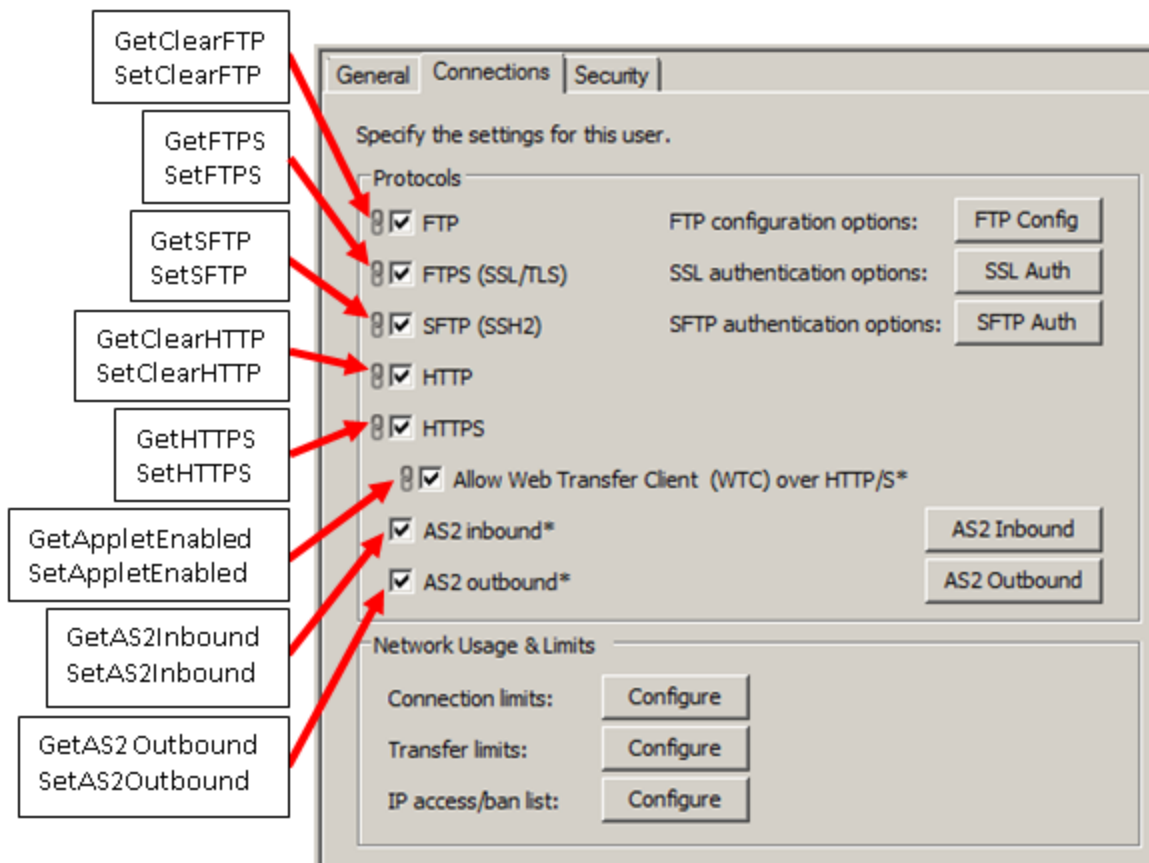


See also ICISite interface: [KickUser](#) and [UnlockUser](#)

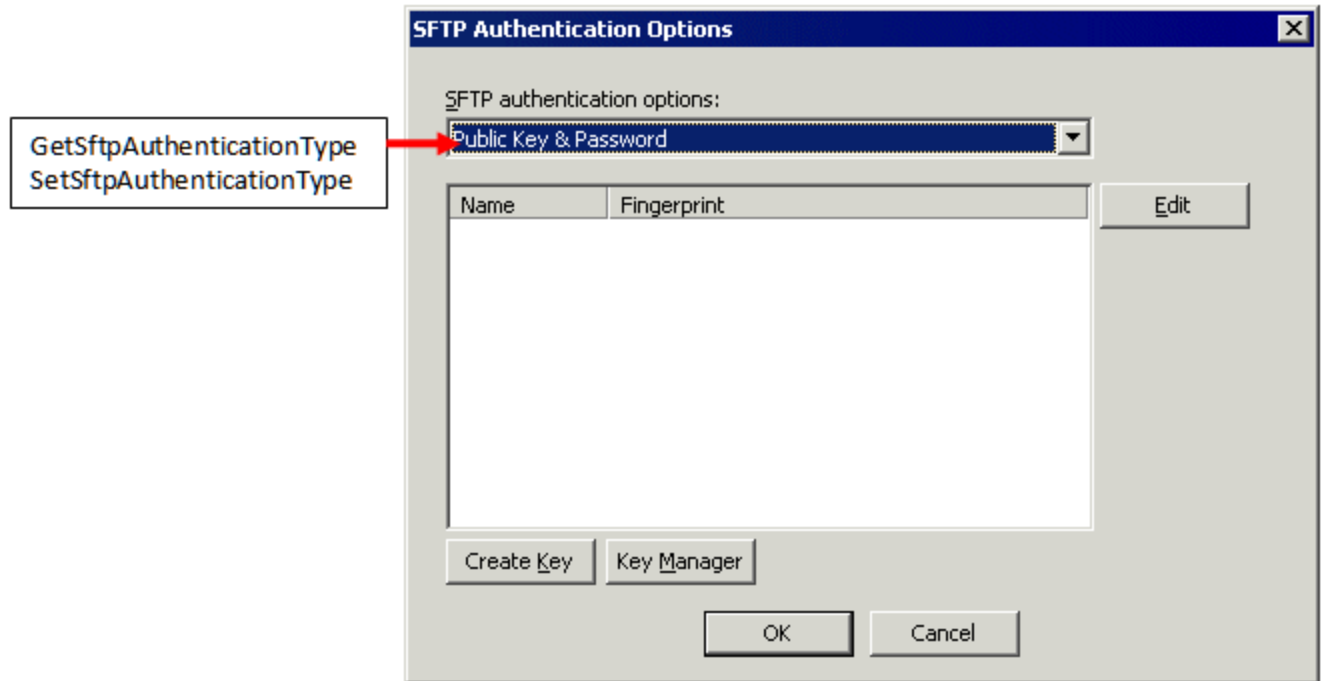
User Connection Limits



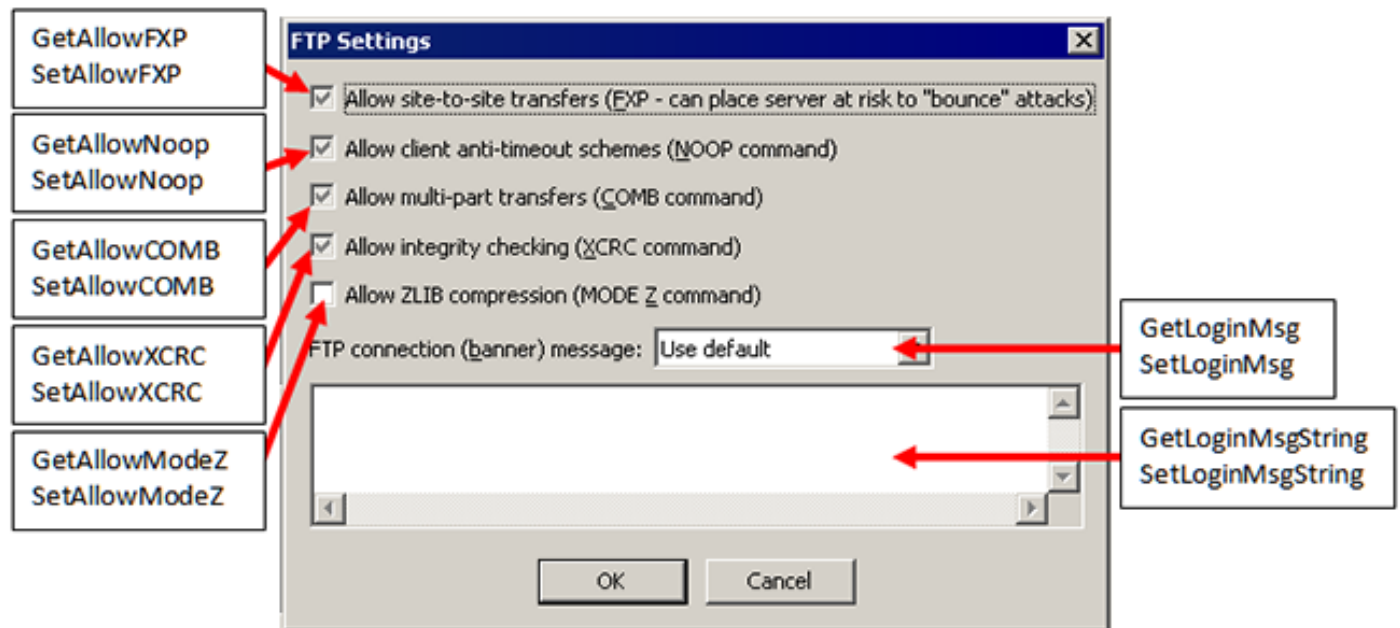
User Connection Protocols



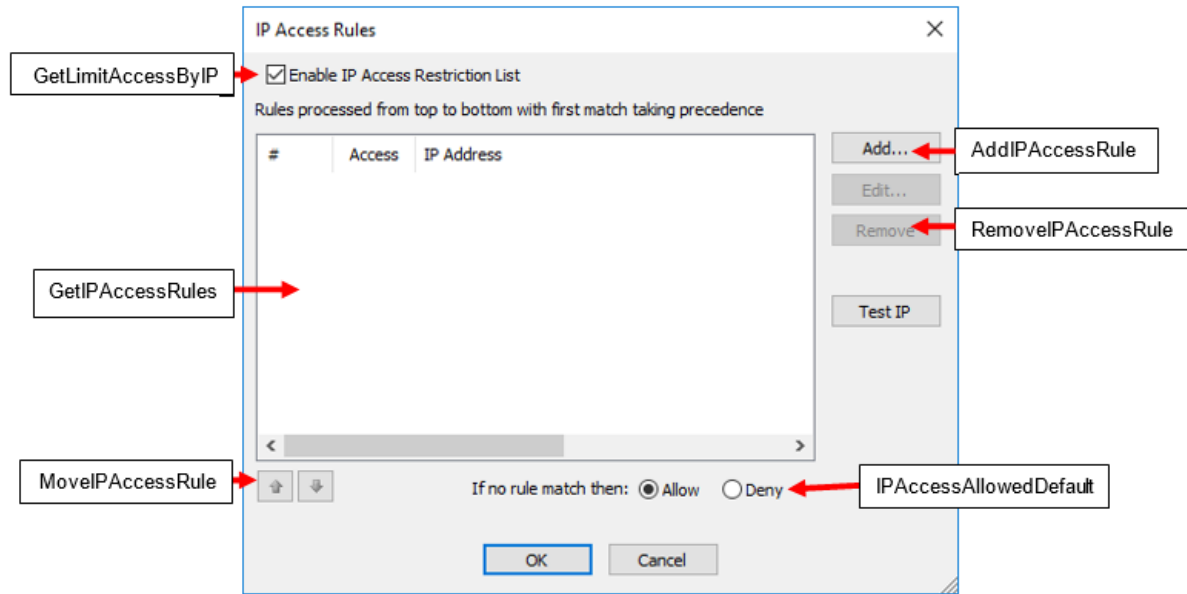
User SFTP Authentication Options



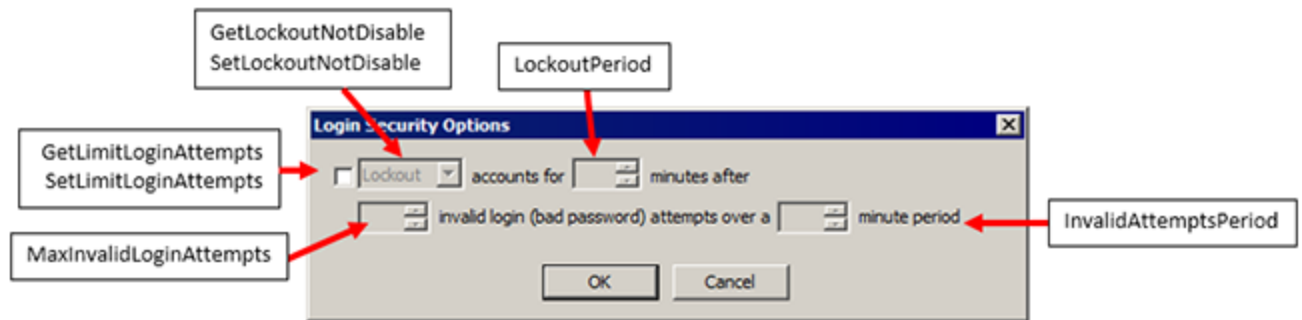
User FTP Security Settings



User IP Access Ban List

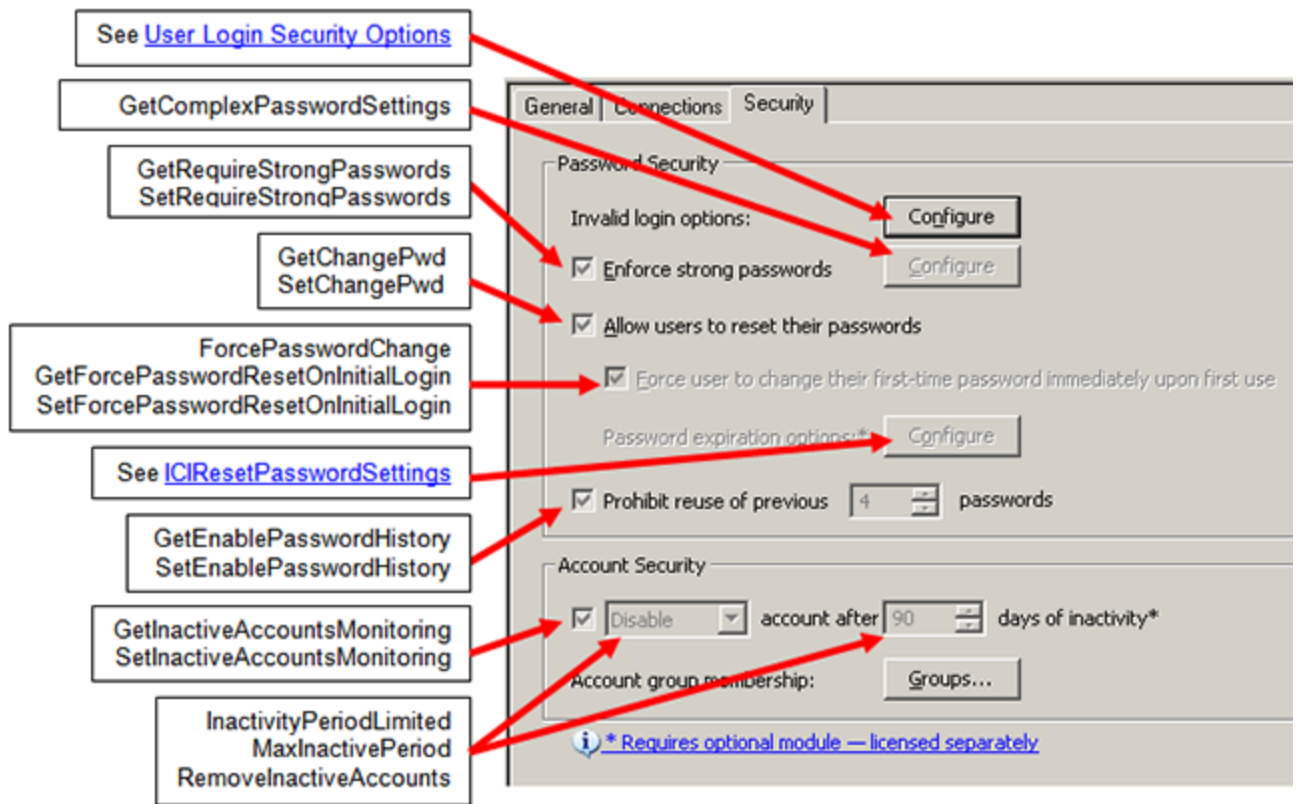


User Login Security Options

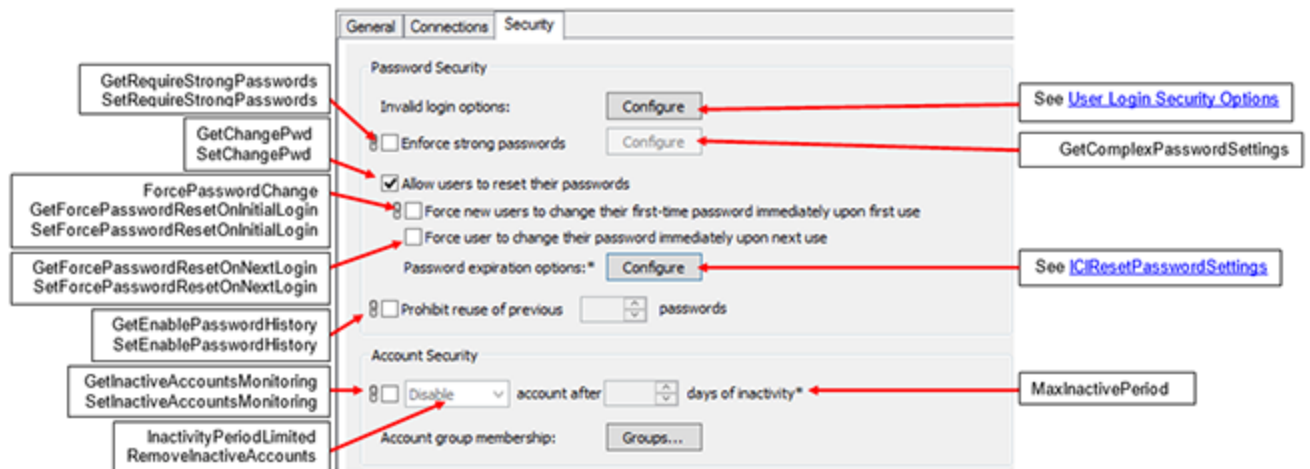


User Security and Password Settings

In 7.3.6 and earlier:

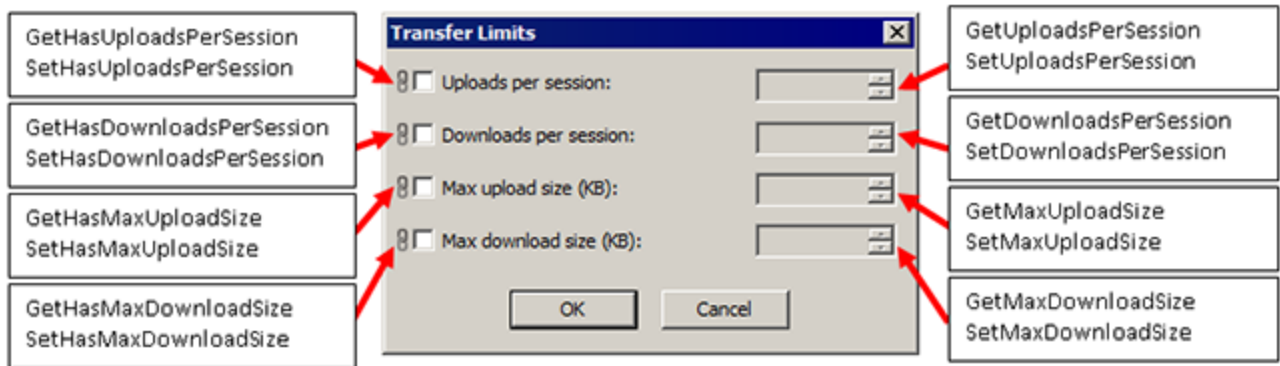


In 7.3.7 and later:



User Transfer Limits Settings

User connection limits in the API include getting/setting the number of uploads and download per session and maximum upload and download sizes. The dialog box below, available from the user's Connections tab, is labeled with some of the transfer limits parameters that are available in the ICIClientSettings interface.



Example of using the Custom property

```
// eftTestCustomFields.js
// CREATED: 24 May 2007
// Script to confirm use of EFT5 Custom Fields properties
// of a CClientSettings in Javascript through API.
//
var args = WScript.Arguments;
if ( args.length < 5 )
{
    WScript.Echo("Usage: eftTestCustomFields <eft
    host> <eft port> <userid> <password> <user id>\n");
    WScript.Echo("Example:  cscript eftTestCustomFields.js
    192.168.20.101 1100 admin admin foo\n");
    WScript.Quit(1);
}
var sHost = args(0);
var iPort = args(1);
var sUser = args(2);
var sPass = args(3);
var sAccountName = args(4);
var server = new ActiveXObject("SFTPCOMInterface.CIServer");
WScript.Echo("Connecting...");
server.Connect(sHost, iPort, sUser, sPass);
WScript.Echo("Connected!");
var site = server.Sites().SiteByID(1);
// for simplicity, assume first site
var oSettings = site.GetUserSettings(sAccountName);
WScript.Echo("Obtained user settings for user account
    '" + oSettings.FullName + "' with home directory: " +
    oSettings.GetHomeDirString()
    );
WScript.Echo("Custom1='" + oSettings.Custom1
    + "'");
```

```

WScript.Echo("Custom2='" + oSettings.Custom2
+ "'");
WScript.Echo("Custom3='" + oSettings.Custom3
+ "'");

// Now set the custom fields to new values
WScript.Echo("\nSetting new values...");
oSettings.Custom1 = "Random=" + Math.floor((65535*Math.random
()))
+ " [" + (new Date()).toLocaleString() + "]";
oSettings.Custom2 = "Random=" + Math.floor((65535*Math.random
()))
+ " [" + (new Date()).toLocaleString() + "]";
oSettings.Custom3 = "Random=" + Math.floor((65535*Math.random
()))
+ " [" + (new Date()).toLocaleString() + "]";
server.ApplyChanges();

// Now REQUERY to CONFIRM the custom fields are the new
values
WScript.Echo("Requerying values...");
var oSettings = site.GetUserSettings(sAccountName);
WScript.Echo("\nObtained NEW user settings for user
account '" + oSettings.FullName + "' with home directory: "
+ oSettings.GetHomeDirString() );
WScript.Echo("Custom1='" + oSettings.Custom1
+ "'");
WScript.Echo("Custom2='" + oSettings.Custom2
+ "'");
WScript.Echo("Custom3='" + oSettings.Custom3
+ "'");
server.Close();
WScript.Echo("\nDone");
WScript.Quit(0);

```

ICINewUserData Interface

The ICINewUserData used with the ICISite [CreateUserEx2](#) property to create a new user account and include all of the user details.

Properties

- Property **CreateHomeFolder**

Specifies whether the account's home folder is the default root folder.

```
HRESULT CreateHomeFolder([out, retval] VARIANT_BOOL *pVal);
HRESULT CreateHomeFolder([in] VARIANT_BOOL newVal);
```

EFT v6.3.8 and later

- Property **Description**

Specifies the account description.

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

EFT v6.3.8 and later

- Property **Email**

Specifies the e-mail address of the account.

```
HRESULT Email([out, retval] BSTR *pVal);
HRESULT Email([in] BSTR newVal);
```

EFT v6.3.8 and later

- Property **FullName**

Specifies the full name of the user.

```
HRESULT FullName([out, retval] BSTR *pVal);
HRESULT FullName([in] BSTR newVal);
```

EFT v6.3.8 and later

- Property **FullPermissionsForHomeFolder**

Specifies whether the user has FULL permissions in the home folder.

```
HRESULT FullPermissionsForHomeFolder([out, retval] VARIANT_
BOOL *pVal);
HRESULT FullPermissionsForHomeFolder([in] VARIANT_BOOL
newVal);
```

EFT v6.3.8 and later

- Property **Login**

Specifies the account username.

```
HRESULT Login([out, retval] BSTR *pVal);
HRESULT Login([in] BSTR newVal);
```

EFT v6.3.8 and later

- Property **Password**

Specifies the account password.

```
HRESULT Password([out, retval] BSTR *pVal);
HRESULT Password([in] BSTR newVal);
```

EFT v6.3.8 and later

- Property **PasswordType**

Specifies the password type:

- Standard - A plain text password is required.
- Anonymous - Any password, including nothing, allows an anonymous connection.
- (removed in v7.3.3.21 and later) Anonymous (Force e-mail) - Any well-formed e-mail address is the password
- (removed in v7.3.3.21 and later) OTP S/KEY MD4 - Used for logging in to an OTP-enabled server.
- (removed in v7.3.3.21 and later) OTP S/KEY MD5 - Used for logging in to an OTP-enabled server.

```
HRESULT PasswordType([out, retval] long *pVal);
HRESULT PasswordType([in] long newVal);
```

EFT v6.3.8 and later

- Property **SettingsLevel**

Specifies the Settings Level to which this account belongs.

```
HRESULT SettingsLevel([out, retval] BSTR *pVal);
HRESULT SettingsLevel([in] BSTR newVal);
```

EFT v6.3.8 and later

- Property **TwoFactorAuthentication**

Specifies whether RADIUS/RSA is enabled, disabled, or inherited.

```
HRESULT TwoFactorAuthentication([out, retval] SFTPAAdvBool
*pVal);
HRESULT TwoFactorAuthentication([in] SFTPAAdvBool newVal);
```

EFT v6.3.8 and later

ICComplexPasswordSettings Interface

The IComplexPasswordSettings interface is used to specify complex password settings, such as minimum password length.

```
interface IComplexPasswordSettings : IDispatch{
```

See also:

- [Difference Between VARIANT_BOOL and SFTPAAdvBool](#)
- Scroll down to see the dialog box equivalents.
- [ICIServer::GetComplexPasswordSettings](#), [ICIServer::SetComplexPasswordSettings](#); [ICISite::GetComplexPasswordSettings](#), [ICISite::SetComplexPasswordSettings](#); [ICIClientSettings::GetComplexPasswordSettings](#), and [ICIClientSettings::SetComplexPasswordSettings](#) methods

Members

- Property **MaxRepeatingCharacters** As Long

Retrieves or specifies the maximum number of repeating characters allowed.

```
HRESULT MaxRepeatingCharacters([out, retval] LONG* pVal);
HRESULT MaxRepeatingCharacters([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Max repeating characters: " & CStr
(oCPS.MaxRepeatingCharacters)
```

Specifying:

```
oCPS.MaxRepeatingCharacters = 3
```

EFT v6.3 and later

- Property **MaxUsernameCharactersInPassword** As Long

Retrieves or specifies the maximum number of username characters allowed in passwords.

```
HRESULT MaxUsernameCharactersInPassword([out, retval] LONG*
pVal);
HRESULT MaxUsernameCharactersInPassword([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Max username characters: " & CStr
(oCPS.MaxUsernameCharactersInPassword)
```

Specifying:

```
oCPS.MaxUsernameCharactersInPassword = 4
```

EFT v6.3 and later

- Property **MinCharCategoriesInPassword** As Long

Retrieves or specifies the minimum number of character categories in passwords.

```
HRESULT MinCharCategoriesInPassword([out, retval] LONG*
pVal);
HRESULT MinCharCategoriesInPassword([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Minimal character categories in passwod is: " &
CStr(oCPS.MinCharCategoriesInPassword)
```

Specifying:

```
oCPS.MinCharCategoriesInPassword = 3
```

EFT v6.3 and later

- Property **MinPasswordLength** As Long

Retrieves or specifies the minimum password length.

```
HRESULT MinPasswordLength([out, retval] LONG* pVal);
HRESULT MinPasswordLength([in] LONG newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Minimal password length is: " & CStr
(oCPS.MinPasswordLength)
```

Specifying:

```
oCPS.MinPasswordLength = 8
```

EFT v6.3 and later

- Property **PasswordDictionaryPath** As String

Retrieves or specifies the path to the dictionary file.

```
HRESULT PasswordDictionaryPath([out, retval] BSTR* pVal);
HRESULT PasswordDictionaryPath([in] BSTR newVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "Dictionary path is: " & oCPS.PasswordDictionaryPath
```

Specifying:

```
oCPS.PasswordDictionaryPath = "C:\dictionary.txt"
```

EFT v6.3 and later

- Property **ProhibitDictionaryWordsInPassword** As Boolean

Retrieves or specifies whether dictionary words are prohibited in passwords.

```
HRESULT ProhibitDictionaryWordsInPassword([out, retval]
VARIANT_BOOL* pVal);
HRESULT ProhibitDictionaryWordsInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Prohibit dictionary is enable: " & CStr
(oCPS.ProhibitDictionaryWordsInPassword)
```

Specifying:

```
oCPS.ProhibitDictionaryWordsInPassword = True
```

EFT v6.3 and later

- Property **ProhibitReverseDictionaryWordsInPassword** As Boolean

Retrieves or specifies whether reverse dictionary words are prohibited in passwords.

```
HRESULT ProhibitReverseDictionaryWordsInPassword([out,
retval] VARIANT_BOOL* pVal);
HRESULT ProhibitReverseDictionaryWordsInPassword([in]
VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Prohibit reverse dictionary is enable: " &
CStr(oCPS.ProhibitReverseDictionaryWordsInPassword)
```

Specifying:

```
oCPS.ProhibitReverseDictionaryWordsInPassword = True
```

EFT v6.3 and later

- Property **RequireDigitsInPassword** As Boolean

Retrieves or specifies whether digits are required in passwords.

```
HRESULT RequireDigitsInPassword([out, retval] VARIANT_BOOL*
pVal);
HRESULT RequireDigitsInPassword([in] VARIANT_BOOL newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Digits is required: " & CStr
(oCPS.RequireDigitsInPassword)
```

Specifying:

```
oCPS.RequireDigitsInPassword = False
```

EFT v6.3 and later

- Property **RequireLowerCaseInPassword** As Boolean

Retrieves or specifies whether lowercase is required in passwords.

```
HRESULT RequireLowerCaseInPassword([out, retval] VARIANT_
BOOL* pVal);
HRESULT RequireLowerCaseInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Lower case is required: " & CStr
(oCPS.RequireLowerCaseInPassword)
```

Specifying:

```
oCPS.RequireLowerCaseInPassword = False
```

EFT v6.3 and later

- Property **RequireNonAlphaNumericInPassword** As Boolean

Retrieves or specifies whether non-alphanumeric characters are required in passwords.

```
HRESULT RequireNonAlphaNumericInPassword([out, retval]
VARIANT_BOOL* pVal);
HRESULT RequireNonAlphaNumericInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Non-alphanumeric is required: " & CStr
(oCPS.RequireNonAlphaNumericInPassword)
```

Specifying:

```
oCPS.RequireNonAlphaNumericInPassword = True
```

EFT v6.3 and later

- Property **RequireUnicodeInPassword** As Boolean

```
HRESULT RequireUnicodeInPassword([out, retval] VARIANT_BOOL*
pVal);
HRESULT RequireUnicodeInPassword([in] VARIANT_BOOL newVal);
```

EFT v7.4 and later

- Property **RequireUpperCaseInPassword** As Boolean

Retrieves or specifies whether uppercase is required in passwords.

```
HRESULT RequireUpperCaseInPassword([out, retval] VARIANT_
BOOL* pVal);
HRESULT RequireUpperCaseInPassword([in] VARIANT_BOOL
newVal);
```

Example (VBScript):**Retrieving:**

```
MsgBox "Upper case is required: " & CStr
(oCPS.RequireUpperCaseInPassword)
```

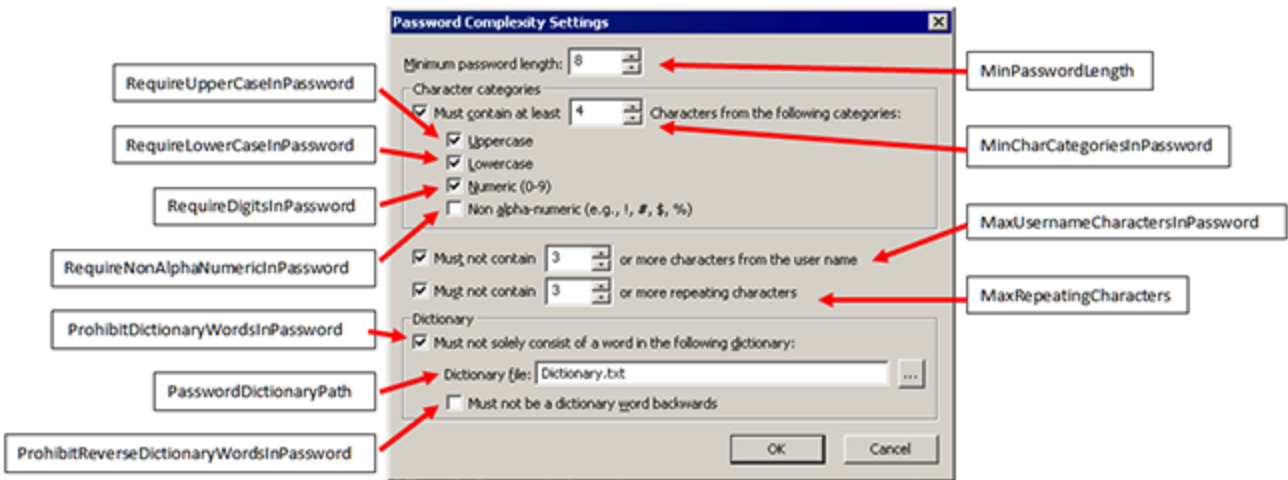
Specifying:

```
oCPS.RequireUpperCaseInPassword = False
```

EFT v6.3 and later

Dialog Box Equivalents

The ICIComplexPasswordSettings interface members correlate to the following fields and controls in the EFT administration interface. (Select the "Enforce strong passwords" check box.)



ICConnectedUser Interface

The ICConnectedUser interface is used to retrieve information about connected users. See also the [ICISite::ConnectedUsers](#) property.

```
interface ICConnectedUser : IDispatch
```

NOTE: "Function" indicates that the member is a method; "Property" indicates that it is a property.

Members

- Property **ConnectedAt** As Date read-only
Retrieves the date/time the user connected.

```
HRESULT ConnectedAt([out,retval] DATE* pVal);
```

Example (VBScript):

```
MsgBox "Connected at: " & oCU.ConnectedAt
```

EFT v6.3 and later

- Property **ConnectionType** As String read-only
Retrieves the type of connection (e.g., FTP, HTTP, HTTPS, SFTP).

```
HRESULT ConnectionType([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Connection type: " & oCU.ConnectionType
```

EFT v6.3 and later

- Property **DataConnection** As String read-only

Retrieves the data connection state ("No data connection.", "RETRieving file", "STORing file", "RETRieving folder listing", and "N/A").

```
HRESULT DataConnection([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Data connection state: " & oCU.DataConnection
```

EFT v6.3 and later

- Property **FileName** As String read-only

Retrieves the filename currently downloading/uploading.

```
HRESULT FileName([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "File name:" & CStr(oCU.FileName)
```

EFT v6.3 and later

- Property **FileSize** As Signed Double Long read-only

Retrieves the file size of the file currently downloading.

```
HRESULT FileSize([out,retval] LONGLONG* pVal);
```

Example (VBScript):

```
MsgBox "File size: " & CStr(oCU.FileSize)
```

EFT v6.3 and later

- Property **ID** As Long read-only

Retrieves the connection ID.

```
HRESULT ID([out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "ID: " & oCU.ID
```

EFT v6.3 and later

- Property **IP** As String read-only

Retrieves the user's connecting IP address.

```
HRESULT IP([out,retval] BSTR* pVal);
```

Example (VBScript):

Retrieving:

```
MsgBox "IP: " & oCU.IP
```

EFT v6.3 and later

- Property **Log** As String

Retrieves the name of the transfer log.

```
HRESULT Log([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Log: " & oCU.Log
```

EFT v6.3 and later

- Property **Login** As String read-only

Retrieves the username.

```
HRESULT Login([out, retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "User: " & oCU.Login
```

EFT v6.3 and later

- Property **Structure** As String read-only

Retrieves data connection structure ("File" or "N/A").

```
HRESULT Structure([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Structure: " & oCU.Structure
```

EFT v6.3 and later

- Property **TransferMode** As String read-only

Retrieves the transfer mode. (Possible values are "stream" or "N/A")

```
HRESULT TransferMode([out,retval] BSTR* pVal);
```

Example (VBScript):

```
MsgBox "Structure: " & oCU.Structure
```

EFT v6.3 and later

- Property **TransferredSize** As Signed Double Long read-only

Retrieves the transferred size in bytes.

```
HRESULT TransferredSize([out,retval] LONGLONG* pVal);
```

Example (VBScript):

```
MsgBox "Transferred size: " & CStr(oCU.TransferredSize)
```

EFT v6.3 and later

- Property **TransferTime** As Long read-only

Retrieves the transfer time in seconds.

```
HRESULT TransferTime([out,retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "Transfer time: " & CStr(oCU.TransferTime)
```

EFT v6.3 and later

- Property **type** As String read-only

Retrieves the mode of connection ("ASCII", "IMAGE", "EBCDIC"). Only for FTP, FTPS explicit, FTPS implicit connections; for others it returns "N/A."

```
HRESULT Type([out,retval] BSTR* pVal);
```

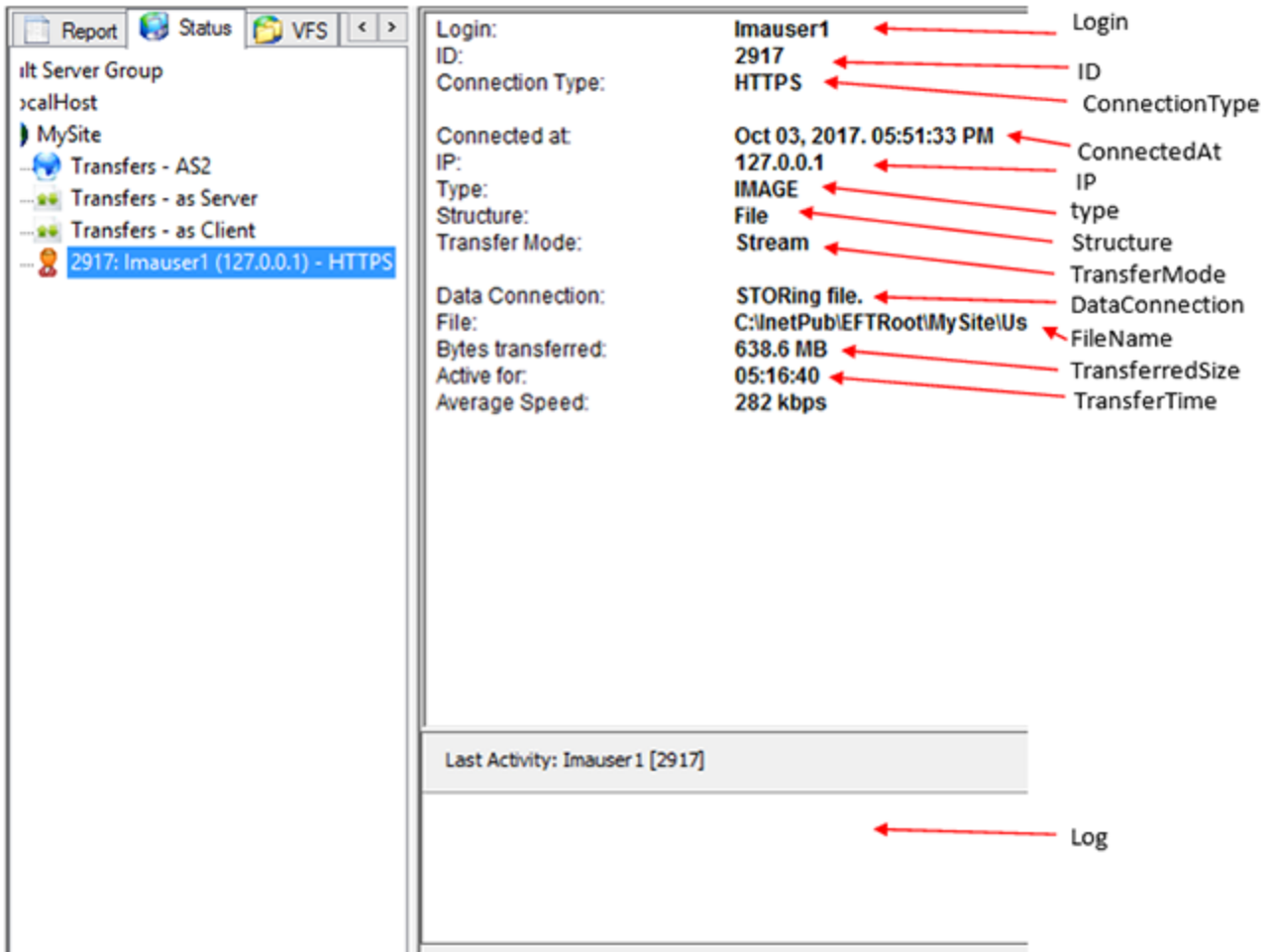
Example (VBScript):

```
MsgBox "Type: " & oCU.Type
```

EFT v6.3 and later

Dialog Box Equivalents

The ICICConnectedUser interface members correlate to the following fields and controls in the EFT administration interface. (EFT Enterprise v6.3 is shown in illustration; the properties are the same in later versions.)



ICIUserAccountActionParams

The UserAccountActionParams interface is used to specify parameters for the User Account Action.

Members

- Property **Action As** [UserAccountActionType](#)

```
HRESULT Action([out, retval] UserAccountActionType *pVal);
HRESULT Action([in] UserAccountActionType newVal);
```

EFT v7.4.13

- Property **Parameter** As Variant

```
HRESULT Parameter([out, retval] VARIANT *pVal);
HRESULT Parameter([in] VARIANT newVal);
```

EFT v7.4.13

ICIAbortUserOperationActionParams

The **ICIAbortUserOperationActionParams** interface sets or retrieves parameters used to abort a message-composed operation for the EFT Outlook Add-In, Send, Reply, or Dropoff portals, or Request File-related events, typically because of blocked or modified content in an ICAP scan.

Web Objects

This chapter contains Web-related COM objects.

Workspaces Settings

In addition to the [ICIWorkspacePermissions](#) and [ICIWorkspace](#) interfaces, Workspaces settings are also in the [ICIServer](#), [ICISite](#), and [ICIClientSettings](#) interfaces, and enums. Also refer to the [ICIAddressBook](#) interface.

In ICIServer:

- [WSSeatsMax](#)
- [WSSeatsUsed](#)

In ICISite:

- [CreateGuestSettingsLevel](#)
- [DropOffPortalPath](#)
- [EnableWorkspaces](#)
- [EnableWorkspacesDropOff](#)
- [EnableWorkspacesExternalUsers](#)
- [EnableWorkspacesOAI](#)
- [GetClientDataField](#), [SetClientDataField](#)
- [GetPrivacyPolicy](#), [SetPrivacyPolicy](#)
- [GetTermsOfService](#), [SetTermsOfService](#)
- [GetTransactionalWorkspacesIDs](#)
- [GetWorkspace](#)

- [GetWorkspaceGUID](#)
- [GetWorkspaceMaxDuration](#)
- [SetWorkspaceMaxDuration](#)
- [GetWorkspacesDropOffAddressBook](#)
- [SetWorkspacesDropOffAddressBook](#)
- [GetWorkspacesDropOffMaxDuration](#)
- [SetWorkspacesDropOffMaxDuration](#)
- [GetWorkspacesDropOffMaxMessageSize](#)
- [SetWorkspacesDropOffMaxMessageSize](#)
- [GetWorkspacesOAIMaxDuration](#)
- [SetWorkspacesOAIMaxDuration](#)
- [GuestSettingsLevels](#)
- [RemoveWorkspace](#)
- [SendPortalPath](#)
- [WorkspacesDomainsToForbidInviteeAddresses](#)
- [WorkspacesOAIDomainsToLimitInviteeAddresses](#)
- [WorkspacesDropOffAllowRecipientAddresses](#)
- [WorkspacesDropOffDomainsToLimitRecipientAddresses](#)
- [WorkspacesDropOffEnableRecaptcha](#)
- [WorkspacesDropOffLimitMessageSize](#)
- [WorkspacesDropOffRecaptchaSecretKey](#)
- [WorkspacesDropOffRecaptchaSiteKey](#)
- [WorkspacesDropOffSecureMessageBodyPolicy](#)
- [WorkspacesEnableDomainsToForbidInviteeAddresses](#)
- [WorkspacesEnableDomainsToLimitInviteeAddresses](#)
- [WorkspacesGAEnableParticipationlessAccountPolicy](#)
- [WorkspacesGAGrantHF](#)
- [WorkspacesGAParticipationlessAccountPolicy](#)
- [WorkspacesGAParticipationlessAccountPolicyDelayInDays](#)
- [WorkspacesOAIDomainsToLimitInviteeAddresses](#)
- [WorkspacesOAIDomainsToForbidInviteeAddresses](#)
- [WorkspacesOAIEnableDomainsToForbidInviteeAddresses](#)
- [WorkspacesOAIEnableDomainsToLimitInviteeAddresses](#)

- [WorkspacesOAISharingMode](#)
- [WSSeatsRemaining](#)
- [WSSeatsUsed](#)

In ICIClientSettings:

- [GetAllowWorkspaces](#)
- [SetAllowWorkspaces](#)
- [GetAllowWorkspacesOAI](#)
- [SetAllowWorkspacesOAI](#)
- **GDPR-related settings:**
 - [CompanyName](#)
 - [Duns](#)
 - [GetGDPRPrivacyPolicyConsentStatus](#), [SetGDPRPrivacyPolicyConsentStatus](#)
 - [GetGDPRSubjectStatus](#), [SetGDPRSubjectStatus](#)
 - [GetGDPRTermsOfServiceConsentStatus](#), [SetGDPRTermsOfServiceConsentStatus](#)
 - [Mobile](#)
 - [PartnerId](#)
 - [ResetAllFieldAttributeDefaults](#)

In the Enum Reference:

- [CaptchaType](#)
- [EventProperty](#), Workspace properties
- [SizeUnits](#)
- [WorkspacesOAISharingModeType](#)
- [WorkspacesParticipationlessAccountPolicy](#)
- [WorkspacesSecureMessageBodyPolicy](#)

Dialog Reference

For EFT v8, all Workspaces settings were moved to the Site > Web tab. Refer to [ICISite Interface](#).

ICIWorkspacePermissions

The ICIWorkspacePermissions interface properties are used to specify permissions of a particular folder used in Workspaces. (Refer to [ICIWorkspace](#) for examples.)

- Property **CreateFolder** As Boolean

```
HRESULT CreateFolder([out, retval] VARIANT_BOOL *pVal);
HRESULT CreateFolder([in] VARIANT_BOOL newVal);
```

EFT v7.1 and later

- Property **DeleteFile** As Boolean

```
HRESULT DeleteFile([out, retval] VARIANT_BOOL *pVal);
HRESULT DeleteFile([in] VARIANT_BOOL newVal);
```

EFT v7.1 and later

- Property **DeleteFolder** As Boolean

```
HRESULT DeleteFolder([out, retval] VARIANT_BOOL *pVal);
HRESULT DeleteFolder([in] VARIANT_BOOL newVal);
```

EFT v7.1 and later

- Property **DownloadFile** As Boolean

```
HRESULT DownloadFile([out, retval] VARIANT_BOOL *pVal);
HRESULT DownloadFile([in] VARIANT_BOOL newVal);
```

EFT v7.1 and later

- Property **RenameFileFolder** As Boolean

```
HRESULT RenameFileFolder([out, retval] VARIANT_BOOL *pVal);
HRESULT RenameFileFolder([in] VARIANT_BOOL newVal);
```

EFT v7.1 and later

- Property **UploadFile** As Boolean

```
HRESULT UploadFile([out, retval] VARIANT_BOOL *pVal);
HRESULT UploadFile([in] VARIANT_BOOL newVal);
```

EFT v7.1 and later

ICIWorkspace

The ICIWorkspace properties and methods are used to retrieve and to create a Workspace.

- Function **AddParticipant**(bstrName As String) As Object

```
HRESULT AddParticipant([in] BSTR bstrName, [out, retval];
```

EFT v7.1 and later

- Property **ExpireTime** As Date (read only)

```
HRESULT ExpireTime([out, retval] DATE* pVal);
```

EFT v7.2.1

- Function **GetParticipantPermissions**(bstrName As String) As Object

```
HRESULT GetParticipantPermissions([in] BSTR bstrName, [out,
retval] IDispatch** ppdisplPermissions);
```

EFT v7.1 and later

- Property **Name** As String

```
HRESULT Name([out, retval] BSTR *pVal);
```

EFT v7.1 and later

- Property **Owner** As String

```
HRESULT Owner([out, retval] BSTR *pVal);
```

EFT v7.1 and later

- Property **ParticipantList** As Variant

```
HRESULT ParticipantList([out, retval] VARIANT* pVal);
```

EFT v7.1 and later

- Property **Path** As String

```
HRESULT Path([out, retval] BSTR *pVal);
```

EFT v7.1 and later

- Function **RemoveParticipant**(bstrName As String) As Object

```
HRESULT RemoveParticipant([in] BSTR bstrName, [out, retval]
```

EFT v7.1 and later

- Sub **SetParticipantPermissions**(bstrName As String, ppdisplPermissions As Object)

```
HRESULT SetParticipantPermissions([in] BSTR bstrName, [in]
IDispatch* ppdisplPermissions);
```

EFT v7.1 and later

- Property **TimeCreated** As String

```
HRESULT TimeCreated([out, retval] BSTR *pVal);
```

EFT v7.1 and later

Workspace Examples, VBScript script fragments:

1. Getting a Workspace object and printing its information:

```
guid = site.GetWorkspaceGUID("/Usr/testuser/share")
Set ws = site.GetWorkspace(guid)
WScript.Echo vbCrLf
WScript.Echo "Workspace name: " & ws.Name & vbCrLf
WScript.Echo "Workspace path: " & ws.Path & vbCrLf
WScript.Echo "Workspace Owner: " & ws.Owner & vbCrLf
WScript.Echo "Workspace TimeCreated: " & ws.TimeCreated &
vbCrLf
WScript.Echo "Workspace ParticipantList: "
Dim arList : arList = ws.ParticipantList
for i = LBound(arList) to UBound(arList)
WScript.Echo CStr(i) & " - " & arList(i)
next
```

2. A function that removes all participants from a workspace (make sure to call SetWorkspace after calling this function):

```
Function RemoveParticipants (ws)
    Dim arList : arList = ws.ParticipantList
    for i = LBound(arList) to UBound(arList)
        WScript.Echo CStr(i) & " - Deleting " & arList(i)
        ws.RemoveParticipant(arList(i))
    next
End Function
```

3. Changing a user's permissions:

```
Set perm = ws.GetParticipantPermissions("user")

WScript.Echo "Permissions for user" & vbCrLf
WScript.Echo "DownloadFile: " & perm.DownloadFile & vbCrLf
WScript.Echo "UploadFile: " & perm.UploadFile & vbCrLf
WScript.Echo "DeleteFile: " & perm.DeleteFile & vbCrLf
WScript.Echo "RenameFileFolder: " & perm.RenameFileFolder &
vbCrLf
```

```

WScript.Echo "CreateFolder: " & perm.CreateFolder & vbCrLf
WScript.Echo "DeleteFolder: " & perm.DeleteFolder & vbCrLf

' Change permissions
perm.DownloadFile = true
perm.UploadFile = true
perm.DeleteFile = true
perm.RenameFileFolder = true
perm.CreateFolder = true
perm.DeleteFolder = true

' Set permissions
ws.SetParticipantPermissions "user", perm

' Set workspace
site.SetWorkspace guid, ws

```

ICIAddressBook

The ICIAddressBook properties and methods are used to manage the Workspaces Drop-Off portal address book.

- Sub **AddRecord**(Name As String, Address As String)

```
HRESULT AddRecord([in] BSTR name, [in] BSTR address);
```

EFT v7.4.2 and later

- Property **AllRecords** As Variant

read only

```
HRESULT AllRecords([out, retval] VARIANT* pVal);
```

EFT v7.4.2 and later

- Sub **Clear**()

```
HRESULT Clear();
```

EFT v7.4.2 and later

- Function **FindRecord**(Name As String) As String

```
HRESULT FindRecord([in] BSTR name, [out, retval] BSTR
*address);
```


EFT v7.4.2 and later

- Sub **RemoveRecord**(Name As String)

```
HRESULT RemoveRecord([in] BSTR name);
```

EFT v7.4.2 and later

ICIPrivacyPolicy Interface

The ICIPrivacyPolicy interface is used to specify the Privacy Policy on the Site for all web portals. The Privacy Policy applies to new account registration, anonymous pickup/dropoff, and after first login.

Members

- Property **AgreementLabels** As String

Retrieves or modifies

```
HRESULT AgreementLabels([out, retval] BSTR *pVal);
HRESULT AgreementLabels([in] BSTR val);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.AgreementLabels = "C:\pplabels.json"
```

EFT v8 and later

- Property **AgreementPath** As String

Retrieves or modifies privacy policy agreement path.

```
HRESULT AgreementPath([out, retval] BSTR *pVal);
HRESULT AgreementPath([in] BSTR val);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.AgreementPath = "C:\pp.json"
```

EFT v8 and later

- Property **EffectiveDate** As Variant

Retrieves or modifies privacy policy effective date.

```
HRESULT EffectiveDate([out, retval] VARIANT *pVal);
HRESULT EffectiveDate([in] VARIANT newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.WScript.echo pp.EffectiveDate
```

EFT v8 and later

- Property **Mode** As [GDPRPrivacyPolicyMode](#)

Retrieves or modifies Privacy Policy mode.

```
HRESULT Mode([out, retval] GDPRPrivacyPolicyMode *pVal);
HRESULT Mode([in] GDPRPrivacyPolicyMode newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.Mode = GDPRPrivacyPolicyMode_Shown
```

EFT v8 and later

- Property **PreventFTPSUntilConsentGranted** As Boolean

Retrieves or modifies "Prevent S/FTP/S until PP is agreed to" option state.

```
HRESULT PreventFTPSUntilConsentGranted([out, retval]
VARIANT_BOOL *pVal);
HRESULT PreventFTPSUntilConsentGranted([in] VARIANT_BOOL
newVal);
```

Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.PreventFTPSUntilConsentGranted
```

EFT v8 and later

- Property **RevertToImpliedConsentForNotEUDataSubjects** As Boolean

Retrieves or modifies "Revert to implied consent for users identified as not being EU data subjects" option state.

```
HRESULT RevertToImpliedConsentForNotEUDataSubjects([out,
retval] VARIANT_BOOL *pVal);
```

```
HRESULT RevertToImpliedConsentForNotEUDataSubjects ([in]
VARIANT_BOOL newVal);
```

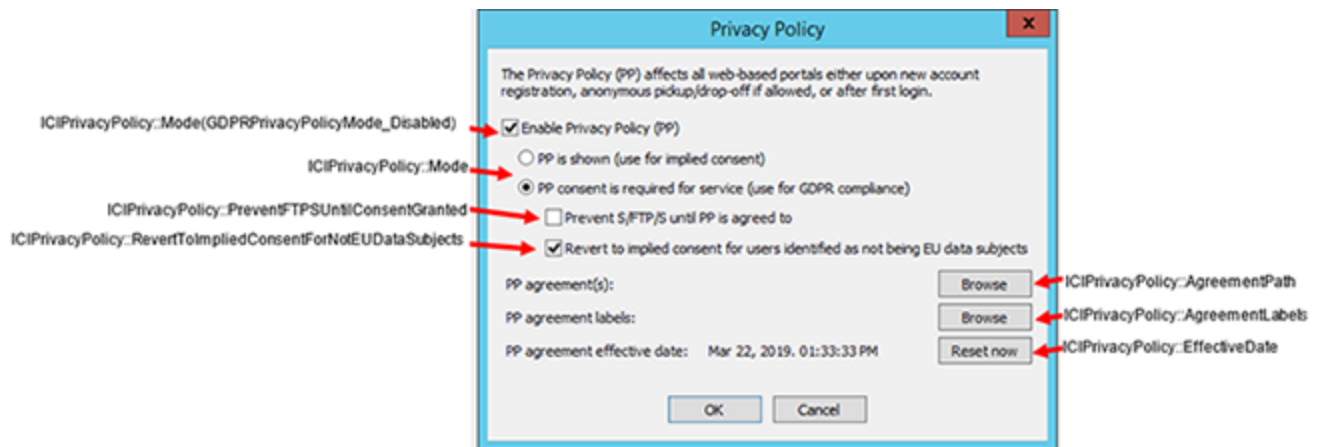
Example (VBScript):

```
set pp = oSite.GetPrivacyPolicy
pp.RevertToImpliedConsentForNotEUDataSubjects =False
```

EFT v8 and later

Dialog Box Equivalents

The Privacy Policy dialog box is available from the Site > Web tab.



ICITermsOfService Interface

The ICITermsOfService interface provides the properties of the Terms of Service.

Members

- Property **AgreementLabels** As String

Retrieves or modifies Terms of Service labels.

```
HRESULT AgreementLabels([out, retval] BSTR *pVal);
HRESULT AgreementLabels([in] BSTR val);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.AgreementLabels = "C:\toslabels.json"
```

EFT v8 and later

- Property **AgreementPath** As String

Retrieves or modifies Terms of Service agreement path.

```
HRESULT AgreementPath([out, retval] BSTR *pVal);
HRESULT AgreementPath([in] BSTR val);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.AgreementPath = "C:\tos.json"
```

EFT v8 and later

- Property **EffectiveDate** As Variant

Retrieves or modifies Terms of Service effective date.

```
HRESULT EffectiveDate([out, retval] VARIANT *pVal);
HRESULT EffectiveDate([in] VARIANT newVal);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
WScript.echo tos.EffectiveDate
```

EFT v8 and later

- Property **Mode** As [GDPRTermsOfServiceMode](#)

Retrieves or modifies Terms of Service mode.

```
HRESULT Mode([out, retval] GDPRTermsOfServiceMode *pVal);
HRESULT Mode([in] GDPRTermsOfServiceMode newVal);
```

Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.Mode = GDPRTermsOfServiceMode_Implied
```

EFT v8 and later

- Property **PreventFTPSUntilAgreed** As Boolean

Retrieves or modifies "Prevent FTPS connection until agreed" option state.

```
HRESULT PreventFTPSUntilAgreed([out, retval] VARIANT_BOOL
*pVal);
HRESULT PreventFTPSUntilAgreed([in] VARIANT_BOOL newVal);
```

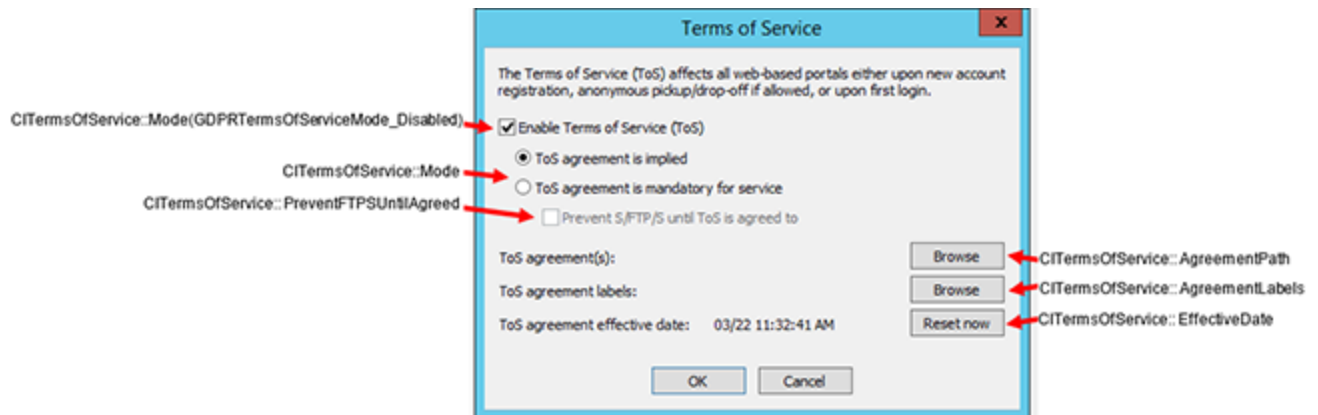
Example (VBScript):

```
set tos = oSite.GetTermsOfService
tos.PreventFTPSUntilAgreed =True
```

EFT v8 and later

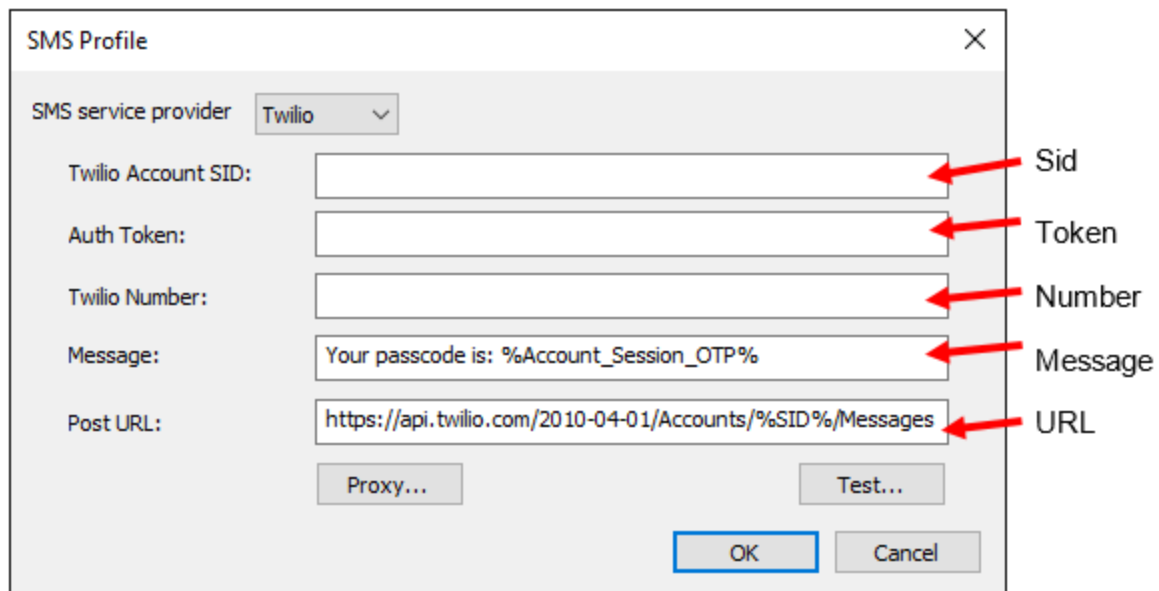
Dialog Box Equivalents

The Terms of Service dialog box is available from the Site > Web tab.



ICITwilioSmsProviderProfile

The ICITwilioSmsProviderProfile interface is used to manage the Twilio SMS profile for multifactor authentication.



Properties

- Property **Message** As String

Provide the Message that will be sent in the SMS, or keep the default. Do not edit anything between the percent symbols.

```
HRESULT Message([out, retval] BSTR* pVal);
HRESULT Message([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Number** As String

Provide the Twilio Number copied from your Twilio account. You must include the country code and the + PLUS SIGN (e.g., in the USA, +1) in front of the number.

```
HRESULT Number([out, retval] BSTR* pVal);
HRESULT Number([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Sid** As String

Provide the Twilio Account SID copied from your Twilio account.

```
HRESULT Sid([out, retval] BSTR* pVal);
HRESULT Sid([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Token** As String

Provide the Auth Token copied from your Twilio account.

```
HRESULT Token([out, retval] BSTR* pVal);
HRESULT Token([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Url** As String

Provide the Post URL copied from your Twilio account.

```
HRESULT Url([out, retval] BSTR* pVal);
HRESULT Url([in] BSTR newVal);
```

EFT v8.0.4 and later

ICIUploadForm

The ICIUploadForm interface is used to manage Upload forms on a Site.

The screenshot shows the 'Edit Existing Form' dialog box. It contains the following fields and controls:

- Enabled:** A checked checkbox.
- Form name:** A text box containing 'This is my Form name'. An annotation 'Name' points to this field.
- Description:** A text box containing 'This is the Description'. An annotation 'Description' points to this field.
- Display name:** A text box containing 'This is the Display name'. An annotation 'DisplayName' points to this field.
- Display instructions:** A text box containing 'This is the Display instructions'. An annotation 'DisplayInstructions' points to this field.
- Show form when:** Two radio buttons: 'For each file uploaded' (selected) and 'Once for batch upload'. An annotation 'type' points to the 'Once for batch upload' radio button.
- Show form to:** A dropdown menu showing 'All Users' and '(Permission group)'. An annotation 'ShowFormToPermissionGroup' points to the dropdown.
- Elements:** A table with columns 'Name', 'Type', 'Value', and 'Required'. It contains one row: 'MyElement', 'Drop-down', 'val1,val2,val3', 'No'. An annotation 'ElementsList GetElement' points to the table.
- Buttons:** 'Add', 'Edit', 'Remove', 'OK', and 'Cancel'. An annotation 'AddElement DeleteElement UpdateElement' points to the 'Remove' button.

Properties and Methods

- Sub **AddElement**(pElement As Object)

```
HRESULT AddElement([in] IDispatch* pElement);
```

EFT v8.0.4 and later

- Sub **DeleteElement**(elementName As String)

```
HRESULT DeleteElement([in] BSTR elementName);
```

EFT v8.0.4 and later

- Property **Description** As String

```
HRESULT Description([out, retval] BSTR *pVal);
HRESULT Description([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **DisplayInstructions** As String

```
HRESULT DisplayInstructions([out, retval] BSTR *pVal);
HRESULT DisplayInstructions([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **DisplayName** As String

```
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ElementsList** As Variant

```
HRESULT ElementsList([out, retval] VARIANT *pVal);
```

EFT v8.0.4 and later

- Property **Enabled** As Boolean

```
HRESULT ([out, retval] VARIANT_BOOL *pVal);
HRESULT Enabled([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Function **GetElement**(elementName As String) As Long

```
HRESULT GetElement([in] BSTR elementName, [out, retval]
    long *elementIndex);
```

EFT v8.0.4 and later

- Property **Name** As String

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **ShowFormToPermissionGroup** As String

```
HRESULT ShowFormToPermissionGroup([out, retval] BSTR *pVal);
HRESULT ShowFormToPermissionGroup([in] BSTR newVal);
```


EFT v8.0.4 and later

- Property **type** As [UploadFormComType](#)

```
HRESULT Type([out, retval] UploadFormComType *pVal);
HRESULT Type([in] UploadFormComType newVal);
```

EFT v8.0.4 and later

- Sub **UpdateElement**(pElement As Object, elementIndex As Long)

```
HRESULT UpdateElement([in] IDispatch* pElement, [in] long
elementIndex);
```

EFT v8.0.4 and later

ICIUploadFormElement

The ICIUploadFormElement is used to manage an Upload Form element.

The screenshot shows a 'Form Element' dialog box with the following fields and labels indicated by red arrows:

- Name**: Points to the 'Element name' text box.
- DisplayLabel**: Points to the 'Display label' text box.
- type**: Points to the 'Type' dropdown menu, which currently shows 'String'.
- Values**: Points to the 'Values' text box.
- Required**: Points to the 'Required' checkbox, which is checked.

At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Properties

- Property **DisplayLabel** As String

```
HRESULT DisplayLabel([out, retval] BSTR *pVal);
HRESULT DisplayLabel([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Name** As String

```
HRESULT Name([out, retval] BSTR *pVal);
HRESULT Name([in] BSTR newVal);
```

EFT v8.0.4 and later

- Property **Required** As Boolean

```
HRESULT Required([out, retval] VARIANT_BOOL *pVal);
HRESULT Required([in] VARIANT_BOOL newVal);
```

EFT v8.0.4 and later

- Property **type** As [UploadFormElementComType](#)

```
HRESULT Type([out, retval] UploadFormElementComType *pVal);
HRESULT Type([in] UploadFormElementComType newVal);
```

EFT v8.0.4 and later

- Property **Values** As String

```
HRESULT Values([out, retval] BSTR *pVal);
HRESULT Values([in] BSTR newVal);
```

EFT v8.0.4 and later

WTC-Related Settings

Web Transfer Client Settings are available in the [ICIServer Interface](#), [ICIServerNodeInfo Interface](#), [ICISite Interface](#), [ICIClientSettings Interface](#), and [Enum Reference](#).

In the ICIServer Interface:

- Property **WTCSessionsUsed** As Long

```
HRESULT WTCSessionsUsed([out, retval] LONG* pVal);
```

EFT v7.3.2 and later

- Property **WTCSessionsMax** As Long

```
HRESULT WTCSessionsMax([out, retval] LONG* pVal);
```

EFT v7.3.2 and later

In the ICIServerNode Interface:

- Property **ActiveWTCSessions** As Long

```
HRESULT ActiveWTCSessions([out, retval] LONG* plVal);
```

EFT 7.3

In the ICISite Interface:

- Property **WTCSessionsActive** As Long (read only)

Prior to v7.3.2, this property retrieves the number of active WTC sessions in use.

In v7.3.2 and later, this property retrieves number of used WTC client access licences.

```
HRESULT WTCSessionsActive(
    [out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "WTC sessions count: " & CStr
(oSite.WTCSessionsActive)
```

EFT v6.3 and later

- Property **WTCSessionsRemaining** As Long (read only)

Prior to v7.3.2, this property retrieves the number of WTC sessions available.

In v7.3.2 and later, this property retrieves the maximum number of WTC client access licences.

```
HRESULT WTCSessionsRemaining(
    [out, retval] LONG* pVal);
```

Example (VBScript):

```
MsgBox "WTC sessions remaining: " & CStr
(oSite.WTCSessionsRemaining)
```

EFT v6.3 and later

In the ICIClientSettings Interface:

- Function **GetAppletEnabled**([pInherited]) As Boolean

For an example using this method, refer to the GetCertInfo script in [Script Examples](#).

Retrieve whether Web Transfer Client is allowed for a user or Settings Template.

```
HRESULT GetAppletEnabled(
    [out, optional] VARIANT *pInherited,
    [out, retval] VARIANT_BOOL *pVal);
```

VARIANT_BOOL equals True(-1) = Enabled; False(0) = Disabled

Example for v6.1 and later:

```
If Client.AppletEnabled(isInherited) And isInherited Then  
MsgBox "WTC is allowed for client because it is allowed for  
Settings Template"  
End If
```

EFT v3.5.1 and later

In the Enum Reference:**ServerModule** (v6.1 and later)

Used in the ICIServer Interface [ModuleRegistrationState](#) method to retrieve an EFT module state. Used with the RegistrationState enumerator.

- WTCModule = 3

Enums

Enumerators, or "enums," are constants or variables used within methods and properties.

For example, in the ICIBackupArchiveInfo and ICIServer Interface, the **ARMDatabaseType** enum is used to specify whether the ARM database is Oracle or SQLServer.

Below is an alphabetical reference for the enums used in the EFT COM API.

Jump to: [C](#) - [D](#) - [E](#) - [F](#) - [G](#) - [H](#) - [I](#) - [L](#) - [M](#) - [N](#) - [P](#) - [R](#) - [S](#) - [T](#) - [U](#) - [V](#) - [W](#)

ADAuthManagerLogonAttribute (v6.3 and later)

Used in the [ICIADAuthManagerSettings](#) **LogonAttribute** property.

- NT4AccountName = 0
- UserPrincipalName = 2
- DisplayName = 1
- CommonName = 3

ADAuthManagerType (v6.3 and later)

Used in the [ICIADAuthManagerSettings](#) **type** property.

- AuthManagerAD = 0
- AuthManagerNTLM = 1

AdminAccountType (v6.1 and later)

Used to specify the type of EFT administrator account in the [ICIAdminAccount](#) **type** property, and the ICIServer [CreateAdmin](#), [CreateAdminID](#), and [RemoveServerAdminAccount](#) methods.

- EFTAccount = 0
- LocalComputerAccount = 1
- ADAccount = 2
- ADGroupAccount = 3 (added in v8)

AdminLoginType (v6.1 and later)

Used to specify the way an EFT administrator connects to the server in the ICIServer [ConnectEx](#), [GetBackupArchiveInfo](#), and [RestoreServerConfigurationEx](#) methods.

- EFTLogin = 0
- IWALogin = 1
- NetLogon = 2

AdminPermissionsPolicy (v6.1 and later)

Used in the [ICAdminPermission](#) property.

- ChangePassword = 4; Change password administrator; disable users and change passwords
- COMManagement = 5; Can manage server using the API
- EventRulesManagement = 7; Can manage Event Rules (EFT v7.2 and later)
- ManagePersonalData = 8 (added in v8)
- ReportManagement = 6; Can run and edit reports
- ServerManagement = 0; Server administrator; full control
- SiteManagement = 1; Site administrator; full control except over server settings
- STManagement = 2; Settings Template administrator; control over assigned template and users belonging to that template
- UserCreation = 3; User administrator; create and disable users and change passwords

AgeRestrictionsType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) AgeRestrictions property.

- AgeRestrictions_Unknown,
- AgeRestrictions_EFT,
- AgeRestrictions_External,
- AgeRestrictions_Other,
- AgeRestrictions_NotApplicable

ARMAuthenticationType (v6.3 and later)

Used in the [ICBackupArchiveInfo](#) and ICIServer - Server Interface [ARMAuthenticationType](#) properties to specify the ARM database authentication type.

- WindowsAuthentication = 0
- SQLServerAuthentication = 1

ARMDatabaseType (v6.3 and later)

Used in the [ICBackupArchiveInfo](#) and ICIServer - Server Interface [ARMDatabaseType](#) properties to specify the ARM database type.

- Oracle = 0; Oracle database
- SQLServer = 1; SQL Server database

AS2AuthenticationType (v6.3 and later)

Used in the ICIAS2InboundSettings [AuthenticationType](#) property used to specify the AS2 authentication mechanism.

- AS2AuthTypeUsernameAndPassword = 0
- AS2AuthTypeMsgLevelSecurity = 1

AS2ContentType (v6.1 and later)

Used to specify the type of the document to send in the [ICIAS2OutboundSettings](#) and [ICIAS2SendActionParams](#) **ContentType** properties.

- X12 = 0
- EDIConsent = 3
- EDIFACT = 1
- Binary = 4
- XML = 2
- Plaintext = 5

AS2DoubleContentPolicy (v6.3 and later)

Used in the ICIAS2InboundSettings [DoubleContentPolicy](#) property to specify the AS2 duplicate contents policy.

- AS2DoubleContentPolicyOverwrite = 0
- AS2DoubleContentPolicyNurate = 1
- AS2DoubleContentPolicyReject = 2

AS2DoubleMsgIDPolicy (v6.3 and later)

Used in the ICIAS2InboundSettings [DoubleMsgIDPolicy](#) property to specify the AS2 duplicate message ID policy.

- AS2DoubleMsgIDPolicyAccept = 0
- AS2DoubleMsgIDPolicyReject = 1

AS2EncryptionAlgorithm (v7.4.9 and later)

Used in the ICIAS2OutboundSettings [EncryptionAlgorithm](#) property to specify allowed encryption algorithms.

- Const Encryption_3DES = 0
- Const Encryption_DES = 1
- Const Encryption_AESCBC128 = 2

- Const Encryption_AESCBC192 = 3
- Const Encryption_AESCBC256 = 4

AS2EncryptMissingPolicy (v6.3 and later)

Used in the ICIAAS2InboundSettings [EncryptMissingPolicy](#) property.

- AS2EncryptMissingPolicyAccept = 0
- AS2EncryptMissingPolicyReject = 1

AS2IDMismatchPolicy (v6.3 and later)

Used in the ICIAAS2InboundSettings [EFTIDMismatchPolicy](#) and [PartnerIDMismatchPolicy](#) properties to specify the mismatch policy.

- AS2IDMismatchPolicyAccept = 0
- AS2IDMismatchPolicyReject = 1

AS2MICHash (v7.4.11 and later)

- MDN_AS_REQUESTED = 0,
- MDN_AS_REQUESTED_OR_SHA1,
- MDN_SHA1,
- MDN_MD5,
- MDN_None,
- MDN_SHA_256
- MDN_SHA_384,
- MDN_SHA_512,
- MDN_SHA_224,
- MDN_AS_REQUESTED_OR_SHA256,

AS2OutboundProfileStatus (v6.3 and later)

Used in the ICIClientSettings [AS2TestOutboundProfile](#) method to retrieve the result for AS2 Test Outbound Profile operation.

- AS2StatusOk = 0
- AS2StatusVerifyMDNFailed = 4
- AS2StatusNotRegistered = 1
- AS2StatusDatabaseFailure = 5
- AS2StatusCreateRequestFailed = 2

- AS2StatusUnexpectedError = 6
- AS2StatusSendDataFailed = 3

AS2ProxyType (v6.3 and later)

Used in the ICIAS2OutboundSettings [ProxyType](#) property to specify the AS2 proxy type.

- AS2ProxyDMZ = 0
- AS2ProxyHTTP = 1

AS2ReceiptDelivery (v6.1 and later)

Used to specify the receipt delivery mechanism in the [ICIAS2OutboundSettings](#) and [ICIAS2SendActionParamsReceiptDelivery](#) property.

- Sync = 0
- Async = 1

AS2ReceiptPolicy (v6.1 and later)

Used to specify the type of the receipt required in [the ICIAS2OutboundSettings](#) and [ICIAS2SendActionParamsReceiptPolicy](#) property.

- NoReceipt = 0
- UnsignedReceipt = 1
- SignedReceipt = 2

AS2SignatureAlgorithm (v7.4.9 and later)

The AS2SignatureAlgorithm enum is used in the ICIAS2OutboundSettings [SignatureAlgorithm](#) property.

- Signature_MD5 = 1
- Signature_SHA1 = 0
- Signature_SHA_224 = 5
- Signature_SHA_256 = 2
- Signature_SHA_384 = 3
- Signature_SHA_512 = 4

AS2SignatureMissingPolicy (v6.3 and later)

Used in the [ICIAS2InboundSettingsSignatureMissingPolicy](#) property to specify the AS2 signature-missing policy.

- AS2SignatureMissingPolicyAccept = 0
- AS2SignatureMissingPolicyReject = 1

AS2TransactionDirection (v6.3 and later)

Used in the [ICIAS2TransactionInfo Interface](#) **Direction** property to retrieve the direction of an AS2 transaction.

- AS2TransDirectionInbound = 0
- AS2TransDirectionOutbound = 1

AS2TransactionStatus (v6.3 and later)

Used in the [ICIAS2TransactionInfo Interface](#) **Status** property to retrieve the status of an AS2 transaction.

- AS2TransactionInProgress = 0
- AS2TransactionSuccess = 1
- AS2TransactionFailedCanResend = 2
- AS2TransactionFailedNoResend = 3
- AS2TransactionOkSendMDNFailed = 4

AutobanSensitivity (v6.1 and later)

Used to specify the sensitivity for Flood/Hammer Autoban settings in the ICISite Interface [AutobanSensitivity](#) property.

- SensitivityOff = 0
- SensitivityVeryLow = 1
- SensitivityLow = 2
- SensitivityMedium = 3
- SensitivityHigh = 4
- SensitivityVeryHigh = 5

AWLogMode (v6.5 and later)

AWLogMode enumeration represents different Advanced Workflow debug logging modes (see also ICIAAdvancedWorkflowParams::[LogMode](#)).

- AWLogNone = 0; no logging
- AWLogMinimal = 1; minimal logging
- AWLogNormal = 2; normal logging
- AWLogVerbose = 3; verbose logging

AWSS3AuthType (v7.4.6 and later)

Used in the [ICICloudAWSS3GeneralParams](#) interface.

- AWSS3AuthType_Anonymous = 0,
- AWSS3AuthType_Standard = 1

AWSS3BucketAccessStyle (v7.4.6 and later)

- Const AWSS3BucketAccessStyle_PathStyle = 1
- Const AWSS3BucketAccessStyle_VirtualHostedStyle = 0

AWSS3EncryptionType (v7.4.6 and later)

Used in the [ICICloudAWSS3GeneralParams](#) interface.

- AWSS3EncryptionType_None = 0,
- AWSS3EncryptionType_Server = 1
- AWSS3EncryptionType_Client = 2

AWSS3MatchingFileNamePolicy (v7.4.6 and later)

Used in the [ICICloudAWSS3UploadParams](#) and [ICICloudAWSS3DownloadParams](#) interfaces.

- AWSS3MatchingFileNamePolicy_Overwrite = 0
- AWSS3MatchingFileNamePolicy_Skip = 1
- AWSS3MatchingFileNamePolicy_Numerate = 2

AWSS3Region (v7.4.6 only; Refer to ICICloudAWSS3GeneralParams.[Region](#) property for v7.4.7 and later)

Used in the [ICICloudAWSS3GeneralParams](#) interface

- AWSS3Region_UsEast1 = 0,
- AWSS3Region_UsEast2 = 1
- AWSS3Region_UsWest1 = 2
- AWSS3Region_UsWest2 = 3
- AWSS3Region_EuWest1 = 4
- AWSS3Region_EuWest2 = 5
- AWSS3Region_EuWest3 = 14
- AWSS3Region_EuCentral1 = 6
- AWSS3Region_ApNortheast1 = 7
- AWSS3Region_ApNortheast2 = 8
- AWSS3Region_ApSouth1 = 9
- AWSS3Region_ApSoutheast1 = 10

- AWSS3Region_ApSoutheast2 = 11
- AWSS3Region_SaEast1 = 12
- AWSS3Region_CaCentral1 = 13

AWSS3StorageClass (v7.4.6 and later)

Used in the [ICICloudAWSS3GeneralParams](#) interface.

- AWSS3StorageClass_Standard = 0
- AWSS3StorageClass_StandardIA = 1
- AWSS3StorageClass_ReducedRedunancy = 2

AzureBlobAuthType (v7.4.6 and later)

Used in the [ICICloudAzureBlobGeneralParams](#) interface.

- AzureBlobAuthType_Anonymous = 0
- AzureBlobAuthType_SharedKey = 1
- AzureBlobAuthType_SASToken = 2 (added in 7.4.9)

AzureBlobMatchingFileNamePolicy (v7.4.6 and later)

Used in the [ICICloudAzureBlobUploadParams](#) and [ICICloudAzureBlobDownloadParams](#) interfaces.

- AzureBlobMatchingFileNamePolicy_Overwrite = 0
- AzureBlobMatchingFileNamePolicy_Skip = 1
- AzureBlobMatchingFileNamePolicy_Numerate = 2

CaptchaType (v7.4.2 and later)

Used in the [ICISite](#) interface **WorkspacesDropOffCaptchaType** property to specify the type of CAPTCHA to use.

- CaptchaType_None = 0,
- CaptchaType_Standard = 1,
- CaptchaType_reCAPTCHA = 2

CertificateFormat (v6.3 and later)

Used in the ICIServer Interface [CreateSSLCertificate](#) method to specify the certificate and private key format.

- PEM = 0; PEM (ASCII) encoded *.crt, *.key
- DER = 1; DER (binary public / private key) *.cer, *.der

- PKCS7 = 2; PKCS#7 P7B public key *.p7
- PKCS12 = 3; PKCS#12 (PFX combined) *.pfx

ChecksumAlgorithm (v8.0.5 and later)

Used in the [ICFileActionsChecksumParams](#) to specify the algorithm used.

- ChecksumCRC32 = 0,
- ChecksumMD5 = 1,
- ChecksumSHA1 = 2,
- ChecksumSHA256 = 3,
- ChecksumSHA384 = 4,
- ChecksumSHA512 = 5

CleanupTimestampType

- CleanupTimestampType_Created
- CleanupTimestampType_Modified
- CleanupTimestampType_CreatedAndModified

ClientDataField (added in v8)

Used in the ICISite interface [GetClientDataField](#) method.

- ClientDataFieldId_Comments = 13
- ClientDataFieldId_CompanyName = 2
- ClientDataFieldId_Custom1 = 10
- ClientDataFieldId_Custom2 = 11
- ClientDataFieldId_Custom3 = 12
- ClientDataFieldId_Description = 4
- ClientDataFieldId_Duns = 8
- ClientDataFieldId_Email = 0
- ClientDataFieldId_Fax = 6
- ClientDataFieldId_FullName = 1
- ClientDataFieldId_Mobile = 3
- ClientDataFieldId_Pager = 7
- ClientDataFieldId_PartnerId = 9
- ClientDataFieldId_Phone = 5

CloudProviderType (v7.4.6 and later)

Used in the [ICloudTransferActionParams](#), [ICIConnectionProfileParams](#), and [ICloudTransferActionParams](#) interfaces.

- CloudProviderType_AWSS3 = 0
- CloudProviderType_AzureBlob = 1
- CloudProviderType_AWSS3Compatible = 2
- CloudProviderType_AzureDataLakeStorageGen2

CloudStorageMonitorScanIntervalUnitType (v7.4.6 and later)

- COMSISecods = 0
- COMSIMinutes = 1
- COMSIHours = 2

CompressFormat (v7.2 and later)

Used in the [ICCompressActionParams](#) interface, [Format](#) property.

- AutoDetect = 0
- Bzip2 = 4
- Gzip = 3
- SevenZip = 2
- Tar = 5
- TarAndGzip = 6
- Zip = 1
- ZCompress = 7

CompressMethod (v7.2 and later)

Used in the [ICCompressActionParams](#) interface, [Method](#) property.

- BZIP_2 = 4
- COPY = 6
- DEFAULT = 0
- DEFLATE = 1
- DELTA = 5
- LZCCOMPRESS = 8
- LZMA = 3
- LZMA_2 = 7
- PPMd = 2

CompressOperation (v7.2 and later)

Used in the [ICCompressActionParams](#) interface, [Operation](#) property.

- Compress = 0
- Decompress = 1

CompressOverwritePolicy (v7.2 and later)

Used in the [ICCompressActionParams](#) interface, [OverwritePolicy](#) property.

- Never = 0
- Always = 1
- IfNewer = 2

ConditionsForConsent (v8 and later)

Used in the [ICIGDPRSettings Interface](#) ConditionsForConsent property.

- ConditionsForConsent_Unknown,
- ConditionsForConsent_EFT,
- ConditionsForConsent_External,
- ConditionsForConsent_Other,
- ConditionsForConsent_NotApplicable

ConditionOperator (v6.1 and later)

Used in the ICCompoundCondition Interface [Add](#) method, the ICEventRule Interface [AddIfStatement](#) method, and the ICISimpleCondition Interface [Operator](#) property

- Contains = 8
- Equals = 1
- ExistAt (added in v7.2) = 1024
- Greater (added in v6.3.1) = 256
- GreaterOrEquals (added in v6.3.1) = 512
- Less = 2
- LessOrEquals = 4
- Match = 16
- MemberOf = 32
- OneOf = 64
- StartsWith = 128

ConnectionProfileType (v7.4.6 and later)

Used in the [ICIconnectionProfileParams](#) interface.

- Protocol = 0
- Cloud = 1

DataSanitizationMethod (v6.3 and later)

Used in the ICISite Interface [DataSanitizationMethod](#) property to specify the data wiping technique.

- DataSanitizationNoWipe = 0; Windows standard – no wipe
- DataSanitizationPseudorandomData = 1; Pseudorandom data
- DataSanitization3PassDoD = 2; 3-pass DoD 5220.22-M

DiagLoggingER (7.4 and later)

Used in the ICIServer interface, DiagLoggingER is used in the Event Rule transfer log.

- DiagLoggingER_None = 0
- DiagLoggingER_FailuresOnly = 1

DirectCollectionType (v8 and later)

Used in the [ICIGDPRSettings](#) interface DirectCollection property.

- DirectCollection_Unknown,
- DirectCollection_EFT,
- DirectCollection_External,
- DirectCollection_Other,
- DirectCollection_NotApplicable

DPOAssignedType (v8 and later)

Used in the [ICIGDPRSettings](#) interface DPOAssignedType property.

- DPOAssigned_ConfiguredByEFT
- DPOAssigned_Other,
- DPOAssigned_NotApplicable

EmailLoginCredentialsMode (v6.3 and later)

Used in the ICISite Interface [EmailLoginCredentialsMode](#) property used to specify how to e-mail users their login credentials.

- EMailCredentialsBoth = 0; e-mail both username and password together
- EMailCredentialsSeparate = 1; e-mail the username and password in separate e-mails
- EMailCredentialsUsernameOnly = 2; e-mail the username only

EncryptedPDType (v8 and later)

Used in the [ICIGDPRSettings](#) interface EncryptedPDType property. (PD=Personal Data)

- EncryptedPD_NotEncrypted,
- EncryptedPD_Encrypted,
- EncryptedPD_CompensatingControl,
- EncryptedPD_NotApplicable

ERChangeLog (v7.2 and later)

ERChangeLog is used in the ICIServer interface in the EventRuleChangeLog property. The Event Rule Change Log must be enabled on the Server before changes will appear in the Change Log.

- ERChangeLog_Off = 0
- ERChangeLog_Optional = 1
- ERChangeLog_Mandatory = 2

EventActionType (v6.0 and later)

Used in the ICIEventAction Interface [type](#) property.

- AbortUserOperationAction (added in v8.0.6)
- AS2SendAction = 512 (added in v6.1)
- AWTTaskAction = 1024 (added in v6.1)
- BackupAction = 2048 (added in v6.1)
- CallSubroutineAction = 16777216 (&H1000000) (added in v8.0)
- CleanupAction = 128
- CloudUploadAction = 0x80000 (added in 7.4.4)
- CloudDownloadAction = 0x100000 (added in 7.4.4)
- CommandAction = 1
- CompressAction = 0x10000 (added v7.2)
- ContentIntegrityControl = 8192 (added v7.0.3)
- CsvExportAction = 0x10000000 (added v8.0.4)
- CsvImportAction = 0x8000000 (added v8.0.4)
- DefineVariableAction = 4194304 (&H400000 (added in v8)

- DownloadAction = 8
- FileAction = 0x8000 (added v7.2)
- FolderAction = 0x4000 (added v7.2)
- GetListingAction = 0x2000000 (added v8.0.4)
- LoopBreakAction = 0x4000000 (added v8.0.4)
- MailAction = 2
- PGPAAction = 32
- ReportAction = 256
- RunPowerShellScriptAction = 8388608 (&H800000) (added in v8)
- StopAction = 64
- SynchronizeAction = 0x10 (v8.0.5)
- UploadAction = 4
- UserAccountAction = 0x200000 (added v7.4.13)
- WebServiceAction = 0x40000 (added v.7.2)
- WindowsEventLog = 4096 (added in v6.3.0)

EventLogType (v6.3 and later)

Used in the [ICIWindowsEventLogActionParams](#) Interface [type](#) property to specify the event log type in the "Write to Windows Event Log" action.

- EventInformation = 0
- EventWarning = 1
- EventError = 2

EventProperty (v6.1 and later)

Used in the [ICICompoundCondition Interface](#) **Add** method, [ICIEventRule Interface](#) **AddIfStatement** method, and [ICISimpleCondition Interface](#).

- **AS2 Properties:**
 - AS2Direction = 6009; BOOL; AS2 Direction
 - AS2Host = 6005; STRING; AS2 Host
 - AS2ID = 6011; STRING; AS2 EFT ID
 - AS2MessageID = 6004; STRING; AS2 Message ID
 - AS2Payload = 6000; STRING; AS2 Payload
 - AS2MDN = 6001; STRING; AS2 MDN
 - AS2LocalMIC = 6002; STRING; AS2 Local MIC

- AS2RemoteMIC = 6003; STRING; AS2 Remote MIC
- AS2TransactionError = 6006; STRING; AS2 Transaction Error
- AS2TransactionResult = 6007; STRING; AS2 Transaction Result
- AS2TransactionVerbose = 6008; STRING; AS2 Transaction Verbose
- AS2PartnerID = 6010; STRING; AS2 Partner ID
- AS2TypeOfContent = 6012; INT; AS2 Content Type
- **Advanced Workflow Properties (7.4.7 and later):**
 - AWTTaskName = 12000
 - AWLogPath
 - AWEErrorCode
 - AWEErrorDescription
 - AWEErrorLine
 - AWResultCode
 - AWResultDescription
 - AWExecutionTimeMilliseconds
- **Context Variable properties (v7.2 and later):**
 - ContextVariable = 8000; STRING
- **Data Types:**
 - INT = Any signed 32-bit integer: -1, 100, -123123, 6547474
 - DWORD = Any unsigned 32-bit integer: 0, 1, 98765
 - STRING = Any string: "Hello", "world"
 - IP = IP or IP mask: "127.0.0.1", "192.168.*.*", "1-100.1-100.1-100.1-100"
 - PATH = Physical path: "d:\folder1\","e:\file.txt"
 - BOOL = Boolean value: VARIANT_TRUE, VARIANT_FALSE
 - VPATH = Virtual path: "/Usr/User1/", "/etc/tmp"
 - LIST = List of strings or paths: See STRING or PATH
 - ULONGLONG = Unsigned 64-bit integer: 123123123123
- **Client Properties:**
 - ClientLogin = 4000; LIST; Logon Name
 - ClientPassword = 4001; LIST; Logon Password
 - ClientAccessGroup = 4002; LIST; User Groups
 - ClientEnabled = 4003; BOOL; Account Enabled

- ClientSettingsLevel = 4004; INT; Settings Template (Value comes from the [ICIClientSettings.ID](#) property of the Settings Template.)
- ClientFullName = 4005; STRING; Full Name
- ClientDescription = 4006; STRING; Description
- ClientComment = 4007; STRING; Comment
- ClientEmail = 4008; STRING; Email Address
- ClientPhone = 4009; STRING; Phone Number
- ClientPager = 4010; STRING; Pager Number
- ClientFax = 4011; STRING; Fax Number
- ClientHomeFolder = 4012; VPATH; Home Folder
- ClientHomeFolderIsRoot = 4013; BOOL; Home folder is root
- ClientQuotaMax = 4014; INT; Quota Max
- ClientQuotaUsed = 4015; INT; Quota Used
- ClientInvalidLoginAttempts = 4016; DWORD; Invalid login attempts
- ClientCanChangePassword = 4017; BOOL; User can change password
- ClientIP = 4018; IP; Home IP
- ClientSSLAllowed = 4019; BOOL; User can connect using SSL
- ClientFTPAAllowed = 4020; BOOL; User can connect using FTP
- ClientSFTPAAllowed = 4021; BOOL; User can connect using SFTP
- ClientLastLogin = 4022; STRING; Last Login Date
- ClientPasswordExpiration = 4023; STRING; Password Expiration Date
- ClientMustResetPasswordAtFirstLogin = 4024; BOOL; User Must Change Password at Next Login
- ClientAccountExpirationDate = 4025; STRING; Account Expiration Date
- ClientAccountLocked = 4026; BOOL; Account Locked Out
- ClientCustomField1 = 4027; STRING; Custom Field 1
- ClientCustomField2 = 4028; STRING; Custom Field 2
- ClientCustomField3 = 4029; STRING; Custom Field 3
- ClientGDPREUSubjectStatus = 13000 (&H32C8) (added in v8; used in Event Rules)
- ClientGDPRPPConsentStatus = 13001 (&H32C9) (added in v8 used in Event Rules)
- ClientGDPRTOConsentStatus = 13002 (&H32CA) (added in v8; used in Event Rules)

- GDPRRightExercised = 13003 (&H32CB) (added in v8; used in Event Rules)
- GDPRRightExercisedArticleID = 13004 (&H32CC) (added in v8; used in Event Rules)
- GDPRReasonGiven = 13005 (&H32CD) (added in v8; used in Event Rules)
- **Connection Properties:**
 - LocalIP = 3000; IP; Local IP
 - RemoteIP = 3001; IP; Remote IP
 - LocalPort = 3002; DWORD; Local Port
 - RemotePort = 3003; DWORD; RemotePort
 - Protocol = 3004; INT; Protocol
 - WebTransferClientConnection = 3005; BOOL
 - HttpQueryString = 3006; STRING (v7.2 and later)
 - HttpHeadersList = 3007; STRING (v7.2 and later)
- **File System Properties:**
 - VirtualPath = 5000; LIST of VPATH; Virtual Path
 - PhysicalPath = 5001; LIST of PATH; Physical Path
 - DestinationVirtualPath = 5002; LIST of VPATH; Virtual Destination Path
 - DestinationPhysicalPath = 5003; LIST of PATH; Physical Destination Path
 - FolderName = 5004; LIST of PATH; Physical Folder Name
 - FileName = 5005; LIST of PATH; File Name
 - DestinationFolderName = 5006; LIST of PATH; Physical Destination Folder Name
 - DestinationFileName = 5007; LIST of PATH; Destination File Name
 - FileCRC = 5012; DWORD; File CRC
 - FileCreationDate = 5009; STRING; File Creation Date
 - FileCreationTime = 5010; STRING; File Creation Time
 - FileExtension = 5022 (&H139E) (added in v8)
 - FileSize = 5011; ULONGLONG; File Size
 - FolderOperation = 5008; STRING; File Change

Values:

- rename
- added
- removed

- ReportPath = 5013; STRING; Report File
- ReportContent = 5014; STRING; Report Content
- ReportFileName = 5015; STRING; Report File Name (v6.3.1 and later)
- VirtualFolderName = 5016; STRING; Virtual Folder Name (v6.3.1 and later)
- BaseFileName = 5017; STRING; The Base File Name is the portion of the filename to the left of the right-most period. The Base File Name was provided as a way to support rename. For example, if a file is downloaded as: SomeFile.ext.tmp, the Base File Name is: SomeFile.ext.
- FileFolderExists = 5018; BOOL; Whether file or folder exists. (v7.2 and later)
- CompressedPhysicalPath = 5019; STRING; Physical path of the file resulting from a compression action. (v7.2 and later)
- CompressedFileName = 5020; STRING; The filename of the file resulting from a compression action. (v7.2 and later)
- CompressedBaseFileName = 5021; STRING; The base filename (the portion of the filename to the left of the right-most period) of the file resulting from a compression action. (v7.2 and later)
- **General Properties:**
 - Time = 1; STRING; Event Time
 - Name = 2; STRING; Event Name
 - Reason = 3; INT; Event Reason
 - TimeStamp = 4; STRING; Event Time Stamp
 - DateStamp = 5; STRING; Event Date Stamp
 - MonitorFolderStatus = 6; STRING; Folder Monitor Health
 - EventName = 7; STRING; Event Full Name
 - TimeStampPrecise (v6.5 and later) = 8; STRING; Event Time Stamp, including milliseconds
 - FolderMonitorFailureReason (v6.5 and later) = 9; STRING; Folder Monitor Failure Reason
 - TransactionID = 10; STRING (v7.2 and later)
- **Server Properties:**
 - ServerRunning = 1000; BOOL; Server Running
 - ServerLogOldName = 1001; PATH; Old Log File Name
 - ServerLogNewName = 1002; PATH; New Log File Name
 - ServerLogOldPath = 1003; PATH; Old Log File Path
 - ServerLogNewPath = 1004; PATH; New Log File Path

- ServerLogType (v6.4 and later) = 1005; DWORD; Log File Format

Values:

- 2 = W3C Extended Log File Format
- 3 = Microsoft IIS Log File Format
- 4 = NCSA Common Log File Format
- 5 = No Log
- ServerLogFolder = 1006; PATH; Log Location
- ServerServerNodeName = 1007; STRING; Node Name
- **Site Properties:**
 - SiteRunning = 2000; BOOL; Site Running
 - SiteName = 2001; STRING; Site name
 - SiteAccountManagementURL = 2002; STRING; Account Management URL
 - SiteDMZAddress
 - SiteDMZPort
- **Transfer properties (v7.2.1 and later):**
 - TransferRate = 9000; ULONGLONG; Speed of transfer in Kbps (total bytes/total seconds)
 - TransferBytes = 9001; ULONGLONG; Number of bytes transferred.
 - TransferSeconds = 9002; ULONGLONG; Duration of transfer in seconds.
- **Workspace Properties**
 - WSName = 7002
 - WSOwner = 7004
 - WSOwnerEmail (v7.2.1) = 7009
 - WSParticipants = 7003
 - WSPhysicalPath = 7001
 - WSUserAccountExists = 7008
 - WSUserEmail = 7007
 - WSUserName = 7005
 - WSUserPermissions = 7006
 - WSUserRemovalReason = 7009 (removed in v7.1.0; refer to ICISite - Single-Site Interface:: [RemoveWorkspace](#) instead)
 - WSVirtualPath = 7000

EventReason (v6.3 and later)

EventReason allows you to specify the reason for a triggered event. It is a value type for Event Conditions (e.g., If event reason is "transfer failed due to file is of banned type," do this; otherwise, do that for On Upload Failed Event Rule). It can also be used in condition values:

- [ICIEventRule.AddIfStatement](#) - creating new IF statement with "If Event Reason is equal to XXX" condition: Event Property is "Reason," value is one of EventReason enum
- [ICICompoundCondition.Add](#) - (adding "If Event Reason is equal to XXX" condition to existing IF statement: Event Property is "Reason," value is one of EventReason enum
- [ICISimpleCondition.Value](#) - retrieving/modifying "XXX" in existing "If Event Reason is equal to XXX" condition: value is one of EventReason enum.
- **Reasons:**
 - ConnectFailedCertRejected (added v8)
 - ConnectFailedBannedIP = 11
 - ConnectFailedRestrictedIP = 10
 - ConnectFailedTooManyConnectionsPerIP = 9
 - ConnectFailedTooManyConnectionsPerSite = 8
 - DisconnectedByAdmin = 18
 - DisconnectedByConnection = 12
 - DisconnectedByError = 15
 - DisconnectedByIPBan = 16
 - DisconnectedByPasswordRetry = 17
 - DisconnectedByRequest = 13
 - DisconnectedByTimeout = 14
 - EmptyReason (added in v8)
 - FolderMonitorHealthFailed = 25
 - IPBannedByUserAccountAction (added in v8)
 - IPBannedDOS = 28
 - IPBannedDOSTemporary (added in v8)
 - IPBannedInvalidPassword = 29
 - IPBannedInvalidUsername = 30
 - IPBannedTooManyInvalidCommands (added in v8)
 - LoginFailedAccountLocked = 27 (the "d" is not a typo)
 - LoginFailedAccountDisabled = 2

- LoginFailedPasswordNotAccepted = 1 (in v6.3.0.57, the "t" in "Accepted" is missing)
- LoginFailedProtocolNotSupported = 6
- LoginFailedRestrictedIP = 7
- LoginFailedTooManyConnectionPerUser = 4
- LoginFailedTooManyConnectionsPerIP = 5
- LoginFailedTooManyConnectionsPerSite = 3
- TransferFailedAborted = 21
- TransferFailedAccessDenied = 20
- TransferFailedBannedFile = 19
- TransferFailedConnectionClosed = 22
- TransferFailedFileNotFound = 23
- TransferFailedQuotaExceeded = 24
- TransferFailedXCRCCheckFailed = 26

EventRuleSchedulingAlgorithm (v7.0 and v7.0.1)

- RoundRobinEventRuleSchedulingAlgorithm = 1

EventRuleStatementType (v6.0 and later)

Used in the [ICIActionStatement Interface](#) type property, the [ICIfStatement Interface](#) type property, and the [ICILoopDatasetStatement](#) type property.

- ActionStatement = 0
- IfStatement = 1
- LoopDatasetStatement = 2 (added in EFT v8.0.4)

EventType (v6.0 and later)

Used to specify the type of event trigger ICISite Interface [EventRules](#) method and the [ICIEventInfo Interface](#) and [ICIEventRuleStatement Interface](#) type properties.

- BeforeFileDownload = 20483 (&H5003)
- BeforeWorkspaceDeleted = 28675 (&H7003) (EFT v7.1- v8.0.5; renamed to OnWorkspaceBeforeDelete)
- EventRuleSubroutine = 36865 (&H9001) (EFT v8 and later)
- MonitorFolder = 4101 (&H1005)
- OnAS2InboundTransactionSucceeded = 24577 (&H6001) (v6.3.0 and later; "On" added in 6.3.) = 24577; AS2 Inbound Transaction Succeeded

- OnAS2InboundTransactionFailed = 24578 (&H6002) (v6.3.0 and later; "On" added in 6.3.1) = 24578; AS2 Inbound Transaction Failed
- OnAS2OutboundTransactionSucceeded = 24579 (&H6003) (v6.3.0 and later; "On" added in 6.3.1) = 24579; AS2 Outbound Transaction Succeeded
- OnAS2OutboundTransactionFailed = 24580 (&H6004) (v6.3.0 and later; "On" added in 6.3.1) = 24580; AS2 Outbound Transaction Failed
- OnChangeFolder = 20491 (&H500B) Folder Changed
- OnClientConnected = 12289 (&H3001) = 12289; User Connected
- OnClientConnectionFailed = 12290; User Connect Failed
- OnClientCreated = 16391 (&H4007); New User Created
- OnClientDeleted = 16393 (&H4009) (v6.4.5 and later) = 16393; User Account Deleted
- OnClientDisabled = 16385 (&H4001) = 16385; User Account Disabled
- Const OnClientDisconnected = 12291 (&H3003); User Disconnected
- OnClientEnabled = 16394 (&H400A) (v6.4.5 and later) = 16394; User Account Enabled
- OnClientLocked = 16392 (&H4008) (v6.1.0 and later) = 16392; User Account Locked
- OnClientLoggedIn = 16388 (&H4004); User Logged in
- OnClientLoggedOut = 16387 (&H4003); User Logged out
- OnClientLoginFailed = 16389 (&H4005); User Login Failed
- OnClientPasswordChanged = 16390 (&H4006); User Password Changed
- OnClientQuotaExceeded = 16386 (&H4002); User Quota Exceeded
- OnCloudStorageMonitor = (&H8001) (added v7.4.6)
- OnDownloadFailed = 20490 (&H500A) ; Download Failed
- OnFileDeleted = 20481 (&H5001); File Deleted
- OnFileDownload = 20484 (&H5004); File Downloaded
- OnFileMoved = 20492 (&H500C); File Moved
- OnFileRenamed = 20485 (&H5005); File Renamed
- OnFileUpload = 20482 (&H5002); File Uploaded
- OnFolderCreated = 20486 (&H5006); Folder Created
- OnFolderDeleted = 20487 (&H5007); Folder Deleted
- OnGDPRRightExercised (added in v8)
- OnIPAddedToBanList = 8195 (&H2003) (v6.3.0 and later); IP Added to Ban List
- OnLogRotate = 4098 (&H1002); Log Rotated
- OnMonitorFolderFailed = 4102 (&H1006); Folder Monitor Failed

- OnSecureMessageAttachmentAfterDownload
- OnSecureMessageAttachmentBeforeDownload
- OnSecureMessageComposed
- OnSecureMessageNotSent
- OnSecureMessageSent
- OnSecureMessageViewed
- OnServiceStarted = 4100 (&H1004); Service Started
- OnServiceStopped = 4099 (&H1003); Service Stopped
- OnSiteStarted = 8193 (&H2001); Site Started
- OnSiteStopped = 8194 (&H2002); Site Stopped
- OnTimer = 4097 (&H1001); Scheduler (Timer) Event
- OnUploadFailed = 20489 (&H5009); Upload Failed
- OnVerifiedUploadFailure = 20494 (&H500E); Verified Upload Failed
- OnVerifiedUploadSuccess = 20493 (&H500D); Verified Upload Succeeded
- OnVerifiedDownloadFailure = 20496 (&H5010); Verified Download Failed
- OnVerifiedDownloadSuccess = 20495 (&H500F); Verified Download Succeeded
- OnWorkspaceBeforeDelete (added v8.0.6; formerly BeforeWorkspaceDeleted)
- OnWorkspaceCreated = 28673 (&H7001) (added v7.1)
- OnWorkspaceDeleted = 28674 (&H7002) (added v7.1)
- OnWorkspaceExpired
- OnWorkspaceUserInvited = 28676 (&H7004) (added v7.1)
- OnWorkspaceUserJoins = 28677 (&H7005) (added v7.1)
- OnWorkspaceUserRemoved = 28678 (&H7006) (added v7.1)
- RESTInvocation = 0x9002 (added v8.0.5)

FileDateFormat (added v8.0.4)

Used in [ICIGetListingActionParamsFileDateFormat](#) property.

- FileDateFormat_Default,
- FileDateFormat_Epoch,
- FileDateFormat_ISO8601

FileEncodingType (v7.2 and later)

Used in the [ICIFileActionParams](#) Interface [Encoding](#) property.

- ANSI = 0 (v7.2 - 8.0.4.x)
- UTF8 = 1 (v7.2 - 8.0.4.x)
- Unicode = 2 (v7.2 - 8.0.4.x)
- UnicodeBigEndian = 3 (v7.2 - 8.0.4.x)
- EncodingANSI = 0 (v8.0.5 and later)
- EncodingUTF8 = 1 (v8.0.5 and later)
- EncodingUnicode = 2 (v8.0.5 and later)
- EncodingUnicodeBigEndian = 3 (v8.0.5 and later)

FileOperationType (v7.2 and later)

Used in the [ICIFileActionParams](#) Interface [Operation](#) property.

- FileRead=0 (v8.0.5 and later)
- FileWrite=1 (v8.0.5 and later)
- FileRename=2 (v8.0.5 and later)
- FileDelete=3 (v8.0.5 and later)
- FileConcatenate=4 (v8.0.5 and later)
- FileChecksum = 5 (v8.0.5 and later)
- FileCreate = 0 (v7.2 - v8.0.4)
- FileRename = 1 (v7.2 - v8.0.4)
- FileDelete = 2 (v7.2 - v8.0.4)

FileOverwriteOptions (v7.2 and later)

Used in the [ICIFileActionParams](#) Interface [OverwriteOption](#) property.

- Overwrite = 0
- Numerate = 1
- Skip = 2

FolderMonitorPollIntervalType (v6.5 and later)

Used to specify Folder Monitor polling by seconds, minutes, or hours.

- FMPISeconds = 0; use seconds
- FMPIMinutes = 1; use minutes
- FMPIHours = 2; use hours

FolderOperationType (v7.2 and later)

Used in the [ICIFolderActionParams](#) Interface [Operation](#) property.

- FolderCreate = 0
- FolderRename = 1
- FolderDelete = 2

FTPDataConnectionMode (v6.2.18 and later)

Used in the ICITransferActionParams Interface [FTPDataConnectionMode](#) property used to specify the mode of FTP data connection.

- FTPMode_Auto = 2
- FTPMode_EPRT (EFT v6.4 and later) = 3
- FTPMode_EPSV (EFT v6.4 and later) = 4
- FTPMode_PASV = 1
- FTPMode_PORT = 0

GDPRPrivacyPolicyConsentStatus (added in v8)

Used in ICIClientSettings::[Get/SetGDPRPrivacyPolicyConsentStatus](#) method.

- PPConsent_Unknown,
- PPConsent_Granted_implicit,
- PPConsent_Granted_explicit,
- PPConsent_Denied,
- PPConsent_Rescinded,

GDPRPrivacyPolicyMode (added in v8)

Used in ICIPrivacyPolicy::[Mode](#) property.

- GDPRPrivacyPolicyMode_Disabled,
- GDPRPrivacyPolicyMode_Shown,
- GDPRPrivacyPolicyMode_ConsentRequired

GDPRTermsOfServiceConsentStatus (added in v8)

Used in [ICIClientSettings](#)::GetGDPRTermsOfServiceConsentStatus.

- ToSConsent_Unknown,
- ToSConsent_Agreed_implicit,
- ToSConsent_Agreed_explicit,
- ToSConsent_Disagreed,
- ToSConsent_Withdrawn

GDPRTermsOfServiceMode (added in v8)

Used in `ICITermsOfService::Mode` property.

- `GDPRTermsOfServiceMode_Disabled`,
- `GDPRTermsOfServiceMode_Implied`,
- `GDPRTermsOfServiceMode_Mandatory`

GDPRSubjectStatus (added in v8)

Used in `ICIClientSettings::Get/SetGDPRSubjectStatus` method.

- `Subject_Unknown`,
- `Subject_Yes`,
- `Subject_No`,

HADeploymentScenario

- `HAAwsSnsSqsDeploymentScenario = 1`
- `HAMsmqDeploymentScenario = 0`

IcapErrorsPolicy (v8.0.5 and later)

Used in the `ICContentIntegrityControlProfile` interface.

- `Fail = 0`
- `Continue = 1`

IcapMaxScanSizeUnits (v7.0.3 and later)

Used in the `ICContentIntegrityControlProfile` interface

- `ScanSizeBytes = 0`
- `ScanSizeKBytes = 1`
- `ScanSizeMBytes = 2`
- `ScanSizeGBytes = 3`

ICAPMode (v7.0.3 and later)

Used in the `ICContentIntegrityControlProfile`.

- `REQMOD`
- `RESPMOD`

IndirectCollectionType (v8 and later)

Used in the `ICIGDPRSettings Interface` `IndirectCollection` property.

- IndirectCollection_Unknown,
- IndirectCollection_EFT,
- IndirectCollection_External,
- IndirectCollection_Other,
- IndirectCollection_NotApplicable

IPAccessRuleType (v6.3 and later)

[Server.GetIPAccessRules](#), [Site.GetIPAccessRules](#), and [Client.GetIPAccessRules](#) return the array of `ICIIPAccessRule` (which can be either Manual or Autoban); `ICIAutobanIPAccessRule.BannedIPs` is the array of `ICIBannedIP`. All are read-only properties; to modify IP Access, use the corresponding methods of Server, Site, or Client (Move, Remove, Unban, etc).

- AutobanIPRule = 0
- ManualIPRule = 1

Example (VBScript):

```
rules = oServer.GetIPAccessRules()
For Each key In rules
If key.type = 0 Then
WScript.echo "AutoBan"
End if
If key.type = 1 Then
WScript.echo "Manual " + key.address
End if
Next
```

LawfulnessOfProcessingType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) `LawfulnessOfProcessingType` property.

- LawfulnessOfProcessing_Unknown,
- LawfulnessOfProcessing_DataSubjectConsent,
- LawfulnessOfProcessing_Contractual
- LawfulnessOfProcessing_Other,
- LawfulnessOfProcessing_NoPD,
- LawfulnessOfProcessing_NotApplicable

LDAPSearchScope (v6.3 and later)

Used in the `ICILDAPAuthManagerSettings Interface` [SearchScope](#) property.

- Scope_Base = 0
- Scope_OneLevel = 1
- Scope_Subtree = 2

LDAPVersion (v6.3 and later)

Used in the ICILDAPAuthManagerSettings Interface [LDAPVersion](#) property.

- LDAP_Version2 = 2
- LDAP_Version3 = 3

LimitedAccessType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) LimitedAccessType property.

- LimitedAccess_ConfiguredByEFT
- LimitedAccess_CompensatingControl,
- LimitedAccess_NotApplicable

LogicalOperator (v6.1 and later)

Used in the ICICompoundCondition Interface [Operator](#) property.

- LogicalOr = 0
- LogicalAnd = 1

LogMode (v6.3 and later)

Used in the ICIServer Interface [LogMode](#) property to specify the logging mode.

- LogModeStandard = 0
- LogModeVerbose = 1

LoopOrderType (v8.0.4 and later)

Used in the ICILoopDatasetStatementParams interface to specify loop order in ICILoopDatasetStatementParams LoopOrder property.

- LoopOrderType_Original,
- LoopOrderType_Ascending,
- LoopOrderType_Descending

MaterialScopeType (v8 and later)

Used in the [ICIGDPRSettings Interface](#).

- MaterialScope_Unknown = 0
- MaterialScope_InScope = 1
- MaterialScope_NotInScope = 2

MatchingFileOption (v8.0.5 and later)

- MatchingFileAppend = 0,
- MatchingFileSkip = 1,
- MatchingFileNumerate = 2,
- MatchingFileOverwrite = 3,
- MatchingFileFail = 4

MFAType (v8.0.4 and later)

Used in the [WorkspacesMfaType](#) property.

- MfaType-email
- MfaType_sms
- MfaTpe_eitiher

Month (v6.3 and later)

Used in the ICITimerEventRuleParams Interface [YearlyMonth](#) property.

- Month_January = 0
- Month_February = 1
- Month_March = 2
- Month_April = 3
- Month_May = 4
- Month_June = 5
- Month_July = 6
- Month_August = 7
- Month_September = 8
- Month_October = 9
- Month_November = 10
- Month_December = 11

NecessityType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) Necessity property.

- Necessity_ConfiguredByEFT
- Necessity_Other,
- Necessity_NotApplicable

NetworkProtocol (v6.1 and later)

Used in the ICITransferActionParams Interface [Protocol](#) property.

- ProtocolLocal = -1
- ProtocolFTP = 0
- ProtocolFTPS = 1
- ProtocolFTPSExpl = 2
- ProtocolSFTP = 3
- ProtocolHTTP = 4
- ProtocolHTTPS = 5
- ProtocolFTPSAuthTLS = 8

PGPCompressionLevel (v6.1 and later)

Used to specify the compression level for a PGP encryption operation in the [ICIPgpActionParams InterfaceCompressionLevel](#) property.

- NoCompression = 0
- CompressionLevel1 = 1
- CompressionLevel2 = 2
- CompressionLevel3 = 3
- CompressionLevel4 = 4
- CompressionLevel5 = 5
- CompressionLevel6 = 6
- CompressionLevel7 = 7
- CompressionLevel8 = 8
- CompressionLevel9 = 9

PGPKeyCipher (v6.3 and later)

Used in the ICIServer Interface [CreatePGPKey](#) method (prior to v8) and ICISite Interface [CreatePGPKey](#) method (added in v8) to specify the PGP key cipher.

- PGPKeyCipher_IDEA = 0
- PGPKeyCipher_3DES = 1
- PGPKeyCipher_CAST5 = 2

- PGPCipher_AES128 = 3
- PGPCipher_AES192 = 4
- PGPCipher_AES256 = 5
- PGPCipher_TWOFISH = 6

PGPKeyType (v6.3 and later)

Used in the ICIServer Interface [CreatePGPKey](#) method (prior to v8) and ICISite Interface [CreatePGPKey](#) method (added in v8) to specify the PGP key type.

- PGPKeyType_DSS = 0
- PGPKeyType_RSA = 1
- PGPKeyType_RSALegacy = 2

PGPLogLevel (v6.3 and later)

Used in the ICISite Interface [PGPLogLevel](#) property to specify the PGP module log level.

- PGPLogLevelStandard = 0
- PGPLogLevelVerbose = 1
- PGPLogLevelDebug (added in v8)

PGPOperation (v6.0 and later)

Used to specify the PGP operation to perform on a file in the ICIPgpActionParams Interface [Operation](#) property.

- Encrypt = 0
- EncryptAndSign = 1 (added in v6.1)
- Sign = 2 (added in v6.1)
- SDA = 3 (added in v6.1)
- Decrypt = 4 (added in v6.1)
- DecryptAndVerify = 5 (added in v6.3)
- VerifyOnly = 6 (added in v6.3)

PGPSigningHash (v6.1 and later)

Used to specify the hash algorithm for a PGP signing operation in the ICIPgpActionParams Interface [SigningHash](#) property.

- MD5Hash = 0
- SHA1Hash = 1
- RIPEMD160Hash = 2

- SHA256Hash = 3
- SHA384Hash = 4
- SHA512Hash = 5

PredefinedReportPeriod (v6.1 and later)

Used to specify one of the predefined reporting periods in the ICIReportActionParams Interface [CustomDate](#) property.

- AllDates = 0
- MonthToDate = 1
- QuarterToDate = 2
- YearToDate = 3
- CurrentWeek = 4
- CurrentMonth = 5
- CurrentQuarter = 6
- CurrentYear = 7
- LastWeek = 8
- LastMonth = 9
- LastQuarter = 10
- LastYear = 11
- Last30Days = 12
- Last12Months = 13
- Today = 14 (added in v6.2.3)
- Yesterday = 15 (added in v6.2.3)
- Last24Hours = 16 (added in v6.2.3)

PrePostTransferCommand (v7.1.1 and later)

Used in the [TransferActionParams](#) interface.

- None = 0
- CreateFolder = 1
- RemoveFolder = 2
- RenameFile = 3
- DelFile = 4
- MainframeSupport = 5

ProcessingPrinciplesType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) ProcessingPrinciples property.

- ProcessingPrinciples_Unknown,
- ProcessingPrinciples_NoMet,
- ProcessingPrinciples_Met,
- ProcessingPrinciples_NotInScope

ProxyAuthenticationType (v6.2.31 and later)

Used in the ICITransferActionParams Interface [ProxyAuthenticationType](#) property used to specify the proxy authentication type.

- ProxyAuthUserAtSite = 0
- ProxyAuthSite = 1
- ProxyAuthUserWithLogon = 2
- ProxyAuthUserPassAcct = 3
- ProxyAuthOpenSite = 4
- ProxyAuthCustom = 5

ProxyType (v6.2.31 and later)

Used in the [ICITransferActionParams Interface](#), [ICIDownloadActionParams Interface](#), and [ICIUploadActionParams Interface](#) **ProxyType** property used to specify the type of Proxy.

- ProxyTypeFTP = 1
- ProxyTypeHTTP = 2
- ProxyTypeDMZ = 3

PurposeType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) Purpose property.

- Purpose_ConfiguredByEFT
- Purpose_Other,
- Purpose_NotApplicable

Recurrence (v6.3 and later)

Used in the ICITimerEventRuleParams Interface [Recurrence](#) property.

- Recurrence_Continually = 0
- Recurrence_Daily = 1
- Recurrence_Weekly = 2
- Recurrence_Monthly = 3
- Recurrence_Yearly = 4

- Recurrence_Once = 5
- Recurrence_Calendar = 6

RegistrationState (v6.1 and later)

Used in the ICIServer Interface [ModuleRegistrationState](#) method to retrieve EFT module registration status. Used with the [ServerModule](#) enumerators.

- Registered = 0
- Evaluation = 1
- Expired = 2

Relation (v6.3 and later)

Used in the [ICITimerEventRuleParams Interface](#) **MonthlyRelativeWeekday** and **YearlyRelativeWeekday** properties to specify the relation of one weekday to the month or year (e.g., first weekday of the month, last weekday of the year).

- Relation_First = 0; first weekday
- Relation_Second = 1; second weekday
- Relation_Third = 2; third weekday
- Relation_Fourth = 3; fourth weekday
- Relation_Last = 4; last weekday

RepeatPattern (v6.3 and later)

Used in the ICITimerEventRuleParams Interface [RepeatPattern](#) property to specify a pattern for Timer Event Rules (e.g., repeats every hour, repeats every 30 minutes, etc.).

- RepeatHours = 0
- RepeatMinutes = 1
- RepeatSeconds = 2

ReportFilterOperator (v6.1 and later)

Used to specify a report filter operator in the [ICIReportActionParams Interface](#) **FilterOperator1** and **FilterOperator2** properties.

- NoFilter = -1
- EqualFilter = 0
- NotEqualFilter = 1
- LessFilter = 2
- LessOrEqualFilter = 3
- GreaterFilter = 4

- GreaterOrEqualFilter = 5
- ContainsFilter = 6
- StartsWithFilter = 7

ReportFiltersCombiningOperator (v6.1 and later)

Used to specify the way of combining two report filters in the ICIReportActionParams Interface [FilterAndOr](#) property.

- NoCombining = -1
- AndCombining = 0
- OrCombining = 1

ReportFormat (v6.1 and later)

Used to specify the report format in the ICIReportActionParams Interface [ReportFileFormat](#) property.

- HTMLFormat = 0
- PDFFormat = 1
- VPFormat = 2

ReportPeriodType (v6.1 and later)

Used to specify a type of reporting period (range or predefined) in the ICIReportActionParams Interface [DateFormat](#) property.

- Predefined = 0
- Range = 1

RESTInvocationAuthType (v8.0.5 an later)

Used in the [ICIRESTInvocationAuthParams](#) interface.

- RIAT_None = 0,
- RIAT_PGsClient = 1,
- RIAT_TokenSecret = 2,
- RIAT_PowerShellScript = 3

RESTInvocationVerifyType (v8.0.5 and later)

Used in the [ICIRESTInvocationVerifyParams](#) interface.

- RIVT_None = 0,
- RIVT_PowerShellScript = 1

RightStateType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) RightOfAccess, RightToBeForgotten, RightToExport, RightToObject, RightToRectify, and RightToRestrict properties.

- RightState_Unknown,
- RightState_ViaWTC,
- RightState_UponRequest,
- RightState_Other,
- RightState_NotApplicable

RightToBeNotifiedStateType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) RightToBeNotified property.

- RightToBeNotifiedState_Unknown,
- RightToBeNotifiedState_ViaEventRules,
- RightToBeNotifiedState_UponRequest,
- RightToBeNotifiedState_Other,
- RightToBeNotifiedState_NotApplicable

RiskAssessmentType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) RiskAssessment property.

- RiskAssessment_EFTGeneratedDPIAReport,
- RiskAssessment_Other,
- RiskAssessment_NotApplicable

RiskMitigationType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) RiskMitigation property.

- RiskMitigation_ConfiguredByEFT
- RiskMitigation_Other,
- RiskMitigation_NotApplicable

SecretsModuleType (v8.0.1 and later)

- SecretsModuleType_LocalPasswordsStorage,
- SecretsModuleType_AzureKeyVault,
- SecretsModuleType_AzureKeyVaultManagedIdentities

ServerModule (v6.1 and later)

Used in the ICIServer Interface [ModuleRegistrationState](#) method to retrieve an EFT module state. Used with the [RegistrationState](#) enumerator.

- Const **EFTServer** = 0
- Const **SSHModule** = 1
- Const **HTTPModule** = 2
- Const **WTCModule** = 3
- Const **ARModule** = 4
- Const **PGPModule** = 5
- Const **HSMModule** = 6 (In v7.4.7 and later, this is for the ASM (in Enterprise) or the ESM (in Express))
- Const **AS2Module** = 7
- Const **AWModule** = 8
- Const **DMZGWModule** = 9
- Const **FolderMonitorEventModule** = 10 (v7.2.2 and later)
- Const **TimerEventModule** = 11 (v7.2.2 and later)
- Const **FileTransferClientModule** = 12 (v7.2.2 and later)
- Const **WorkspacesModule** = 13 (v7.2.3 and later)
- Const **AccelerateModule** = 14 (v7.3 and later)
- Const **AAMModule** = 15 (v7.3.3 and later)
- Const **MTCModule** = 16 (v7.3.9 and v7.4.7 and later)
- Const **CICModule** = 17 (v7.3.9 and v7.4.7 and later)
- Const **CCMModule** = 18 (v7.3.9 and v7.4.7 and later)
- Const **EnterpriseActionsModule** (v8.0.7 and later)
- Const **FTPModule** (v8.0.7 and later)
- Const **RemoteAgentModule** (v8.0.7 and later)
- Const **SecureFormsModule** (v8.0.7 and later)

ServerNodeSiteState (v7.2.1 and later)

Used for the [ICIServerNodeSiteInfo](#) Interface **State** property to specify state of cluster node.

- SiteIsRunning = 0
- SiteIsStopped = 1
- SiteFailedToStart = 2

SFTPAdvBool (v6.0 and later)

Used in numerous properties and methods. [Difference between VARIANT_BOOL and SFTPAdvBool](#).

- abFalse = 0
- abTrue = 1
- abInherited = -2

SizeUnits (v7.4.2 and later)

Used in the ICISite interface [SetWorkspacesDropOffMaxMessageSize](#) method.

- SizeUnits_MByte = 0,
- SizeUnits_GByte = 1

SOCKSType (v6.2.31 and later)

Used in the [ICIDownloadActionParams Interface Properties](#)[ICITransferActionParams Interface](#), and [ICIUploadActionParams Interface](#) **SOCKSType** property to specify the SOCKS proxy type for a transfer action.

- SOCKSNone = -1
- SOCKS4 = 0
- SOCKS5 = 1

StopType (v6.1 and later)

Used in the [ICIStopActionParams Interface](#) **Action** property.

- StopRule = 2
- StopEvent = 4
- StopRuleAndEvent = 6

SuppressForgotPasswordMode (v7.4.11 and later)

Allows you to suppress the ability for users to reset their password for all domains, only internal domains, or only external domains.

- SuppressForgotPasswordMode_AllDomain,
- SuppressForgotPasswordMode_OnlyInternalDomain,
- SuppressForgotPasswordMode_OnlyExternalDomain

SyncDirectionType (v8.0.5 and later)

Used in the [ICISynchronizeActionParams](#) interface to specify whether to mirror locally or remotely.

- SyncDT_MirrorRemote,
- SyncDT_MirrorLocal

SyncOverwriteType (v8.0.5 and later)

Used in the [ICISynchronizeActionParams](#) interface to specify whether to overwrite, skip by file size, or skip by file size and time.

- SyncOT_UseOverwrite,
- SyncOT_SkipByFileSize,
- SyncOT_SkipByFileSizeAndTime

TerritorialScopeType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) TerritorialScope property.

- TerritorialScope_Unknown,
- TerritorialScope_InUnion,
- TerritorialScope_NotInUnion,
- TerritorialScope_InScope,
- TerritorialScope_NotInScope

TimeUnits (v7.3.3 and later)

The TimeUnits enum is used in the ICISite interface [Get/SetWorkspacesMacDuration](#) and [Get/SetWorkspacesMacOAIDuration](#) methods.

- TimeUnits_Minute = 0,
- TimeUnits_Hour = 1,
- TimeUnits_Day = 2,
- TimeUnits_Week = 3,
- TimeUnits_Month = 4,
- TimeUnits_Year = 5

TransferEncodingType (v6.5 and later)

Allows you to specify transfer encoding mode. Used in the ICISite interface [SFTPEncoding](#) and [FTPEncoding](#) properties, and the ICITransferActionParams interface [TransferEncoding](#) property.

- TransferEncodingAutoDetect = 0; Automatically detect encoding
- TransferEncodingUTF8 = 1; Use UTF-8 encoding
- TransferEncodingASCII = 2; Use ASCII encoding

TransferOverwriteType (v6.2.31 and later)

Used in the [ICIDownloadActionParams Interface](#), [ICITransferActionParams Interface](#), and [ICIUploadActionParams Interface](#) **OverwriteType** property used to specify how to replace files that have the same names when transferring files.

- TransferOTOverwrite = 0; Always overwrite
- TransferOTSkip = 1; Never overwrite
- TransferOTSmart = 2; Check CRC; if CRCs are equal, skip, otherwise, overwrite
- TransferOTNumerate = 3; Create new file with same name plus a number

TransferSafeguardsType (v8)

Used in the [ICIGDPRSettings Interface](#) TransferSafeguards property.

- TransferSafeguards_ConfiguredByEFT,
- TransferSafeguards_Other,
- TransferSafeguards_NotApplicable

TransparentInformationType (v8 and later)

Used in the [ICIGDPRSettings Interface](#) TransparentInformation property.

- TransparentInformation_Unknown,
- TransparentInformation_EFT,
- TransparentInformation_External,
- TransparentInformation_Other,
- TransparentInformation_NotApplicable

TwoFactorAuthenticationMode (v6.3 and later)

Used in the [ICISite Interface](#) TwoFactorAuthentication property to specify special authentication mode for Site users.

- TwoFactorAuth_None = 0
- TwoFactorAuth_RADIUS = 1
- TwoFactorAuth_RSA = 2

UploadFormComType (v8.0.4 and later)

Used in the ICIUploadForm interface [type](#) property to define when to show the Upload Form, for each file upload or once for batch upload.

- UploadFormComType_File,
- UploadFormComType_Batch

UploadFormElementComType (v8.0.4 and later)

Used in the ICIUploadFormElement interface type property to define the form element type

- UploadFormElementComType_String,
- UploadFormElementComType_Multistring,
- UploadFormElementComType_Dropdown,
- UploadFormElementComType_Radio,
- UploadFormElementComType_Toggle

UserAccountActionType (7.4.13 and later)

Used in [ICIUserAccountActionParams](#) to specify property action.

- BanAccountIPPermanently
- BanAccountIPTemporarily
- DeleteAccount
- DeleteAccountRetainVFS
- DisableAccount
- EnableAccount = 7 (added in v8);
- KickUser
- LockAccountForNMinutes
- PersonalDataDeleteFields = 8 (added in v8)
- PersonalDataResetToDefaults = 12 (added in v8)
- PersonalDataSetEnabled = 9 (added in v8)
- PersonalDataSetModifiable = 10 (added in v8)
- PersonalDataSetVisible = 11 (added in v8)
- SetEUDataSubjectStatus = 15 (added in v8)
- SetPrivacyPolicyConsent = 13 (added in v8)
- SetTermsOfServiceConsent = 14 (added in v8)

VirtualFolderType (8.0.6 and later)

- VirtualFolderType_None = 0,
- VirtualFolderType_PhysicalFolder = 1,
- VirtualFolderType_AWSS3Storage = 2,
- VirtualFolderType_AzureStorage = 3,
- VirtualFolderType_ConnectionProfile = 4

WebServiceVerbType (v7.2 and later)

Used in the [ICIWebServiceActionParams](#) interface [Verb](#) property.

- VerbGet = 0
- VerbPost = 1
- VerbPut = 2
- VerbDel = 3

WebSSONameOrAttribute (v7.3.3 and later)

- WebSSONameId = 0,
- WebSSOAttribute = 1

WebSSONameOrAttributeIdentifierFormat (v7.3.3 and later)

- Unspecified = 0
- EmailAddress = 1
- X509SubjectName = 2
- WindowsDomainQualifiedName = 3
- KerberosPrincipalName = 4
- EntityIdentifier = 5
- URIRelativeReference = 6
- Basic = 7

Weekday (v6.3 and later)

Used in the [ICITimerEventRuleParams Interface](#) [MonthlyWeekday](#) and [YearlyWeekday](#) properties to specify the day of the week.

- Weekday_Sunday = 0
- Weekday_Monday = 1
- Weekday_Tuesday = 2
- Weekday_Wednesday = 3
- Weekday_Thursday = 4
- Weekday_Friday = 5
- Weekday_Saturday = 6
- Weekday_Day = 7
- Weekday_Weekday = 8
- Weekday_Weekendday = 9

Weekdays (v6.1 - 6.2.31; replaced with Weekday, above)

Used in the [ICITimerEventRuleParams Interface](#) **MonthlyWeekday** and **YearlyWeekday** properties to specify the day of the week.

- Sunday = 0
- Monday = 1
- Tuesday = 2
- Wednesday = 3
- Thursday = 4
- Friday = 5
- Saturday = 6

WorkspacesOAIPasscodeProtectionType (v8.0.2)

- WorkspacesOAIPasscodeProtection_Required = 0
- WorkspacesOAIPasscodeProtection_NotRequired = 1
- WorkspacesOAIPasscodeProtection_SenderChooses = 2

WorkspacesOAISharingModeType (v7.3.3 and later)

This enum is used in the ICISite interface [WorkspacesOAISharingMode](#) property.

- WorkspacesOAISharingMode_Users = 0,
- WorkspacesOAISharingMode_UsersGuests = 1,
- WorkspacesOAISharingMode_UsersGuestsAnonymous = 2

WorkspacesParticipationlessAccountPolicy (v7.4.6 and later)

Used in the ICISite interface [WorkspacesGAParticipationlessAccountPolicy](#) property.

- Const WorkspacesParticipationlessAccountPolicy_Disable = 0
- Const WorkspacesParticipationlessAccountPolicy_DeleteAccountOnly = 1
- Const WorkspacesParticipationlessAccountPolicy_DeleteAccountAndHF = 2

WorkspacesSecureMessageBodyPolicy (v7.4.6 and later)

- WorkspacesSecureMessageBodyPolicy_AlwaysSecure = 0
- WorkspacesSecureMessageBodyPolicy_NeverSecure = 1
- WorkspacesSecureMessageBodyPolicy_UserChoice = 2