DCOM Connector Logon Component (BC-FES-AIT)

HELP.BCFESLOG

Release 4.6C

Copyright

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax
	Tip

Contents

DCOM Connector Logon Component (BC-FES-AIT)

DCOM Connector Logon Component (BC-FES-AIT)

Purpose

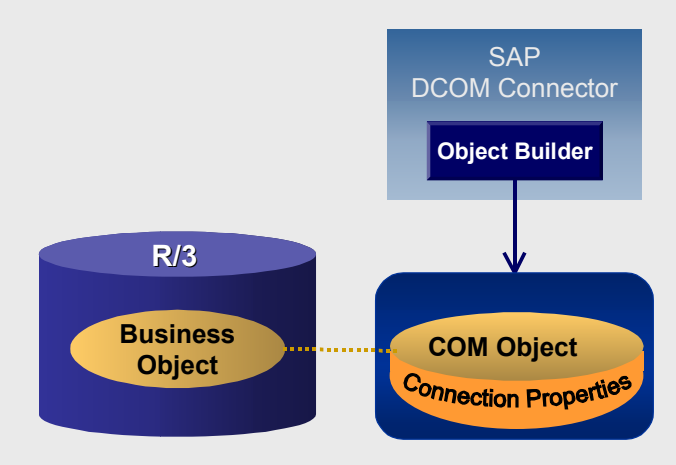
The SAP DCOM Connector

The SAP DCOM Connector product provides remote access to R/3 business logic. More specifically, the SAP DCOM Connector uses services from the Microsoft DCOM MTS to provide a COM interface for calling SAP RFCs and BAPIs remotely. This product can be used instead of the obsolete SAP vrfc32.dll and the SAP logon.ocx products in applications that use DCOM.

The SAP DCOM Connector product with its DCOM Object Builder allows you to create COM objects that act as proxies for business objects or remote functions in an R/3 system.

These COM objects, which we will refer to as the DCOM Connector COM objects, contain properties such as destination system, client, user name, and password, which are necessary for establishing a connection to an SAP system.

As the following diagram illustrates, these connection-related properties exist in the DCOM Connector COM objects in addition to the methods or other attributes of the original business object or remote function.



For example, a DCOM Connector COM object for the *GeneralLedgerAccount* business object contains the various connection properties in addition to the standard *GeneralLedgerAccount* BAPIs, such as *GetDetail*.

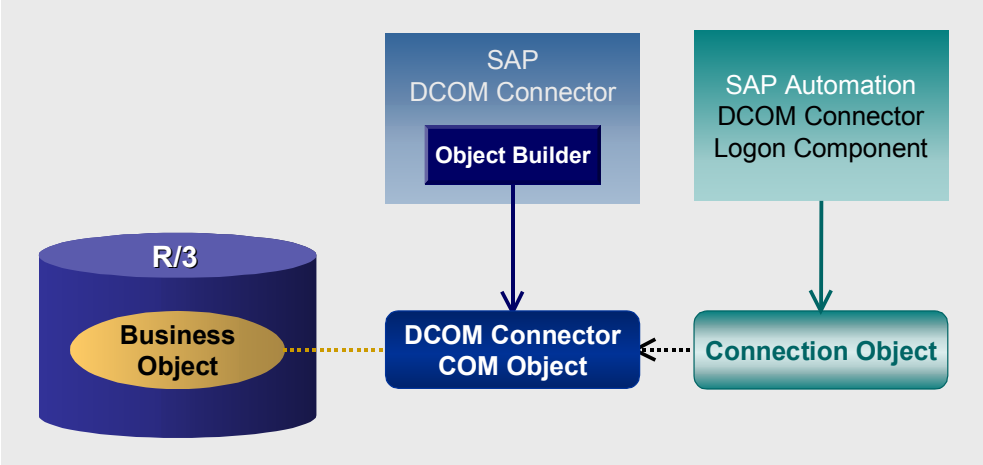
When working with objects based on the SAP DCOM Connector you must set these DCOM Connector COM object connection properties before using the object in a way that requires a connection; for example, before calling one of its BAPIs.

A DCOM Connector COM object includes several standard methods, one of which is *PutSessionInfo*. This method allows you to copy the connection properties into the DCOM Connector COM object. You have to specify the values of the various connection attributes as parameters for this call.

DCOM Connector Logon Component (BC-FES-AIT)

The SAP Automation DCOM Connector Logon

The DCOM Connector Logon Component is part of the SAP Automation suite of products. It provides a Connection class that helps programs using the SAP DCOM Connector handle the connection parameters of DCOM Connector COM objects.



The DCOM Connector Logon Component:

- Provides a Logon dialog with which you can get the necessary connection parameters from an end user
- Allows you to easily copy connection parameters into a DCOM Connector COM object

The DCOM Connector Logon Component can be used instead of the Logon Control (wdtlog.ocx) in applications that use the SAP DCOM Connector.

Features

The DCOM Connector Logon component provides a single class, namely the *Connection* class, for working with various connection parameters.

With the Connection class [Page 28] you can:

- Create a Connection object, whose properties hold logon information
- Provide a Logon dialog to an end user with all the necessary fields for logging onto an R/3 or R/2 system.

  The Logon dialog:

  – Allows you to control whether you allow the end user to choose or change destination systems

  – Allows the end user to add, change, or delete destination definitions

  – Tracks recently used destination systems
- Get various connection properties from the Windows Registry
- Use the various properties of the Connection object to handle logon information
- Copy the connection information from the properties of the Connection object into the connection properties of your DCOM Connector COM objects

DCOM Connector Logon Component (BC-FES-AIT)

Implementation Considerations

The SAP DCOM Connector Logon Component can be used by DCOM-compliant programs, such as those written in Visual Basic, Java, C++, and so on.

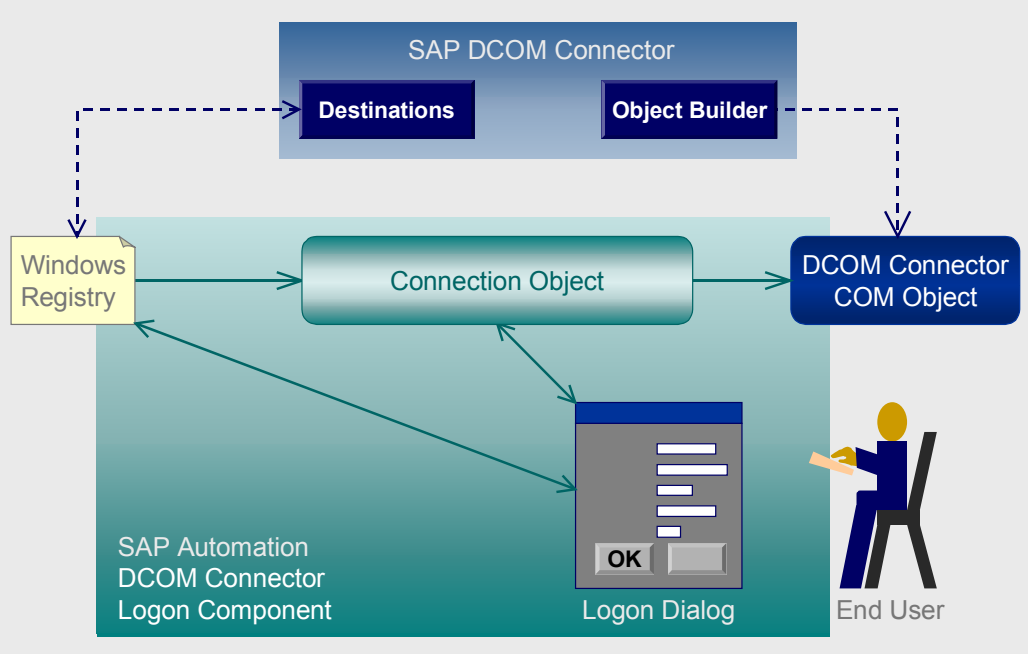
Integration

If you are using the SAP Automation DCOM Connector Logon Component you will be mainly using two of the components of the SAP DCOM Connector:

- The Object Builder, which helps you create the COM objects representing the SAP business objects or function modules, in the client system.
- The Destination Editor, which allows you to define destinations, and then adds the destination entries to the Windows registry.

  Note that an end user can also maintain destinations directly at the Logon dialog of the DCOM Connector Logon Component, instead of using the SAP DCOM Connector.

The following diagram shows the relationship between the SAP DCOM Connector, the COM object you create with it, the Windows registry, the SAP DCOM Connector Logon Component and its Connection object:



Using the DCOM Connector Logon Component

Prerequisites

Before you can use the Connection class, you must:

1. Install the SAP DCOM Connector product.
2. (Optional) Set up the destinations available to your program and your end users [Page 15].

   Use the Destination Editor of the SAP DCOM Connector program to set up those destinations. Alternatively, you can set up destinations directly in the Windows Registry.

   If you set up destination definitions, then the end user of your program can log onto those systems.

   In addition to destinations that you set up, the end user can set up destination(s) at the Logon dialog of the DCOM Connector Logon Component. The user can immediately use those destination definition(s) to log onto an R/3 system.
3. Install the DCOM Connector Logon Component. The installation program also automatically registers the component DLL with the Windows Registry.
4. Set up the component DLL in your programming environment. For example, in Visual Basic use the *Project→References* menu to add a reference to the SAP DCOM Connector Logon Component.

Process Flow

Using the SAP DCOM Connector

When using distributed objects with the SAP DCOM Connector product, you do not need to explicitly log onto an R/3 system in your program. Logging onto a system is done automatically and implicitly whenever the DCOM Connector COM object requires a connection to an SAP System.

Calling a BAPI or calling a function module using the SAP DCOM Connector requires a connection to the SAP system. However, before calling a BAPI or a function module you need only ensure that the connection properties of the DCOM Connector COM object are appropriately set up.

For example, using the GetDetail method of the *GeneralLedgerAccount* business object requires a connection to the R/3 system. However, do not establish this connection in your code. You merely ensure that all the necessary connection properties of the DCOM Connector COM object (which represents the *GeneralLedgerAccount* business object) have the correct values before calling the GetDetail method.

Using the DCOM Connector Logon Component

Using the Connection class of the DCOM Connector Logon Component you can:

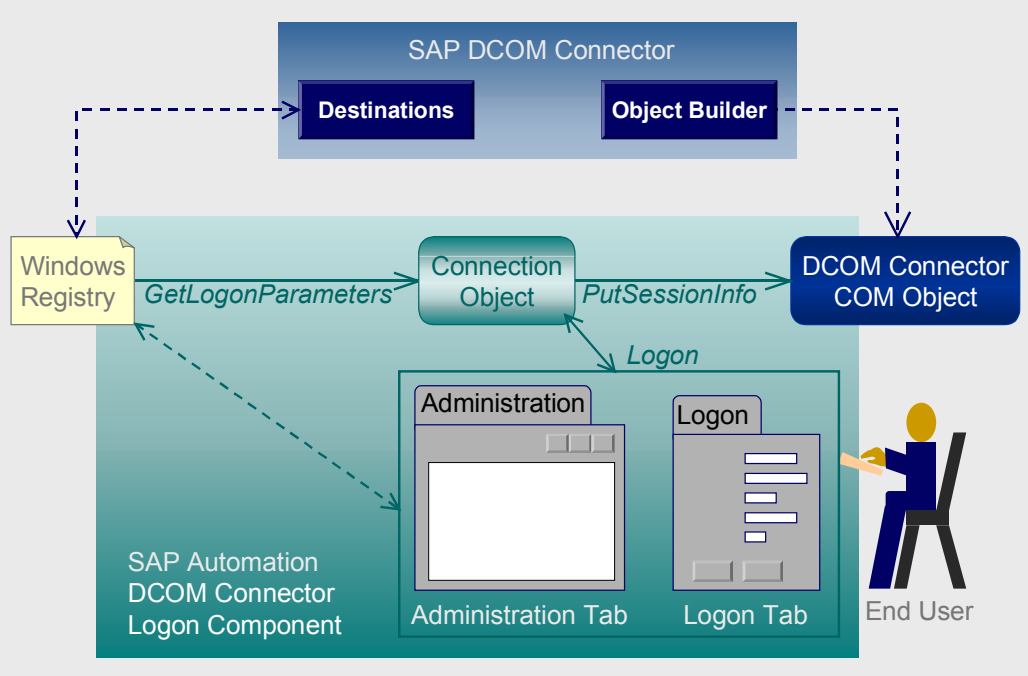
1. Instantiate a Connection object.
2. Get logon parameters from the Windows Registry and store them in the Connection object by using the GetLogonParameters method [Page 32].
3. Optionally display a Logon dialog to the end user by using the Logon method [Page 33].

The end user may add, change, or delete destination definitions at the Logon dialog. Once the user finishes defining a destination, the destination information is applied to the Windows registry.

Once the end user chooses OK at the Logon dialog, the logon parameters are copied into the properties of the Connection object.

4. Regardless of whether you have used the GetLogonParameters or the Logon method for obtaining connection/logon information, you can use the PutSessionInfo method [Page 35] to copy these connection properties into the properties of any DCOM Connector COM object.

The following diagram illustrates the process of using the DCOM Connection Logon Component. It shows the use of the various DCOM Connector Logon methods:



Example

The following Visual Basic code sections create a Connection object, display a Logon dialog to the end user, and then call both a BAPI and a function module requiring logon information. The example lets the user select a destination.

```
' Boolean variable to hold the return value of the Logon method:
Private bLoggedOn As Boolean
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
'...
' Allow the user to select a destination system:
Conn.DestChangeable = True
'
' Call the Logon method to display the Logon dialog
bLoggedOn = Conn.Logon
If bLoggedOn Then
```

Using the DCOM Connector Logon Component

```
...
Else
...
End If

' Using the Connection object with a BAPI:
' Create a COM object for the Customer business object
Dim oCustomerData As Recordset
Dim oReturn As Recordset
Dim oCustomer As SAPCustomer
Set oCustomer = CreateObject("SAP.Customer.1")
' Copy logon properties to the oCustomer COM object
Conn.PutSessionInfo oCustomer
'...
' Call the CheckExistence BAPI of oCustomer
'    This requires that the connection parameters
'    exist in the appropriate properties of oCustomer
oCustomer.BapiCheckExistence Return:=oReturn,
DistributionChannel:= "01", _
Division:= "01", _
SalesOrganization:= "0001", _
CustomerData:=oCustomerData

' Use the Connection object with an RFC:
' Create a COM object for the RFC object
Dim oRFC As Object
Dim oCustList As Recordset
Set oRFC = CreateObject("RFCSampObj.RFCSampObj.1")
' Copy logon properties to the RFC object
Conn.PutSessionInfo oRFC
' Call the function module
'    This requires that the connection parameters
'    exist in the appropriate properties of oRFC
Call oRFC.GetCustList( CustNum:= "", _
CustName:= = "A*",_
pCustList:= oCustList)
```

System Requirements

Development Requirements

To create applications using the DCOM Connector Logon Component, you need the following:

- Windows NT 4.0 (Service Pack 3), Windows 95., or the Windows 98 operating system
- Windows NT 4.0 Option Pack including the MTS development environment
- R/3 Release 4.5A RFC SDK (installed and registered in the system32 directory). This installs the following items, which are required for using the DCOM Connector Logon Component:
  - librfc32.dll
  - SAP DCOM Connector product

Run-time Requirements

The end user of an applications using the DCOM Connector Logon Component needs the following:

- Windows NT 4.0 (Service Pack 3), Windows 95, or the Windows 98 operating system
- Windows NT 4.0 Option Pack including the MTS development environment
- SAP DCOM Connector product. You can install the DCOM Connector by installing the SAP R/3 Release 4.5A RFC SDK.

What's New in Release 4.6B?

- When a user chooses *OK* from the Logon Dialog, the DCOM Connector Logon Component posts the client number, user name, password, and language information entered in the dialog to the Windows Registry.
- When a user chooses *OK* from the Logon Dialog, the DCOM Connector Logon Component validates the entered logon information against the information stored in the R/3 System.
- SAP added the AddDestination [Page 36], EditDestination [Page 39], DelDestination [Page 38], and FindDestination [Page 41] methods to the Connection object. This allows users to add, edit, or delete destinations without using the Logon Dialog.
- Users are no longer allowed to enter a destination in the Logon Dialog's *Destination* combo box. Instead, users must choose a destination from the combo box's selection list.

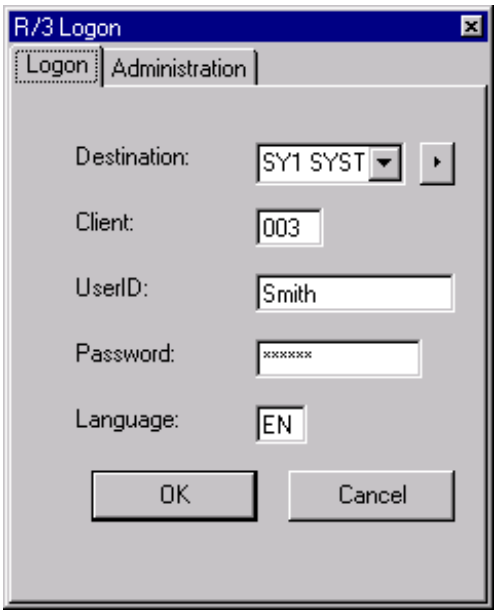
The Logon Dialog

Use

The Logon dialog contains two tabs allowing the user to perform the following tasks:

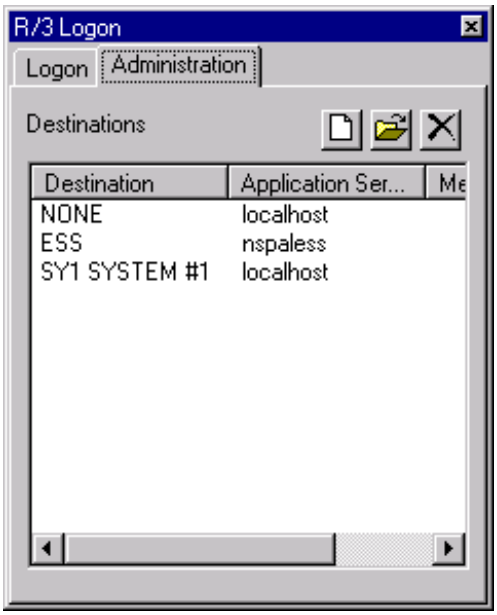
Tab	Tasks
Administration	Add, change, or delete destination systems
Logon	Enter logon information for a particular destination

The following screen shows the *Logon* tab of the *Logon* dialog and some sample values:



The following screen shows the *Administration* tab of the *Logon* dialog and some sample values:

The Logon Dialog



Setting Up Destinations

Use

Before you can log onto an R/3 system through the DCOM Connector Logon component, you must set up at least one destination definition.

A destination is an entry in the Windows Registry containing the SAP logon parameters. These parameters include *system name*, *message server*, *system number*, and *client*, and optionally, *user*, *password*, *language* and *nocleanup*.

In a production environment, you are not likely to have the values of the R/3 user and password in the registry. Your client program can get the user name and password at runtime from the end user. You can use the Logon method [Page 33] to allow the end user to provide this information in a dialog box, which then copies the relevant information into the properties of the Connection object.

Procedure

Outside of the DCOM Connector Logon Component

You can set up destinations for your program and end user to use with either one of the following methods:

- By manually editing entries in the Windows Registry [Page 16], possibly with the *regedit* editor
- By using the SAP DCOM Connector

  This sets up the necessary entries in the Windows Registry.

When setting up destinations outside of the DCOM Connector Logon Component, it is easier to set up destinations through the DCOM Connector. However, an end user can also easily set up destinations when using the Logon dialog of the DCOM Connector Logon Component.

Within the DCOM Connector Logon Component

The end user can set up destinations at the Administration tab of the Logon dialog [Page 17].

Destination Entries in the Windows Registry

The following are two sample Registry destination entries (available in this format only if you edit them directly): one is a default destination entry, called *NONE*. The other is a user-defined entry for a destination called *MySystem* (The actual values of the various parameters were replaced with generic names).

```
\HKEY_LOCAL_MACHINE\SOFTWARE\SAP\MTS\Destination:
NONE= "TYPE=3 ASHOST=myhost SYSNR=00 CLIENT=003 LANG=E NOCLEANUP=1"
MySystem="TYPE=3 ASHOST=hs0020.mynetwork.mycompany.net SYSNR=01
CLIENT=004 USER=smith PASSWD=mypassword LANG=EN"
```

If you define destinations with the SAP DCOM Connector, the Registry entries are encoded.

Using the Logon Dialog to Set Up Destinations

Purpose

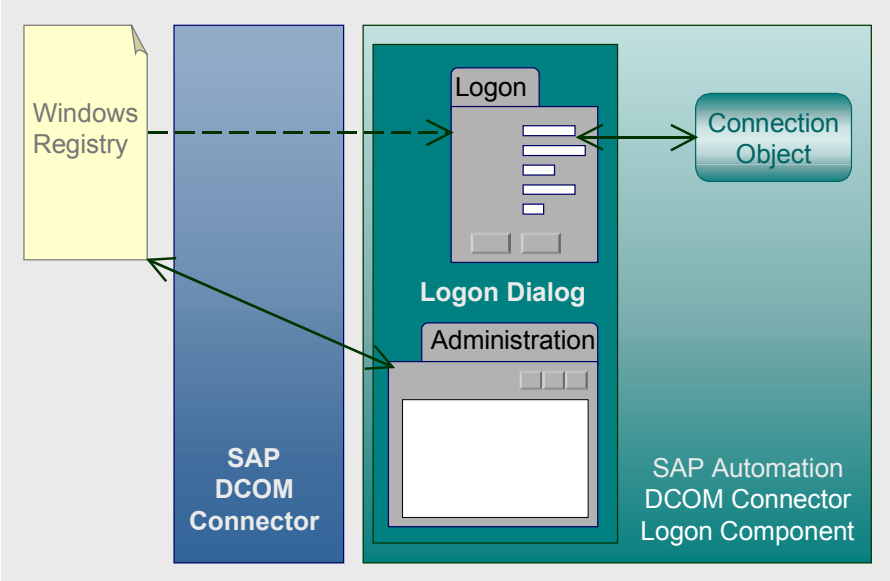
A destination definition in the Windows Registry contains connection information that the SAP DCOM Connector can use for logging into the destination system on behalf of its COM objects.

The DCOM Connector Logon Component allows the end user to define or change the definition of destinations through the *Administration* tab of the *Logon* dialog.

Process Flow

1. The user sets up destination(s) through the *Administration* tab of the *Logon* dialog. In addition to defining the parameters of the destination system, the user can define the client, user and password to use when logging into that system.
2. The DCOM Connector Logon Component uses the CCAdmin component of the SAP DCOM Connector to write the information to the Windows Registry.
3. When the user selects a destination at the *Logon* tab of the *Logon* dialog, the details of that destination are taken from the Windows Registry and from the properties of the Connection object.
4. The user can overwrite the client, user name, password, and language information at the *Logon* tab of the *Logon* dialog. Changes that the user makes at the *Logon* tab of the *Logon* dialog are applied back to the Connection object properties. They are also written to the Windows Registry.

The following diagram illustrates the flow of information between the DCOM Connector Logon Component, its Connection object, and the Windows registry.



Result

Changes to destination definitions done in the Administration tab change the destination entry in the Windows Registry, and are therefore more permanent. Changes at the Logon tab only affect

the Connection object. For example, if the user changes client information for the destination at the Administration tab, the client information changes for the destination entry in the Registry. However, if the user changes the client information at the Logon tab, this information is only written to the Connection object.

Adding or Changing a Destination

Use

A destination definition enables your program to connect to the destination system when its COM objects need such a connection.

The following procedure describe how to add a new destination or change an existing destination definition.

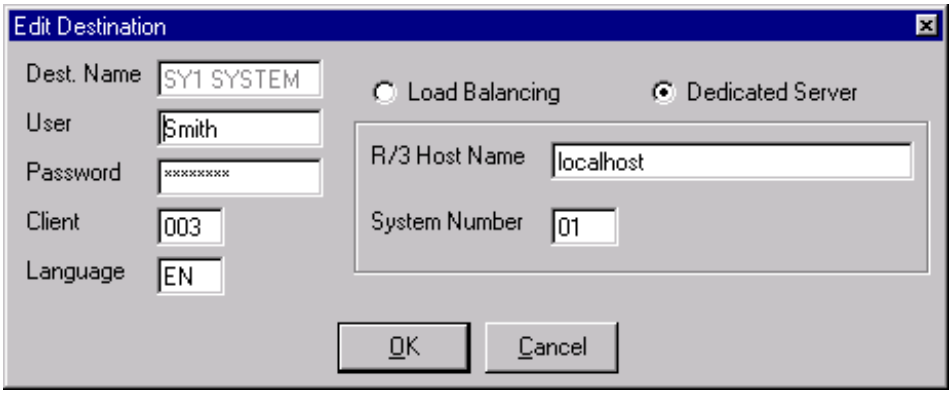
Procedure

1. At the *Logon* dialog, choose the *Administration* tab.
2. Perform one of the following steps:

Task	Action
Add a new destination	Choose *Add Destination.* Enter a name for the destination definition at the *Dest. Name* This is a string of up to 25 characters describing the destination system.
Edit an existing destination	Select the destination name from the *Destinations* list, and choose *Edit Destination*.

   The *Edit Destination* dialog appears, allowing you to define a new destination or to change an existing destination definition.

   The following screen shows an example of the *Edit Destination* dialog for editing an existing destination:



3. Choose between *Load Balancing* and *Dedicated Server*.

   When using a dedicated server you specify the actual R/3 application server to use when logging on. When using load balancing you specify a message server which then selects the least busy application server for you to log onto.

   Once you choose between *Load Balancing* and *Dedicated Server*, enter data as required for logging onto the appropriate system:

Mode	Field	Enter

Load Balancing	*Message Server*	The name of the computer acting as the message server. This can be in a format similar to: **"hs0020.mynetwork.mycompany.net" or it can be an IP address**. You may prefix the computer name with a router name in the following format: **/H/***router-name***/H/***computer-name*
	R/3 Sys Name	The three-character system name
	Group	Group name, such as *PUBLIC*
Dedicated Server	*R/3 Host Name*	The computer name of the application server to use. Use the same format as when specifying the computer name for the *Message Server*, including the option to use a router prefix.
	System Number	R/3 system number

4. Enter your user (user name), password, client, and language to use when logging onto the system you have defined above.

Result

When you choose OK at the *Edit Destination* dialog your changes apply to the Windows Registry, and they are reflected in the *Logon* tab of the *Logon* dialog.

Deleting Destinations

Procedure

5. At the *Logon* dialog, choose the *Administration* tab.
6. Select the destination to be deleted at the *Destinations* list.
7. Choose *Delete Destination*.
8. Confirm the deletion at the *Delete Destination* message box.

Result

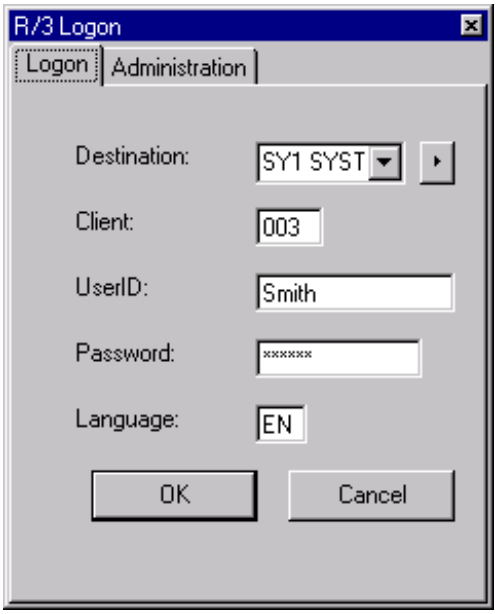
The destination is deleted from the Destinations list at the *Administration* tab. It is also deleted from the Windows Registry.

Using the Logon Dialog to Enter Logon Information

Use

The *Logon* tab of the *Logon* dialog allows you to enter updated user information for logging onto a specific R/3 system.

The following screen shows the *Logon* tab of the *Logon* dialog with some sample values:



Prerequisites

Before entering data at the *Logon* tab, at least one destination must be defined [Page 15].

Procedure

1. Choose the *Logon* tab.
2. Select a destination at the *Destination* field. You can do so by either selecting from the destinations list or from the *Recently-used destinations* list as described in the following table:

Field	Description

Destination	Displays the selected destination. Allows you to select from a list of available destinations. The list of available destinations is taken from the Windows Registry. The first time your client machine uses the DCOM Connector Logon Component, the drop-down list contains destination systems in the order they appear in the Windows Registry, with the first entry appearing in the *Destination* field. After connecting to an R/3 system through the DCOM Connector Logon Component at least once, the *Destination* field shows the most recently accessed system. The drop-down list contains the other Registry entries.
Recently-used destinations (Arrow-shape button next to the Destination field)	Allows you to view and select destinations from a list of up to five destinations that you have accessed previously. The list is organized by the order in which the destinations were accessed. The most recently accessed destination appears at the top of the list. The list of recently accessed destinations is stored in the following file on your hard disk: C:\WINNT\SAPDLogn.rid

The Destination drop-down list and the Recently-used destination list may be disabled, in which case you cannot select a destination other than the one that appears in the dialog by default.

3. If the Windows Registry contains a definition for the destination you have selected, the *Logon* dialog displays this information.

   If any of the information is missing from the destination entry in the Registry, or if you wish to override any of the information from the Registry, you can enter new data at the *Logon* dialog.

   The following table describes how the various fields are treated:

Client, UserID, Language	The Logon dialog gets the values for the *Client*, *UserID*, and *Language* fields from the Registry based on the selected destination. You have to supply values for any of the fields that are not available in the Registry entry. You can override any of the existing values.

Password	The *Password* is not copied from the Registry, even if it is included in the Registry entry for the destination. This is because the Logon Dialog cannot retrieve the password. Initially the password string in the Logon dialog is '!@#$%^', but it is hidden with a string of asterisk characters (*). If you types in a password and choose OK at the Logon dialog, the password entered is copied into the Password property of the Connection object. If you do not enter a password, the password is taken from the registry when necessary, if it is available in the Registry. If you do not enter a password, and it is not available at the Registry, the program would not be able to log onto the destination system.

4. Choose OK.

Result

When you choose OK at the *Logon* tab, the DCOM Connector Logon Component fills the relevant properties of the Connection object with the data from the equivalent fields of the dialog.

Note that the *Logon* dialog does not necessarily connect you to an R/3 system when you choose OK. It merely gets the necessary information to be used whenever a connection is required later on.

Calling the Logon Dialog

Use

The following procedure describes how to invoke the *Logon* dialog from a program.

Prerequisites

If you wish to include destinations for the user to choose from, you must set up these destinations [Page 15]. You can do so by either setting the destination(s) directly at the Windows Registry, or by using the SAP DCOM Connector.

This step is not required if you expect the user to set up the destinations at the Logon dialog [Page 17].

Procedure

1. Set the value of the DestChangeable property of the Connection object to specify whether the user can select or change destination system. See the details below.
2. Call the Logon method [Page 33].
3. Check the return value. If it is True, the user chose OK.

Controlling the Destination Fields

The Destination Field and the Destination Property of the Connection Object

You can specify the destination to be displayed at the *Destination* field, by assigning the destination string to the Destination property of the Connection object, before calling the *Logon* dialog with the Logon Method.

The DestChangeable Property of the Connection Object

The DestChangeable property of the Connection object determines if the user can select or change destination at the Logon tab of the Logon dialog.

If DestChangeable is set to False, both the list of available destinations and the list of recently-used destinations are disabled.

If DestChangeable is set to True, the user can access both destination lists, and select a destination from either list.

Controlling Destination Selection

If you want your end user to use a specific destination system perform the following before calling the *Logon* dialog:

1. Prevent the user from selecting or changing a destination at the *Logon* dialog, by setting the value of the DestChangeable property of the Connection object [Page 29] to False.
2. Set the value of the Destination property of the Connection object to the desired system.

Result

When the user chooses OK at the *Logon* tab, the values of the *Destination*, *Client*, *UserID*, *Password*, and *Language* fields are copied into the equivalent Connection properties and the Windows Registry.

Example

The following Visual Basic sections of code display a *Logon* dialog to the end user. The example lets the user select a destination.

```
' Boolean variable to hold the return value of the Logon method:
Private bLoggedOn As Boolean
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
'...
' Allow user to select a destination system:
Conn.DestChangeable = True
'
' Call the Logon method to display the Logon dialog
bLoggedOn = Conn.Logon
If bLoggedOn Then
    ' Display the values of the logon parameters
    MsgBox "Destination: " & Conn.Destination & vbCrLf & _
        "Client: " & Conn.Client & vbCrLf & _
        "User: " & Conn.User & vbCrLf & _
        "Password: " & Conn.Password & vbCrLf & _
        "Language: " & Conn.Language & vbCrLf, _
        vbInformation
Else
    MsgBox "No Logon Information", vbExclamation
End If
```

DCOM Connector Logon Class Reference

The Connection Class

The Connection class is a member of the SAPLogonLib.

Use

A Connection object holds logon/connection information in its properties. It allows you to copy these properties into your DCOM Connector COM objects.

See Also

Connection Class Properties [Page 29], Connection Class Methods [Page 31], Using the DCOM Connector Logon Component [Page 8]

Connection Class Properties

The following table summarizes the properties of the Logon class.

Property Name	Description	Data Type	Modifiable?
ApplicationServer	Application server information, in a format similar to: **"hs0020.mynetwork.mycompany.net" or an IP address**	String	Read-only
Client	Client number, such as "000" or "003"	String	Read/write
DestChangeable	If false, the end-user is not allowed to select or change the destination field in the logon GUI.	Boolean	Read/write
Destination	Destination system, that is, system to log onto. This is a string of up to 25 characters describing the system. It may contain the short name of the system, but it does not have to be the system name or ID. It is the name of the destination entry in the SAP DCOM Connector and in the Windows Registry.	String	Read/write
Host	The first part of the application server string (the portion before the first period). For example, **"hs0020 " if the ApplicationServer is**: **"hs0020.mynetwork.mycompany.net"**	String	Read-only
Language	Logon language	String	Read/write
Password	Password. The default value of the password property of the Connection object when it is first created, is '!@#$%^'.	String	Read/write
SAPRelease	Release number of the R/3 system of the destination	String	Read-only
System	System name	String	Read-only
SystemNumber	System number	String	Read-only
User	User name	String	Read/write

Sources for Property Values

The DCOM Connector Logon Component gets the values for the properties of the Connection object from either the Windows Registry, or from fields of the SAP DCOM Connector product, which in turn gets the values from the Registry.

Even when the end user creates or changes destination definitions at run time through the Logon dialog, the DCOM Connector Logon Component saves this information to the Registry, and then takes the applicable information from the Registry as needed.

Connection Class Properties

The following table lists the various properties of the Connection object, and shows where their values are taken from. It lists the name of the source Windows Registry entry or SAP DCOM Connector field:

Property Name	Source	SAP DCOM Connector Field	Registry Entry
ApplicationServer	Registry (Required entry)	(R/3 Host name field in the DCOM Connector screen)	ashost
Client	Registry	—	client
DestChangeable	User defined	—	—
Destination	User defined	(Name of the Destination entry)	(Name of the Destination entry)
Host	DCOM Connector (the first part of the R/3 host name field)	rfchost	—
Language	Registry	—	lang
Password	Registry	—	passwd
SAPRelease	DCOM Connector	rfcsaprl	—
System	DCOM Connector	rfcsysid	—
SystemNumber	Registry (Required entry)	(System number in the DCOM Connector screen)	sysnr
User	Registry	—	user

Getting Values from the SAP DCOM Connector

The values for the Host, SAPRelease, and System properties are obtained with the *SystemInfo* Method of the CCRegistry object of the SAP DCOM Connector component.

To use this method in your program use the CCADMIN DLL of the SAP DCOM Connector. Call the *SystemInfo* Method of the CCRegistry object, specifying destination. The values of the above fields are returned through the PPSystemInfo recordset parameter of the method.

The Host, SAPRelease, and System properties are only available if the entry in the Windows Registry contains a valid password.

Note that you do not need a valid password in the Registry for a connection to work: your program can get the valid password from the end user, for example.

However, not having a valid password in the Registry affects your ability to query the value of these properties in your program.

Connection Class Methods

Method	Function
GetLogonParameters [Page 32]	Sets the *Client*, *User*, *Password*, and *Language* properties of the Connection object according to the value of these parameters in the Windows Registry entry for the specified destination. If no destination is specified, the value of the Destination property of the Connection object is used.
Logon [Page 33]	Displays a logon dialog to an end user allowing the user to: • add, change, or delete destination definitions • enter logon information for a particular destination entry Any changes in destination definitions are then saved in the Windows Registry. The various logon parameters are copied into the Connection object's properties.
PutSessionInfo [Page 35]	Copies connection properties from the Connection object into your DCOM Connector COM objects. This is similar to the SAP DCOM Connector PutSessionInfo method, but unlike the SAP DCOM Connector, which requires specifying the values of all the logon information as parameters, this version requires only one parameter: the COM object into which to copy the properties.
AddDestination [Page 36]	Adds a new destination entry to the Windows Registry without invoking the logon dialog box. This function overwrites old information in the Windows Registry if you provide an existing destination name. If you want to use a load balancing or dedicated server, you must specify the correct parameters.
DelDestination [Page 38]	Deletes a destination from the Windows Registry without invoking the logon dialog box. If you try to delete a non-existing destination from the Registry, this method does nothing.
EditDestination [Page 39]	Edits a destination in the Windows Registry without invoking the logon dialog box. • If the you want to use load balancing, you must enter MsgServName, SysName, and Group parameters. If you want to use a dedicated server, you need to provide AppServName and SysNum parameters. • You only need to provide parameters you would like to change. If you do not want to change some fields in the Registry, you must leave these parameters blank. • If the destination does not exist in the Windows Registry, this method does nothing.
FindDestination [Page 41]	Finds a destination in the Windows Registry. This method returns True if the destination is found in the Windows Registry; False if otherwise.

GetLogonParameters

Use

Sets the *Client*, *User*, and *Language* properties of the Connection object according to the value of these parameters in the Windows Registry entry for the specified destination. Sets the *Password property of the* Connection object to be the initial string '!@#$%^'. GetLogonParameters sets this dummy string because it cannot retrieve the password from the Windows Registry.

Syntax

GetLogonParameters([optDest])

Parameters

Parameter	Data Type	Description
optDest	String	Destination (a 25-character string), which is the name of the destination in the SAP DCOM Connector (or in the Registry). Optional.

Comments

- If no destination is specified, the value of the Destination property of the Connection object is used.
- If you use the optional parameter to specify destination, then the Destination property of the Connection object is updated as well.

Example

The following code gets the logon parameters of the destination *System1* from the Registry.

```
Dim Conn As New Connection
Conn.GetLogonParameters ("System1")
```

The following code also gets the logon parameters of the destination *System1*, but it takes the value for destination from the property of the Connection object.

```
Dim Conn As New Connection
Conn.Destination = "System1"
Conn.GetLogonParameters
```

See Also

Logon [Page 33], PutSessionInfo [Page 35]

Logon

Use

Invokes a Logon dialog box [Page 13] to allow an end user to:

- add, change, or delete destination definitions
- enter logon information for a particular destination entry

The values entered at the *Destination*, *Client*, *UserID*, *Password*, and *Language* fields are copied into the *Destination*, *Client*, *User*, *Password*, and *Language* properties of the Connection object respectively.

Any changes the user makes in destination definitions are saved in the Windows Registry.

Syntax

Logon()

Parameters

None

Data Type

Boolean

Return Value

True if the user chose OK, False otherwise.

Comments

- The destination displayed at the *Destination* field of the dialog is taken from the Destination property of the Connection object.
- If the Destination property is undefined, then the most recently used destination is displayed at the Destination field of the dialog.

  The list of recently used destinations is stored in the following file on the client's hard disk:

  C:\WINNT\SAPDLogn.rid
- The value of the DestChangeable property of the Connection object determines whether:
  - The user can select a destination at the Destination field of the *Logon* tab.
  - The Recently accessed destination list at the *Logon* tab is enabled

    The value of the DestChangeable property does not change the user's ability to define destinations at the *Administration* tab.
- The values for Client, UserId, and Language are taken from the Windows Registry.
- The Password is initialized to '!@#$%^' and is displayed as a string of asterisks (*) when the dialog is displayed to the user.

Example

The following example displays the *Logon* dialog to the user, allowing the user to select or change destination. The destination displayed at the Destination field when the dialog appears is the most recently used destination.

```
' Boolean variable to hold the return value of the Logon method:
Private bLoggedOn As Boolean
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
'...
' Allow the user to select a destination system:
Conn.DestChangeable = True
'
' Display the Logon dialog
bLoggedOn = Conn.Logon
    If bLoggedOn Then
     ...
    Else
     ...
    End If
```

The following example displays the *Logon* dialog to the user, but without allowing the user to select or change destination. The program specifies *Sy1* as the destination to use.

```
Dim Conn As New Connection
Dim bLogOK As Boolean
Conn.DestChangeable = False
Conn.Destination = "Sy1 SYSTEM #1"
bLogOK = Conn.Logon
```

See Also

GetLogonParameters [Page 32], PutSessionInfo [Page 35], The Logon dialog [Page 13]

PutSessionInfo

Use

Copies Destination, Client, UserID, Password, and Language to the specified DCOM Connector COM object's connection properties.

Syntax

PutSessionInfo(*tObject*)

Parameters

Parameter	Data Type	Description
tObject	Object	The target COM object into which the logon properties are copied

Comments

- This method is similar to the SAP DCOM Connector PutSessionInfo method, but unlike the SAP DCOM Connector, which requires specifying the values of all the logon information as parameters, this version requires only one parameter: the COM object into which to copy the properties.
- If the Password property of the Connection object contains an actual password string, then it is copied into the Password property of the DCOM Connector COM Object.

  If the Password property of the Connection object is '!@#$%^' (which is the default initial value of Password when the Connection object is created) then this means that no actual password had been assigned to it yet. In this case the empty string "" is copied into the Password property of the DCOM Connector COM Object.

Example

The following example copies the connection properties of the Connection object into the appropriate properties of a DCOM Connector COM object representing a customer business object.

```
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
' Create a COM object for the Customer business object
Dim oCustomer As SAPCustomer
Set oCustomer = CreateObject("SAP.Customer.1")
' Copy logon properties to the oCustomer COM object
Conn.PutSessionInfo oCustomer
```

See Also

GetLogonParameters [Page 32], Logon [Page 33]

AddDestination

Use

Adds a new destination entry to the Windows Registry without invoking the logon dialog box.

Syntax

AddDestination(Dest, [Client], [User], [Pwd], [Lang], [AppServName], [SysNum], [MsgServName], [SysName], [Group], [LoadBal])

Parameters

Parameter	Data Type	Description
Dest	String	The new destination name
Client	String	The client number
User	String	The user's logon ID
Pwd	String	The user's password
Lang	String	The language
AppServName	String	The application server's host name
SysNum	String	The system's number
MsgServName	String	The message server's host name
SysName	String	The system's name
Group	String	The group's name
LoadBal	Integer	Specify whether you wish to use load balancing or a dedicated server. '1' indicates that you will be using load balancing and '0' indicates you will be using a dedicated server.

Comments

- If you want to use load balancing, you must enter MsgServName, SysName, and Group parameters. If you want to use a dedicated server, you must provide AppServName and SysNum parameters.
- This function overwrites old information in the Windows Registry if you provide an existing destination name.

Example

The following example adds a destination to the Registry.

```
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
' Add a new destination to the registry (using dedicated server)
Conn.AddDestination "SAPTEST", "800", "I002222", "init", "en",
"nspaless.pal.sap-ag.de", "00", "", "", "", 0
' Add a new destination to the registry (using load balancing)
```

```
' However, this will overwrite the existing entry in the registry
Conn.AddDestination "SAPTEST", "800", "I002222", "init", "en", "", "",
"nspal.pal.sap-ag.de", "ESS", "PUBLIC", 1
```

See Also

DelDestination [Page 38], EditDestination [Page 39], FindDestination [Page 41]

DelDestination

Use

Deletes a destination from the Windows Registry without invoking the logon dialog box.

Syntax

DelDestination(Dest)

Parameters

Parameter	Data Type	Description
Dest	String	The destination name

Comments

- If you try to delete a non-existing destination from the Registry, this method does nothing.

Example

The following example deletes a destination from the Windows Registry.

```
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
' Add a new destination to the registry (using dedicated server)
Conn.AddDestination "SAPTEST", "800", "I002222", "init", "en",
"nspaless.pal.sap-ag.de", "00", "", "", "", 0
' Delete a destination from the registry
Conn.DelDestination "SAPTEST"
' Deleting a non-existing destination from the registry is OK.
Conn.DelDestination "ESS"
```

See Also

AddDestination [Page 36], EditDestination [Page 39], FindDestination [Page 41]

EditDestination

Use

Edits a destination in the Windows Registry without invoking the logon dialog box.

Syntax

EditDestination(Dest, [Client], [User], [Pwd], [Lang], [AppServName], [SysNum], [MsgServName], [SysName], [Group], [LoadBal])

Parameters

Parameter	Data Type	Description
Dest	String	The destination name
Client	String	The client number
User	String	The user's logon ID
Pwd	String	The user's password
Lang	String	The language
AppServName	String	The application server's host name
SysNum	String	The system's number
MsgServName	String	The message server's host name
SysName	String	The system's name
Group	String	The group's name
LoadBal	Integer	Specify whether you wish to use load balancing or a dedicated server. '1' indicates that you will be using load balancing and '0' indicates you will be using a dedicated server.

Comments

- If the you want to use load balancing, you must enter MsgServName, SysName, and Group parameters. If you want to use a dedicated server, you need to provide AppServName and SysNum parameters.
- You only need to provide parameters you would like to change. If you do not want to change some fields in the Registry, you must leave these parameters blank.
- If the destination does not exist in the Windows Registry, the EditDestination method does nothing.

Example

The following example edits a destination in the Registry.

```
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
' Add a new destination to the registry
```

```
Conn.AddDestination "SAPTEST", "800", "I002222", "init", "en",
"nspaless.pal.sap-ag.de", "00", "", "", "", 0
' Edit this destination in the registry, so only user ID and password
' are changed, else remain the same.
Conn.EditDestination "SAPTEST", , "I002200", "newpw"
```

See Also

AddDestination [Page 36], DelDestination [Page 38], FindDestination [Page 41]

FindDestination

Use

Finds a destination in the Windows Registry.

Syntax

FindDestination(Dest)

Parameters

Parameter	Data Type	Description
Dest	Boolean	The destination name

Return Value

Returns True if the destination is found in the Windows Registry; False if otherwise.

Example

The following code example checks if a destination is in the Registry. If it is, it allows you to edit the destination; if not, it adds it as a new destination.

```
' Instantiate the Connection object:
Private Conn As New SAPLogonLib.Connection
If Conn.FindDestination(Dest) = True
Then
      Conn.EditDestination "SAPTEST", "800", "I002222", "init", "en",
"nspaless.pal.sap-ag.de", "00", "", "", "", 0
Else
      Conn.AddDestination "SAPTEST", "800", "I002222", "init", "en",
"nspaless.pal.sap-ag.de", "00", "", "", "", 0
End If
```

See Also

AddDestination [Page 36], DelDestination [Page 38], EditDestination [Page 39]