IADsTools Documentation

This document provides information on calling functions in the IADsTools Microsoft® ActiveX® DLL from a script.

Note: When passing string variables as parameters, this DLL requires you to use the CStr( ) function to pass the STRING data type. For example, the following VBScript triggers Active Directory™ on “server1” to pull any changes from “server2” that have been recorded for the Configuration information (includes Sites, Subnets, and domain controller data). Instead of passing the actual strings, variables are used.

```
Dim TargetServer
Dim SourceServer
Dim NamingContext

TargetServer="server1"
SourceServer="server2"
NamingContext="cn=configuration,dc=mydomain,dc=com"

Set DLL=CreateObject("IADsTools.DCFunctions")
Result=DLL.ReplicaSync(Cstr(TargetServer),Cstr(NamingContext),Cstr(SourceServer))
if result=-1 Then
        Wscript.Echo "The error returned was: " + DLL.LastErrorText
else
        Wscript.Echo "The command completed successfully."
end if
```

GetPerformanceData()

Type: Function.

Purpose: Reads the current value of the counters loaded in InitPerformanceData().

Parameters: None, however, the counters to be processed must be populated in the private COUNTERS array by calling InitPerformanceData first.

Return: If successful, a LONG representing number of performance counters read successfully. If unsuccessful, returns -1 as LONG.

TranslateNT4ToDN()

Type: Function.

Purpose: Converts Windows NT 4.0-style credentials (domain\user) to a distinguished name by looking up the account and reading the distinguished name identifying that object (achieved by ADSI).

Parameters:

STRING	*Nt4Name*
STRING	*Password*
INTEGER	*Source*
INTEGER	*UseCreds*

Explanations:
Nt4Name
Name of account to convert - this must be in the format "domain\user".
Password
Password to use when connecting to a server to do the resolution.
Source
Identifies how to resolve the name.
1=By domain name (extracted from passed in domain\user).
2=By domain controller (DsGetDcName is called based on passed in domain name from domain\user).
3=Global Catalog server (currently disabled).
UseCreds
0 (zero) or no value= use credentials of currently logged-on user
1 = use credentials supplied from SetUserCredentials()

Return: If successful, STRING containing distinguished name of users account. If unsuccessful, empty STRING.

TranslateDNToNT4()

Type: Function.

Purpose: Converts distinguished name of a user account (cn=administrator,cn=users,...) to a Windows NT 4.0-style domain\user format (achieved by ADSI).

Parameters:

STRING	*DNString*
INTEGER	*Source*

Explanations:
DNString
Distinguished name String of name to convert - this must be in the format "cn=user1,cn=users,dc=Microsoft,dc=com".
Source
Identifies how to resolve the name.
1=By domain name (currently disabled).

2=By domain controller (DsGetDcName is called with domain name extracted from distinguished name).
3=Global Catalog server (currently disabled).

Return: If successful, STRING containing Windows NT 4.0-style user account domain\userID. If unsuccessful, empty STRING

InitPerformanceData()

Type: Function.

Purpose: Initializes the counters to be queried when GetPeformanceData() is called.

Parameters:

STRING	*ServerName*

Explanations:
ServerName
ServerName where remote query will take place.

Return: If successful, returns 0 (zero). If unsuccessful, returns 1 (one).

PerfCounterName()

Purpose: Returns the Perf Counter Name requested in an enumeration of Group Policy objects returned from the GetPerformanceData() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of Perf Data. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the Perf Counter as STRING. If unsuccessful, returns an empty STRING.

PerfCounterValue()

Purpose: Returns the Perf Counter value requested in an enumeration of Group Policy objects returned from the GetPerformanceData() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of Perf Data. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the value of the Perf Counter as STRING. If unsuccessful, returns an empty STRING.

ClosePerformanceData()

Type: Function.

Purpose: Releases resources used by InitPerformanceData() and GetPerformanceData().

Parameters: None.

Return: If successful, returns 0 (zero). If unsuccessful, returns 1 (one).

GetSiteForServer()

Type: Function.

Purpose: Resolves the site the computer is a member of.

Parameters:

STRING	*ServerName*
INTEGER	*UseCredentials*

Explanations:
ServerName
FQDN or Net name of the server to determine the site for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().
UseCache (optional)
0 (zero) or no value= do not use the existing cache, poll the server real-time.
1 = use the cache if possible. If the entry is not found in the cache, poll the server real-time.

Return: If successful, returns Site name as STRING. If unsuccessful, returns empty STRING.

EnableDebugLogging()

Type: Function.

Purpose: Sets the level of debugging for the IADsTools DLL.

Parameters:

INTEGER	*Level*
STRING	*LogPath*

Explanations:
Level
1=Errors Only.
2=Errors and Warnings.
3=Errors, Warnings, and Informational Messages.
<anything else> disables logging.
LogPath
If specified, routes events to the specified log file instead of the Event Log.

Return: None.

SetUserCredentials()

Type: Function.

Purpose: Stores the credentials to be used for subsequent calls by functions in the DLL - saves on having to pass these in multiple times to different functions.

Parameters:

STRING	*UserName*
STRING	*DomainName*
STRING	*UserDN*
STRING	*Password*

Explanations:
UserName
The user account ID (such as "username").
DomainName
The NetBIOS logon domain for the user (for example, "Microsoft").
UserDN
The distinguished name of the users account (for example, "cn=username,cn=users,dc=Microsoft,dc=com").
Password
Users password.

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns 1 (one) as LONG.

Comments: Use TranslateNT4toDN and TranslateDNtoNT4 to get the parameters.

ClearUserCredentials()

Type: Function.

Purpose: Clears the credentials cache to avoid using incorrect data.

Parameters: None.

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns 1 (one) as LONG.

GetInterSiteTopologyGenerator()

Type: Function.

Purpose: Given a site, gets the single-part name (from the distinguished name) of the computer that owns the role of creating/maintaining the inter-site connection objects for all servers in a site.

Parameters:

STRING	*ServerName*
STRING	*SiteName*
INTEGER	*UseCreds*

Explanations:

ServerName
FQDN or Net name of the server to use as the source for the information.
SiteName
The Site Name to determine the ISTG for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the name of the ISTG as STRING. If unsuccessful, returns empty STRING.

GetSiteList()

Type: Function.

Purpose: Generates a list of the sites given a server to source the data from. The output of this is stored in the property SiteEntryName(*i*) where *i* is the number of sites found.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:
ServerName
  FQDN or Net name of the server to source the information from.
UseCreds (optional)
  0 (zero) or no value= use credentials of currently logged-on user.
  1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns number of Sites found as LONG. If unsuccessful, returns -1 as LONG.

GetSiteLinks()

Type: Function.

Purpose: Generates a list of the site links given a server to source the data from. The output of this is stored in the property SiteLinkName(*i*) where *i* is the number of Sites found.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:
ServerName
  FQDN or Net name of the server to source the information from.
UseCreds (optional)
  0 (zero) or no value= use credentials of currently logged-on user.
  1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns number of site links found as LONG. If unsuccessful, returns -1 as LONG.

GetSiteLinkBridges()

Type: Function.

Purpose: Generates a list of the Site Link Bridges given a server to source the data from. The output of this is stored in the property SiteLinkBridgeName(*i*) where *i* is the number of Site Link Bridges found.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:

ServerName
    FQDN or Net name of the server to source the information from.
UseCreds (optional)
    0 (zero) or no value= use credentials of currently logged-on user.
    1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns number of site link bridges found as LONG. If unsuccessful, returns -1 as LONG.

GetSiteLinkBridgeProperties()

Type: Function.

Purpose: Queries Active Directory™ for the configuration of a given Site Link Bridge.

Parameters:

STRING	*RemoteServer*
STRING	*SiteLinkBridgeName*
STRING	*Transport*
INTEGER	*UseCreds*

Explanations:
RemoteServer
    FQDN or Net name of the server to use as the source for the information.
SiteLinkBridgeName
    The textual name of the requested site Link Bridge.
Transport
    "IP" or "SMTP" (or appropriate Transport).
UseCreds (optional)
    0 (zero) or no value= use credentials of currently logged-on user.
    1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

Comments: Returns data about the Site Link Bridge in the properties beginning with "SiteLinkBridge".

SiteLinkBridgeTransport()

Purpose: Returns the transport of the requested Site Link Bridge in an enumeration of site links from the GetSiteLinkBridges() function.

Parameters:

LONG	*Index*

Explanation of Parameter:

Index
The item number in the enumerated list of site link bridges. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the Site Link Bridge transport as STRING. If unsuccessful, returns an empty STRING.

SiteLinkBridgeSiteCount()

Purpose: Returns the number of sites in the list associated with the most recent queried Site Link Bridge.

Parameters: None.

Return: The number of sites as LONG. If unsuccessful, returns –1 as LONG.

SiteLinkBridgeSiteList()

Purpose: Returns the name of each site present in a given Site Link Bridge

Parameters:
LONG *Index*

Explanation of Parameters:
Index
The item number in the enumerated list of Site Link Bridges- if nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the site as STRING. If unsuccessful, returns an empty STRING.

GetDSAConnections()

Type: Function.

Purpose: Generates a list of NTDS Connection Objects for a given server.

Parameters:
STRING *ServerName*
STRING *SiteName*
STRING *RequestedServer*
INTEGER *UseCreds*

Explanations:
ServerName
FQDN or Net name of the server to use as the source for the information.

SiteName
    The name of the Site the Requested Server is in.
RequestedServer
    The FQDN or NetBIOS name of the server for which the connections should be enumerated.
UseCreds (optional)
    0 (zero) or no value= use credentials of currently logged-on user
    1 = use credentials supplied from SetUserCredentials()

Return: If successful, returns number of NTDS connections found as LONG. If unsuccessful, returns -1 as LONG.

Comments: Additional information about each NTDS connection object is stored in the properties beginning with "DSAConnection".

ConvertLDAPToDNS()

Type: Function.

Purpose: Converts "dc=Microsoft,..." to Microsoft.com format.

Parameters:
STRING   *LDAPName*

Explanations:
LDAPName
    LDAP string beginning with domain controller (for example, "dc=Microsoft,dc=com"). This will get converted to Microsoft.com.

Return: If successful, returns DNS style string as STRING. If unsuccessful, returns empty STRING.

CheckDNForSpecialChars()

Type: Function.

Purpose: Checks a string for special characters that need to be formatted specifically for being passed in LDAP queries and returns the new string to the caller.

Parameters:
STRING   *DistinguishedName*

Explanations:
DistinguishedName
    Distinguished name without the preceding "WinNT://" or "LDAP://"

Return: If successful, returns checked/formatted distinguished name as STRING. If unsuccessful, returns empty STRING.

ConvertDNSToLDAP()

Type: Function.

Purpose: Converts DNS format "Microsoft.com" to LDAP "dc=Microsoft,dc=com".

Parameters:

STRING	*DNSName*

Explanations:
DNSName
DNS string in the form of "Microsoft.com". This will get converted to "dc=Microsoft,dc=com"

Return: If successful, returns LDAP style string as STRING. If unsuccessful, returns empty STRING.

TriggerKCC()

Type: Function.

Purpose: Triggers the Knowledge Consistency Checker on the specified server to check the current topology to ensure that it has accounted for any recent changes.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of server to remote this call to (trigger the KCC on).
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

GetServersInSite()

Type: Function.

Purpose: Enumerates the domain controllers in a given site.

Parameters:

STRING	*ServerName*
STRING	*SiteName*
INTEGER	*UseCreds*

Explanations:

ServerName
FQDN or Net name of the server to use as the source for the information.

SiteName
The name of the site to enumerate the servers for.

UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of servers found as LONG. If unsuccessful, returns -1 as LONG.

GetServersInSiteWithWritableNC()

Type: Function.

Purpose: Enumerates the domain controllers in a given site with a given writeable NC and returns the server names in the property ServerInSiteEntryName(*i*) where *i* is the number of servers returned.

Parameters:

STRING	*RemoteServer*
STRING	*SiteName*
STRING	*RequestedNC*
INTEGER	*UseCreds*

Explanations:

RemoteServer
FQDN, IP Address, or NetBIOS name of the server to use as the source for the information.

SiteName
The name of the site to enumerate the servers for.

RequestedNC
The name of the NC to use as prerequisite for servers enumerated.

UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of servers found as LONG. If unsuccessful, returns -1 as LONG.

GetBridgeHeadsInSite()

Type: Function.

Purpose: Enumerates the domain controllers acting as bridgeheads in a given site returns the server names in the property BridgeHeadInSiteName(*i*) where *i* is the number of servers returned.

Parameters:
STRING *ServerName*
STRING *SiteName*
INTEGER *UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the server to use as the source for the information.
SiteName
The Site Name to determine the bridgeheads for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of servers found as LONG. If unsuccessful, returns –1 as LONG.

GetGCList()

Type: Function.

Purpose: Generates a list of the global catalog servers given a server to source the data from. The output of this is stored in the property GCName(*i*) where *i* is the number of s found.

Parameters:
STRING *ServerName*
INTEGER *UseCreds*

Explanations:
ServerName
FQDN or Net name of the server to use as the source for the information.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of global catalogs found as LONG. If unsuccessful, returns -1 as LONG.

BridgeHeadName()

Purpose: Returns the name of the requested bridgehead in an enumeration of sites from the GetBridgeHeadsInSite() function

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of bridgeheads. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the bridgehead server name as STRING. If unsuccessful, returns an empty STRING.

GCName()

Purpose: Returns the name of the requested in an enumeration of sites from the GetGCList() function

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of s. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name as STRING. If unsuccessful, returns an empty STRING.

ServerInSiteEntryUUID()

Purpose: Returns the UUID of the requested server in an enumeration from GetServersInSite().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of servers. If nothing is passed in or the list has not been enumerated, an error is recorded

Return: If successful, returns the UUID as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionServerName()

Purpose: Returns the name of the requested ntdsConnection server name in an enumeration from GetDSAConnections().

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the server name as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionReasonCount()

Purpose: Returns the number of reasons for the connection in an enumeration from GetDSAConnections()

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the number of reasons as LONG. If unsuccessful, returns –1 as LONG.

DSAConnectionReasonPartition()

Purpose: Returns the associated directory partition that is replicated over a given connection in an enumeration from GetDSAConnections()

Parameters:

LONG	*ConnectionIndex*
LONG	*ReasonIndex*

Explanations:
ConnectionIndex

The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.
ReasonIndex
The reason number in the enumerated list of reasons for this connection. This can be determined by using **DSAConnectionReasonCount().** If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the directory partition name as STRING. If unsuccessful, returns an empty STRING

DSAConnectionReasonCode()

Purpose: Returns the associated "reason" why the connection was established for a particular directory partition in an enumeration from GetDSAConnections()

Parameters:

LONG	*ConnectionIndex*
LONG	*ReasonIndex*

Explanations:
ConnectionIndex
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.
ReasonIndex
The reason number in the enumerated list of reasons for this connection. This can be determined by using **DSAConnectionReasonCount().** If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns a comma delimited string of reason codes as STRING. If unsuccessful, returns an empty STRING

Comments: Return values include: "NONE", "GC", "RING", "OPTIMIZATION", "STALE_SERVER"

DSAConnectionAdminGenerated()

Purpose: Returns whether or not the requested ntdsConnection was generated by the administrator.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "AUTO" or "YES" as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionNotifyOverride()

Purpose: Returns whether or not the requested ntdsConnection is set to override the default for inter-site notification

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "YES" or "NO" as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionNotify()

Purpose: Returns whether or not the requested ntdsConnection is enabled for inter-site notification

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "YES" or "NO" as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionTwoWay()

Purpose: Returns whether or not the requested ntdsConnection is enabled for two-way replication

Parameters:
LONG *Index*

Explanations:
Index

The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "YES" or "NO" as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionEnabled()

Purpose: Returns whether or not the requested ntdsConnection is enabled or not

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "YES" or "NO" as STRING. If unsuccessful, returns an empty STRING.

DSAConnectionName()

Purpose: Returns the name of the requested ntdsConnection in an enumeration from GetDSAConnections().

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of connections. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name as STRING. If unsuccessful, returns an empty STRING.

ServerInSiteEntryName()

Purpose: Returns the name of the requested server in an enumeration from GetServersInSite().

Parameters:

LONG	*Index*

Explanations:

Index
The item number in the enumerated list of servers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the server name as STRING. If unsuccessful, returns an empty STRING.

SiteEntryName()

Purpose: Returns the name of the requested site in an enumeration of sites from the GetSiteList() function

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of sites. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the site name as STRING. If unsuccessful, returns an empty STRING.

SiteLinkEntryDN()

Purpose: Returns the distinguished name of the requested Site Link in an enumeration of site links from the GetSiteLinks() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of sites. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the site link distinguished name as STRING. If unsuccessful, returns an empty STRING.

SiteLinkEntryName()

Purpose: Returns the name of the requested Site Link in an enumeration of Site Links from the GetSiteLinks() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of sites. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the Site Link name as STRING. If unsuccessful, returns an empty STRING.

SiteLinkEntryType()

Purpose: Returns the type of the requested Site Link in an enumeration of Site Links from the GetSiteLinks() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of sites. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the Site Link type as STRING. If unsuccessful, returns an empty STRING.

SiteLinkBridgeEntryDN()

Purpose: Returns the distinguished name of the requested Site Link Bridge in an enumeration of Site Link Bridges from the GetSiteLinkBridges() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of site link bridges. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the Site Link Bridge distinguished name as STRING. If unsuccessful, returns an empty STRING.

SiteLinkBridgeEntryName()

Purpose: Returns the name of the requested Site Link Bridge in an enumeration of Site Link Bridges from the GetSiteLinkBridges() function

Parameters:

LONG	*Index*

Explanations:

Index

The item number in the enumerated list of site link bridges. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the Site Link Bridge name as STRING. If unsuccessful, returns an empty STRING.

GetGuidForServer()

Type: Function.

Purpose: Queries Active Directory™ for the GUID (Database UUID) of a given server.

Parameters:

STRING	*RemoteServer*
STRING	*RequestedServer*
INTEGER	*UseCreds*

Explanations:

RemoteServer

FQDN or Net name of the server to use as the source for the information.

RequestedServer

The Site and NetBIOS name of the server to resolve the GUID for in the format "Site\Server".

UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the GUID of the server as STRING. If unsuccessful, returns empty STRING.

GetObjectGuidForServer()

Type: Function.

Purpose: Queries Active Directory™ for the object GUID of a given server.

Parameters:

STRING	*RemoteServer*
STRING	*RequestedServer*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information.
RequestedServer
The single-part name of the server to resolve the object GUID for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the Object GUID of the server as STRING. If unsuccessful, returns empty STRING.

GetActiveDirectoryProperties()

Type: Function.

Purpose: Queries Active Directory™ for general configuration

Parameters:

STRING	*RemoteServer*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user
1 = use credentials supplied from SetUserCredentials()

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

Comments: Returns more data in the form of properties beginning with "AD".

ADStayOfExecution()

Purpose: Returns the period of time that a server attempts to replicate from a retired domain controller from the call GetActiveDirectoryProperties( ).

Parameters: None.

Return: Returns time in number of days as LONG.

ADSPNMappings()

Purpose: Returns the SPN mappings for the current Active Directory™.

Parameters: None.

Return: Returns mappings as STRING.

GetSiteLinkProperties()

Type: Function.

Purpose: Queries Active Directory™ for the configuration of a given Site Link.

Parameters:

STRING	*RemoteServer*
STRING	*SiteLinkName*
STRING	*SiteLinkType*
INTEGER	*UseCreds*

Explanations:

RemoteServer
    FQDN or Net name of the server to use as the source for the information.

SiteLinkName
    The textual name of the requested site Link.

SiteLinkType
    "IP" or "SMTP"

UseCreds (optional)
    0 (zero) or no value= use credentials of currently logged-on user.
    1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

Comments: Returns more data in the properties beginning with "SiteLink".

SiteLinkCost()

Purpose: Returns the cost associated with the most recent queried Site Link.

Parameters: None.

Return: The cost associated with the most recently queried Site Link as LONG. If unsuccessful, returns -1 as LONG.

SiteLinkOptions()

Purpose: Returns the options associated with the most recent queried SiteLink.

Parameters: None.

Return: The options associated with the most recently queried Site Link as STRING. If unsuccessful, returns empty STRING.

SiteLinkSiteCount()

Purpose: Returns the number of sites in the list associated with the most recent queried Site Link.

Parameters: None.

Return: If successful, returns the number of sites as LONG. If unsuccessful, returns -1 as LONG.

SiteLinkName()

Purpose: Returns the name associated with the most recent queried Site Link.

Parameters: None.

Return: If successful, returns name as STRING. If unsuccessful, returns empty as STRING.

SiteLinkReplInterval()

Purpose: Returns the replication interval associated with the most recent queried Site Link.

Parameters: None.

Return: If successful, returns interval as LONG. If unsuccessful, returns -1 as LONG.

GetSubnets()

Type: Function.

Purpose: Queries Active Directory™ for an enumeration and configuration of subnets.

Parameters:
STRING *RemoteServer*
INTEGER *UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.

1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns number of subnets as LONG. If unsuccessful, returns -1 (negative one) as LONG.

Comments: Returns more data in the attributes beginning with "Subnet".

SubnetName()

Purpose: Returns the name of the requested subnet in an enumeration from GetSubnets().

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of transports. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name as STRING. If unsuccessful, returns an empty STRING.

SubnetSiteObject()

Purpose: Returns the name of the requested siteObject for the subnet in an enumeration from GetSubnets().

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of transports. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the site object as STRING. If unsuccessful, returns an empty STRING

GetInterSiteTransports()

Type: Function.

Purpose: Queries Active Directory™ for the configuration of available Transports.

Parameters:

STRING RemoteServer
INTEGER UseCreds

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user
1 = use credentials supplied from SetUserCredentials()

Return: If successful, returns number of transports as LONG. If unsuccessful, returns -1 (negative one) as LONG.

Comments: Returns more data in the properties beginning with "Transport".

TransportName()

Purpose: Returns the name of the requested transport in an enumeration from GetInterSiteTransports().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of transports. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name as STRING. If unsuccessful, returns an empty STRING.

TransportAddress()

Purpose: Returns the address type of the requested transport in an enumeration from GetInterSiteTransports().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of transports. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the address type as STRING. If unsuccessful, returns an empty STRING.

TransportAddress()

Purpose: Returns the options of the requested transport in an enumeration from GetInterSiteTransports().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of transports. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the options as STRING. If unsuccessful, returns an empty STRING.

TransportDLLName()

Purpose: Returns the DLL name of the requested transport in an enumeration from GetInterSiteTransports().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of transports. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the DLL name as STRING. If unsuccessful, returns an empty STRING.

GetDSAProperties()

Type: Function.

Purpose: Queries Active Directory™ for the configuration of a particular domain controller's NTDS Settings.

Parameters:
STRING *RemoteServer*
STRING *RequestedServer*
INTEGER *UseCreds*

Explanations:

RemoteServer
FQDN or Net name of the server to use as the source for the information
RequestedServer
The server whose information should be read in the form of Site\Server (for example Portland\SERVER1)
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user
1 = use credentials supplied from SetUserCredentials()

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

Comments: Returns more data in the properties beginning with "DSA".

DSABridgeHeadTransportCount()

Purpose: Returns the number of BridgeHead transports in the list associated with the most recent queried DSA object.

Parameters: None.

Return: If successful, returns the number of transports as LONG. If unsuccessful, returns -1 as LONG.

DSABridgeHeadTransport()

Purpose: Returns the name of each Bridgehead transport present in a DSA's properties.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of bridgeheads. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the BridgeHead transport as STRING. If unsuccessful, returns an empty STRING.

DSASchemaLocation()

Purpose: Returns the path to the Schema associated with the most recent queried DSA Object.

Parameters: None.

Return: If successful, returns path to the Schema as STRING (may be empty). If unsuccessful, returns empty STRING.

DSAObjectGUID()

Purpose: Returns the object GUID of the server associated with the most recent queried DSA Object

Parameters: None.

Return: If successful, returns the object GUID of the server as STRING. If unsuccessful, returns empty STRING.

DSAComputerPath()

Purpose: Returns the path to the computer object associated with the most recent queried DSA Object.

Parameters: None.

Return: If successful, returns path to computer object as STRING (may be empty). If unsuccessful, returns empty STRING.

DSAInvocationID()

Purpose: Returns the invocation GUID of the server associated with the most recent queried DSA Object

Parameters: None.

Return: If successful, returns the GUID of the server as STRING. If unsuccessful, returns empty STRING.

DSADNSHostName()

Purpose: Returns the host name of the computer object associated with the most recent queried DSA Object.

Parameters: None.

Return: If successful, returns the host name of the computer object as STRING (may be empty). If unsuccessful, returns empty STRING.

DSAMailAddress()

Purpose: Returns the mail address of the computer object associated with the most recent queried DSA Object.

Parameters: None.

Return: If successful, returns mailAddress as STRING (might be empty). If unsuccessful, returns empty STRING.

DSAOptions()

Purpose: Returns the options associated with the most recent queried DSA Object.

Parameters: None.

Return: If successful, returns flags as STRING (may be empty). If unsuccessful, returns empty STRING.

GetSiteProperties()

Type: Function.

Purpose: Queries Active Directory™ for the configuration of a given site.

Parameters:

STRING	*RemoteServer*
STRING	*SiteName*
INTEGER	*UseCreds*

Explanations:

RemoteServer
FQDN or Net name of the server to use as the source for the information.

SiteName
The textual name of the requested site.

UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

Comments: Returns more data in the properties beginning with "Site".

SiteOptions()

Purpose: Returns the options associated with the most recent queried site.

Parameters: None.

Return: If successful, returns options as STRING. If unsuccessful, returns empty STRING.

SiteTopologyGenerator()

Purpose: Returns the generator associated with the most recent queried site.

Parameters: None.

Return: If successful, returns generator as STRING. If unsuccessful, returns empty STRING.

SiteTopologyFailover()

Purpose: Returns the failover time associated with the most recent queried site.

Parameters: None.

Return: If successful, returns the failover as STRING. If unsuccessful, returns empty STRING.

SiteTopologyRenew()

Purpose: Returns the renewal time associated with the most recent queried site.

Parameters: None.

Return: If successful, returns renewal time as string. If unsuccessful, returns empty STRING.

SiteLinkSiteList()

Purpose: Returns the name of each site present in a given site link.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of site links. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the site as STRING. If unsuccessful, returns an empty STRING.

GetDomainNamingFSMO()

Type: Function.

Purpose: Queries the server specified for what it believes is the Domain Tree Management FSMO Owner.

Parameters:

STRING	*RemoteServer*
INTEGER	*UseCreds*

Explanations:

RemoteServer
    FQDN or Net name of the server to use as the source for the information.
UseCreds (optional)
    0 (zero) or no value= use credentials of currently logged-on user.
    1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the name of the FSMO Owner as STRING. If unsuccessful, returns empty STRING.

GetInfrastructureFSMO()

Type: Function.

Purpose: Queries the server specified for what it believes is the Infrastructure FSMO Owner.

Parameters:

STRING	*RemoteServer*
STRING	*DomainID*
INTEGER	*UseCreds*

Explanations:

RemoteServer
    FQDN or Net name of the server to use as the source for the information.
DomainID
    Domain name in LDAP or DNS format.
UseCreds (optional)
    0 (zero) or no value= use credentials of currently logged-on user.
    1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the name of the FSMO Owner as STRING in "Site\Server" format. If unsuccessful, returns empty STRING.

GetRidPoolFSMO()

Type: Function.

Purpose: Queries the server specified for what it believes is the RID Pool FSMO Owner.

Parameters:

STRING	*RemoteServer*
STRING	*DomainID*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information.
DomainID
Domain name in LDAP or DNS format.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the name of the FSMO Owner as STRING in "Site\Server" format. If unsuccessful, returns empty STRING.

GetPDCFSMO()

Type: Function.

Purpose: Queries the server specified for what it believes is the PDC FSMO Owner.

Parameters:

STRING	*RemoteServer*
STRING	*DomainID*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information.
DomainID
Domain name in LDAP or DNS format.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the name of the FSMO Owner as STRING in "Site\Server" format. If unsuccessful, returns empty STRING.

GetSchemaFSMO()

Type: Function.

Purpose: Queries the server specified for what it believes is the Schema FSMO Owner.

Parameters:

STRING	*RemoteServer*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the name of the FSMO Owner as STRING in "Site\Server" format. If unsuccessful, returns empty STRING.

GetRegistryData()

Type: Function.

Purpose: Retrieves a registry value from a remote or local computer.

Parameters:

STRING	*ServerName*
STRING	*BaseKey*
STRING	*SubKey*
STRING	*Value*

Explanations:
ServerName
Name of the server to connect to. This can be NULL or \\computername. Note that if a computer name is specified, must be prefixed with "\\".
BaseKey
HKEY_LOCAL_MACHINE or HKEY_CURRENT_USER
SubKey
The path to the value (for example, System\CurrentControlSet).
Value
The name of the value to retrieve the data for.

Return: If successful, returns the data (REG_DWORD or REG_SZ) as STRING. If unsuccessful, returns empty STRING.

ReplicaSync()

Type: Function.

Purpose: Initiates replication between two domain controllers given a naming context.

Parameters:

STRING	*RemoteServer*
STRING	*NamingContext*
STRING	*SourceReplica*
INTEGER	*UseFlags*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN, IP Address, or Net name of the server to use as the source for the information.
NamingContext
The naming context in LDAP format to synchronize (for example, "cn=configuration,dc=Microsoft,dc=com").
SourceReplica
FQDN or NetBIOS name of the server that changes will be synchronized from.
UseFlags (optional)
0 (zero) or no value = flags set by SetReplicaSyncFlags() will not be used.
1 = flags set by SetReplicaSyncFlags() will be used.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

ReplicaSyncAll()

Type: Function.

Purpose: Initiates replication from all domain controllers given a naming context.

Parameters:

STRING	*RemoteServer*
STRING	*NamingContext*
INTEGER	*UseFlags*
INTEGER	*UseCreds*

Explanations:
RemoteServer
FQDN or Net name of the server to use as the source for the information
NamingContext

The naming context in LDAP format to synchronize (for example, "cn=configuration,dc=Microsoft,dc=com").
UseFlags (optional)
0 (zero) or no value = flags set by SetReplicaSyncAllFlags() will not be used.
1 = flags set by SetReplicaSyncAllFlags() will be used.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

GetGPOVersion()

Type: Function.

Purpose: Determines the Group Policy object version out of Active Directory on a particular domain controller.

Parameters:

STRING	*DomainName*
STRING	*DCName*
STRING	*GPO*
INTEGER	*UseCreds*

Explanations:
DomainName
The name of the domain where the Group Policy object resides in DNS format (for example, *Microsoft*.com).
DCName
FQDN or NetBIOS name of the domain controller where the Group Policy object should be read.
GPO
The GUID of the Group Policy object (for example, 000000-0000-0000-0000-000000000000)
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the version number as LONG. If unsuccessful, returns -1 as LONG

GetGPOSysVolVersion()

Type: Function.

Purpose: Determines the Group Policy object Version out of SYSVOL on a particular domain controller.

Parameters:
STRING *DomainName*
STRING *DCName*
STRING *GPO*
INTEGER *UseCreds*

Explanations:
DomainName
The name of the domain where the Group Policy object resides in DNS format (for example, *Microsoft*.com).
DCName
FQDN or NetBIOS name of the domain controller where the Group Policy object should be read.
GPO
The GUID of the Group Policy object (for example, 000000-0000-0000-0000-000000000000).
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the version number as LONG. If unsuccessful, returns -1 as LONG.

GetGPOs()

Type: Function.

Purpose: Enumerates the Group Policy objects for a given domain controller.

Parameters:
STRING *DomainName*
STRING *DCName*
INTEGER *UseCreds*

Explanations:
DomainName
The name of the domain to use in the query in DNS format (for example, *Microsoft*.com).
DCName
FQDN or NetBIOS name of the domain controller where the Group Policy object should be read.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.

1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of Group Policy objects as LONG. If unsuccessful, returns -1 as LONG.

Comments: The resultant Group Policy objects can be found by enumerating GPOName(*i*) and GPOGuid(*i*) where *i* is 1 to the number of Group Policy objects returned from this function. This function also automatically enumerates the version numbers in Active Directory and in SYSVOL and can be found in the properties GPOVersion() and GPOSysVolVersion().

GPOName()

Purpose: Returns the friendly name of the Group Policy object requested in an enumeration of Group Policy objects returned from the GetGPOs() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of Group Policy objects. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the friendly name of the Group Policy object as STRING. If unsuccessful, returns an empty STRING.

GPOGuid()

Purpose: Returns the GUID of the Group Policy object requested in an enumeration of Group Policy objects returned from the GetGPOs() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of Group Policy objects. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the GUID of the Group Policy object as STRING. If unsuccessful, returns an empty STRING.

GPOVersion()

Purpose: Returns the version of the Group Policy object requested in an enumeration of Group Policy objects returned from the GetGPOs() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of Group Policy objects. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the version of the Group Policy object as STRING. If unsuccessful, returns an empty STRING.

GPOSysVolVersion()

Purpose: Returns the version of the Group Policy object in SYSVOL requested in an enumeration of Group Policy objects returned from the GetGPOs() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of Group Policy objects. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the SYSVOL version of the Group Policy object as STRING. If unsuccessful, returns an empty STRING

GetHighestCommittedUSN()

Type: Function.

Purpose: Given a domain controller, it receives the highest USN committed for the server.

Parameters:
STRING   *ServerName*
INTEGER   *UseCreds*

Explanations:
ServerName
    The FQDN or NetBIOS name of the server to remote this call to in order to determine its high USN.
UseCreds (optional).

0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the USN as VARIANT. If unsuccessful, returns -1 as VARIANT.

DCAddress()

Purpose: Returns the IP address or NetBIOS name of the domain controller that was found from a DsGetDcName call().

Parameters: None.

Return: If successful, returns the IP Address/name as STRING. If unsuccessful, returns an empty STRING.

DCAddressType()

Purpose: Returns the type of address returned in DCAddress() that was found from a DsGetDcName call().

Parameters: None.

Return: If successful, returns DS_INET_ADDRESS or DS_NETBIOS_ADDRESS as STRING. If unsuccessful, returns an empty STRING.

TreeName()

Purpose: Returns the tree name of the domain controller of the domain controller that was found from a DsGetDcName call().

Parameters: None.

Return: If successful, returns the tree name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

ReturnedFlags()

Purpose: Returns the flags from a DsGetDcName() call.

Parameters:
STRING *Flags*

Explanations:
Flags
A comma delimited string of flags as documented below:

DS_PDC_FLAG
DS_GC_FLAG
DS_DS_FLAG
DS_KDC_FLAG
DS_TIMESERV_FLAG
DS_CLOSEST_FLAG
DS_WRITABLE_FLAG
DS_GOOD_TIMESERV_FLAG
DS_PING_FLAGS
DS_DNS_CONTROLLER_FLAG
DS_DNS_DOMAIN_FLAG
DS_DNS_FOREST_FLAG

Return: If successful, returns a CR delimited list of flags as STRING. If unsuccessful or no flags to return, returns empty STRING.

DCSiteName()

Purpose: Returns the name of the Site of the domain controller that was found from a DsGetDcName call().

Parameters: None.

Return: If successful, returns the name of the site of the domain controller as STRING. If unsuccessful, returns an empty STRING.

DCName()

Purpose: Returns the name of the domain controller that was found from a DsGetDcName call().

Parameters: None.

Return: If successful, returns the name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

ClientSiteName()

Purpose: Returns the name of the site the client is in resulting from a DsGetDcName call().

Parameters: None.

Return: If successful, returns the site name as STRING. If unsuccessful, returns an empty STRING.

SetReplicaSyncFlags()

Purpose: Allows the caller to set the flags to use in a ReplicaSync() call.

Parameters:
STRING *Flags*

Explanations:
Flags

A comma delimited string of flags as documented below:

DS_REPSYNC_ASYNCHRONOUS_OPERATION
DS_REPSYNC_WRITEABLE
DS_REPSYNC_PERIODIC
DS_REPSYNC_INTERSITE_MESSAGING
DS_REPSYNC_ALL_SOURCES
DS_REPSYNC_FULL
DS_REPSYNC_URGENT
DS_REPSYNC_NO_DISCARD
DS_REPSYNC_FORCE

Return: Does not return a value.

SetReplicaSyncAllFlags()

Purpose: Allows the caller to set the flags to use in a ReplicaSyncAll() call.

Parameters:
STRING *Flags*

Explanations:
Flags

A comma delimited string of flags as documented below:

DS_REPSYNCALL_NO_OPTIONS
DS_REPSYNCALL_ABORT_IF_SERVER_UNAVAILABLE
DS_REPSYNCALL_SYNC_ADJACENT_SERVERS_ONLY
DS_REPSYNCALL_ID_SERVERS_BY_DN
DS_REPSYNCALL_DO_NOT_SYNC
DS_REPSYNCALL_SKIP_INITIAL_CHECK
DS_REPSYNCALL_PUSH_CHANGES_OUTWARD
DS_REPSYNCALL_CROSS_SITE_BOUNDARIES

Return: Does not return a value.

SetDsGetDcNameFlags()

Purpose: Allows the caller to set the flags to use in a DsGetDcName() call.

Parameters:
STRING *Flags*

Explanations:
Flags
A comma delimited string of flags as documented below:
DS_FORCE_REDISCOVERY
DS_DIRECTORY_SERVICE_REQUIRED
DS_DIRECTORY_SERVICE_PREFERRED
DS_GC_SERVER_REQUIRED
DS_PDC_REQUIRED
DS_IP_REQUIRED
DS_KDC_REQUIRED
DS_TIMESERV_REQUIRED
DS_WRITABLE_REQUIRED
DS_GOOD_TIMESERV_PREFERRED
DS_AVOID_SELF
DS_IS_FLAT_NAME
DS_IS_DNS_NAME
DS_RETURN_DNS_NAME
DS_RETURN_FLAT_NAME

Return: Does not return a value.

LastCallResult()

Purpose: Returns the error code from the last operation

Parameters: None.

Return: Returns the error number or:
0 (zero) for success.
-1 for failure.

LastErrorText()

Purpose: Returns the error text from the last operation

Parameters: None.

Return: Returns the error code (may be success) from the last operation as STRING.

GetReplicationUSNState()

Type: Function.

Purpose: Reads the data on a given domain controller (the last originating write expressed as a USN for each replication partner) for a given directory partition and returns the data to multiple properties: ReplPartnerGuid(),ReplPartnerName(),ReplPartnerInConflict(), and ReplPartnerUSN().

Parameters:

STRING	*ServerName*
STRING	*NamingContext*
INTEGER	*IsPartialNC*
INTEGER	*UseCreds*

Explanations:

ServerName

FQDN, IpAddress, or NetBIOS name of the server to determine replication status for.

NamingContext

The naming context in LDAP format to get the status of (for example, cn=configuration,dc=Microsoft,dc=com).

IsPartialNC (optional)

0 (zero) or no value = the NamingContext will not be queried as a partial replica.

1 = the function will expect that this naming context is a partial replica.

UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.

1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of replicas as LONG. If unsuccessful, returns -1 as LONG.

Comments: More data is in the properties beginning with "ReplPartner".

GetNamingContexts()

Type: Function.

Purpose: Reads the RootDSE attributes to determine the naming contexts for the server passed in.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:

ServerName

FQDN or NetBIOS name of the server to determine the naming contexts for.

UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of contexts as LONG. If unsuccessful, returns 0 (zero) as LONG.

Comments: The property NamingContextName(*i*) is populated with the returned naming contexts where *i* is the number of contexts returned by this function.

GetPartialNamingContexts()

Type: Function.

Purpose: Reads the RootDSE attributes to determine the partial naming contexts for the server passed in.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the server to determine the naming contexts for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of contexts as LONG. If unsuccessful, returns -1 as LONG.

Comments: The property NamingContextName(*i*) is populated with the returned naming contexts where *i* is the number of contexts returned by this function.

NamingContextName()

Purpose: Returns the name of the context requested in an enumeration from GetNamingContext().

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of contexts. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the context as STRING. If unsuccessful, returns an empty STRING.

GetWritableNCsForServer()

Type: Function.

Purpose: Reads the RootDSE attributes to determine the partial naming contexts for the server passed in.

Parameters:

STRING	*RemoteServer*
STRING	*ServerName*
STRING	*SiteName*
INTEGER	*UseCreds*

Explanations:

RemoteServer
The FQDN, IpAddress, or NetBIOS name of the server to use as the source of the information.
ServerName
The NetBIOS name of the server to determine the writable NCs for.
SiteName
The site name of the server defined by ServerName
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of contexts as LONG. If unsuccessful, returns -1 as LONG.

Comments: The property NamingContextName(*i*) is populated with the returned naming contexts where *i* is the number of contexts returned by this function.

GetTrustRelationships()

Type: Function.

Purpose: Reads the trust relationships for the domain that the domain controller connected to resides in.

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:

ServerName
FQDN or NetBIOS name of the server to determine the trusts for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of trusts as LONG. If unsuccessful, returns -1 as LONG.

Comments: The properties TrustXXX is populated with the returned attribute data where *i* is the number of trusts returned by this function

TrustNetBIOSName()

Purpose: Returns the NetBIOS name of the trusted domain in an enumeration from GetTrustRelationships().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of trusts. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the trust NetBIOS name as STRING. If unsuccessful, returns an empty STRING.

TrustDomainName()

Purpose: Returns the domain name of the trusted domain in an enumeration from GetTrustRelationships().

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of trusts. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the trust domain name as STRING. If unsuccessful, returns an empty STRING.

TrustDirection()

Purpose: Returns the direction of the trusted domain in an enumeration from GetTrustRelationships().

Parameters:

LONG	*Index*

Explanations:

Index

The item number in the enumerated list of trusts. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the trust direction as STRING. If unsuccessful, returns an empty STRING.

TrustType()

Purpose: Returns the trust type in an enumeration from GetTrustRelationships().

Parameters:

LONG	*Index*

Explanations:

Index

The item number in the enumerated list of trusts. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the trust type as STRING. If unsuccessful, returns an empty STRING.

GetDirectPartners()

Type: Function.

Purpose: Reads the data on a given domain controller and returns the data to multiple properties: DirectPartnerXXXX() series of properties.

Parameters:

STRING	*ServerName*
STRING	*NamingContext*
INTEGER	*IsPartialNC*
INTEGER	*UseCreds*

Explanations:

ServerName

FQDN or NetBIOS name of the server to determine replication status for

NamingContext

The naming context in LDAP format to get the status of (for example, cn=configuration,dc=Microsoft,dc=com).
IsPartialNC (optional)
0 (zero) or no value = the NamingContext will not be queried as a partial replica
1 = the function will expect that this naming context is a partial replica.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of replicas as LONG. If unsuccessful, returns -1 as LONG.

Comments: More data is available in the properties beginning with "DirectPartner".

GetChangeNotifications()

Type: Function.

Purpose: Reads the data corresponding to which partners are notified when changes in the Active Directory occur on a given domain controller and returns the data to multiple properties: NotificationPartnerXXXX() series of properties

Parameters:

STRING	*ServerName*
STRING	*NamingContext*
INTEGER	*IsPartialNC*
INTEGER	*UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the server to determine replication status for
NamingContext
The naming context in LDAP format to get the status of (for example, cn=configuration,dc=Microsoft,dc=com).
IsPartialNC (optional)
0 (zero) or no value = the NamingContext will not be queried as a partial replica
1 = the function will expect that this naming context is a partial replica.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of replicas as LONG. If unsuccessful, returns -1 as LONG.

Comments: More data is available in the properties beginning with "NotificationPartner".

DsGetDcList()

Type: Function.

Purpose: Gets the list of domain controllers and returns it in properties starting with "DCListEntry..".

Parameters:

STRING	ServerName
STRING	DomainName
LONG	InfoLevel (optional)
INTEGER	UseCreds (optional)

Explanations:

ServerName (optional)

FQDN or NetBIOS name of server to source data from.

DomainName

Domain Name to enumerate domain controllers for in DNS format.

InfoLevel (optional)

1 = Basic

UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.

1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of domain controllers as LONG. If unsuccessful, returns -1 as LONG.

Comments: The results of this function are written to DCListEntryXXXX() properties.

DCListEntryNetBiosName()

Purpose: Returns the NetBIOS name of the domain controller in the list of enumerated DCS resulting from the DsGetDcList() function.

Parameters:

LONG	*Index*

Explanations:

Index

The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the NetBIOS (Network Basic Input/Output System) name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

DCListEntryDnsHostName()

Purpose: Returns the DNS host name of the domain controller in the list of enumerated DCS resulting from the DsGetDcList() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the DNS host name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

DCListEntrySiteName()

Purpose: Returns the site name of the domain controller in the list of enumerated domain controllers resulting from the DsGetDcList() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the site name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

DCListEntryComputerObject()

Purpose: Returns the computer object name of the domain controller in the list of enumerated domain controllers resulting from the DsGetDcList() function.

Parameters:
LONG Index

Explanations:
Index
The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded

Return: If successful, returns the computer object name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

DCListEntryServerObject()

Purpose: Returns the server object name of the domain controller in the list of enumerated domain controllers resulting from the DsGetDcList() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the server object name of the domain controller as STRING. If unsuccessful, returns an empty STRING.

DCListEntryIsPDC()

Purpose: Returns the PDC state of the domain controller in the list of enumerated domain controllers. resulting from the DsGetDcList() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "YES" or "NO" as STRING. If unsuccessful, returns an empty STRING.

DCListEntryHasDS()

Purpose: Returns if the domain controller has AD installed in the list of enumerated domain controllers resulting from the DsGetDcList() function.

Parameters:
LONG   *Index*

Explanations:
Index

The item number in the enumerated list of domain controllers. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns "YES" or "NO" as STRING. If unsuccessful, returns an empty STRING

GetMetaData()

Type: Function.

Purpose: Reads the replPropertyMetaData attribute and extracts the data for each attribute present.

Parameters:

STRING	*ServerName*
STRING	*ObjectDN*
INTEGER	*UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the server to be used as the source of information.
ObjectDN
In LDAP format, the object that the metadata should be enumerated for.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of attributes as LONG. If unsuccessful, returns -1 as LONG.

Comments: The results of this function are written to MetaDataXXXX() properties.

GetMetaDataDifferences()

Type: Function.

Purpose: Given a server, naming context, and USN to start from, determines objects that have changed.

Parameters:

STRING	*ServerName*
LONG	*LocalUSN*
STRING	*NamingContext*
INTEGER	*UseCreds*

Explanations:

ServerName
FQDN or NetBIOS name of the server to be used as the source of information
LocalUSN
Update Sequence Number to use in the query - objects returned will have USNs higher than this number.
NamingContext
The naming context in LDAP format in which to search for changed objects.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of objects as LONG. If unsuccessful, returns -1 as LONG.

Comments: The results of this function are written to MetaDataDifferencesXXXX() properties.

MetaDataDifferencesCount()

Purpose: Returns the number of objects that have changed as a result of the last GetMetaDataDifferences() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the number of changed objects as LONG. If unsuccessful, returns -1 as LONG.

MetaDataDifferencesObjectDN()

Purpose: Returns the distinguished name of the changed object (specified by the index) as a result of the last GetMetaDataDifferences() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the distinguished name of the changed object as STRING. If unsuccessful, returns empty STRING.

MetaDataDifferencesAttribute()

Purpose: Returns the name of the attribute in the changed object (specified by the index) as a result of the last GetMetaDataDifferences() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the attribute in the changed object as STRING. If unsuccessful, returns empty STRING.

MetaDataDifferencesLastWriteTime()

Purpose: Returns the last time the attribute was written as a result of the last GetMetaDataDifferences() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the last write time as STRING. If unsuccessful, returns empty STRING.

MetaDataDifferencesOrigServer()

Purpose: Returns the domain controller that last modified the attribute as a result of the last GetMetaDataDifferences() function.

Parameters:
LONG   *Index*

Explanations:
Index

The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the domain controller name as STRING. If unsuccessful, returns empty STRING.

MetaDataDifferencesOrigUSN()

Purpose: Returns the USN of the changed object on the domain controller that last modified the attribute as a result of the last GetMetaDataDifferences() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the USN as VARIANT. If unsuccessful, returns -1 as VARIANT.

MetaDataName()

Purpose: Returns the name of the requested attribute in an enumeration of metadata from the GetMetaData() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name of the attribute as STRING. If unsuccessful, returns an empty STRING.

MetaDataLastWriteTime()

Purpose: Returns the date/time of the requested attribute in an enumeration of metadata from the GetMetaData() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the time/date stamp as STRING. If unsuccessful, returns an empty STRING.

MetaDataLocalUSN()

Purpose: Returns the USN of the requested attribute in an enumeration of metadata from the GetMetaData() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the USN as VARIANT. If unsuccessful, returns -1 as VARIANT.

MetaDataServerName()

Purpose: Returns the originating server name of the requested attribute in an enumeration of metadata from the GetMetaData() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the originating server name of the attribute as STRING. If unsuccessful, returns an empty STRING.

MetaDataSourceUSN()

Purpose: Returns the originating server source USN of the requested attribute in an enumeration of metadata from the GetMetaData() function.

Parameters:

LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the originating server source USN of the attribute as VARIANT. If unsuccessful, returns –1 as VARIANT.

MetaDataVersionNumber()

Purpose: Returns the property version number of the requested attribute in an enumeration of metadata from the GetMetaData() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of attributes. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the property version number of the attribute as LONG. If unsuccessful, returns –1 as LONG.

TestBind()

Type: Function.

Purpose: Performs name resolution, connect, and LDAP bind to the passed in server

Parameters:
STRING *ServerName*
INTEGER *UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the remote server to bind to.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns TRUE. If unsuccessful, returns FALSE.

NetSendMessage()

Type: Function.

Purpose: Sends a message in a popup window to a network recipient.

Parameters:

STRING	*ToAddress*
STRING	*FromName*
STRING	*Message*

Explanations:
ToAddress
FQDN or NetBIOS name of the computer/user to send the message to.
FromName
Text name of the sender.
Message
Text Message.

Return: If successful, returns 0 (zero) as INTEGER. If unsuccessful, returns error code or 1 (one) as INTEGER.

GetReplicationUSNStateEx()

Type: Function.

Purpose: Reads the data on a given domain controller (the last originating write expressed as a USN for each replication partner) for a given directory partition and returns the data to multiple properties: ReplPartnerGuid(),ReplPartnerName(),ReplPartnerInConflict(), and ReplPartnerUSN().

Parameters:

STRING	*ServerName*
STRING	*NamingContext*
INTEGER	*IsPartialNC*
INTEGER	*UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the server to determine replication status for
NamingContext
The naming context in LDAP format to get the status of (for example, cn=configuration,dc=Microsoft,dc=com)
IsPartialNC (optional)
0 (zero) or no value = the NamingContext will not be queried as a partial replica.
1 = the function will expect that this naming context is a partial replica.
UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of replicas as LONG. If unsuccessful, returns -1 as LONG.

GetRDNForObject()

Type: Function.

Purpose: Extracts just the RDN of the passed-in parameter of distinguished name

Parameters:

STRING	*DN*

Explanations:
DN
Distinguished name (LDAP format) (for example, cn=myuser,cn=users,dc=Microsoft,dc=com).

Return: If successful, returns relative distinguished name as STRING. If unsuccessful, returns empty STRING.

GetDirectPartnersEx()

Type: Function.

Purpose: Reads the data on a given domain controller and returns the data to multiple properties: DirectPartnerXXXX() series of properties.

Parameters:

STRING	*ServerName*
STRING	*NamingContext*
INTEGER	*IsPartialNC*
INTEGER	*UseCreds*

Explanations:
ServerName
FQDN or NetBIOS name of the server to determine replication status for.
NamingContext
The naming context in LDAP format to get the status of (for example, cn=configuration,dc=Microsoft,dc=com).
IsPartialNC (optional)
0 (zero) or no value = the NamingContext will not be queried as a partial replica.
1 = the function will expect that this naming context is a partial replica.
UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns the number of replicas as LONG. If unsuccessful, returns -1 as LONG.

Comments: This uses an API to extract the data.

NotificationPartnerSyncFlags()

Purpose: Returns a comma delimited string of returned flags for the replication partner requested in an enumeration of partners returned from the GetChangeNotifications() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the flags as STRING. If unsuccessful, returns empty STRING.

NotificationPartnerName()

Purpose: Returns the name of the replication partner for the change notification requested in an enumeration of partners returned from the GetChangeNotifications() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the replication partner name as STRING. If unsuccessful, returns empty STRING.

NotificationPartnerObjectGuid()

Purpose: Returns the object guid of the replication partner for the change notification requested in an enumeration of partners returned from the GetChangeNotifications() function.

Parameters:
LONG   *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the object GUID as STRING. If unsuccessful, returns empty STRING.

NotificationPartnerAddedTime()

Purpose: Returns the date/time when the change notification was added for the replication partner requested in an enumeration of partners returned from the GetChangeNotifications() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the date/time as STRING. If unsuccessful, returns empty STRING.

NotificationPartnerTransport()

Purpose: Returns the name of the transport used for the replication partner for the change notification requested in an enumeration of partners returned from the GetChangeNotifications() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the transport name as STRING. If unsuccessful, returns empty STRING.

DirectPartnerGuid()

Purpose: Returns the GUID for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the GUID as STRING. If unsuccessful, returns empty STRING.

DirectPartnerName()

Purpose: Returns the name of the replication partner for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns partner name as STRING. If unsuccessful, returns empty STRING.

DirectPartnerTransportGuid()

Purpose: Returns the GUID of the transport used in replication for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns GUID as STRING. If unsuccessful, returns empty STRING.

DirectPartnerTransportDN()

Purpose: Returns the distinguished name of the transport used in replication for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns distinguished name as STRING. If unsuccessful, returns empty STRING.

DirectPartnerFailReason()

Purpose: Returns the fail reason for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the fail reason as LONG. If unsuccessful, returns -1 as LONG.

ReplPartnerInConflict()

Purpose: Returns whether or not the partner in Active Directory is in conflict in an enumeration of partners returned from the GetReplicationUSNState() or GetReplicationUSNStateEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If true, returns 1 (one) as INTEGER. If false, returns 0 (zero) as INTEGER. If unsuccessful, returns –1 as INTEGER.

DirectPartnerInConflict()

Purpose: Returns a status of whether or not the server name is in conflict for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If true, returns 1 (one) as INTEGER. If false, returns 0 (zero) as INTEGER. If unsuccessful, returns -1 as INTEGER.

DirectPartnerNumberFailures()

Purpose: Returns the number of failures for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns number of failures as LONG. If unsuccessful, returns -1 as LONG.

DirectPartnerHighOU()

Purpose: Returns the USN Object Update value for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded

Return: If successful, returns the OU USN as VARIANT. If unsuccessful, returns -1 as VARIANT.

DirectPartnerHighPU()

Purpose: Returns the USN Property Update value for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the PU USN as VARIANT. If unsuccessful, returns -1 as VARIANT.

DirectPartnerLastSuccessTime()

Purpose: Returns the date/time of last successful replication for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG   *Index*

Explanations:
Index
    The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns date/time as STRING. If unsuccessful, returns empty STRING.

DirectPartnerLastAttemptTime()

Purpose: Returns the date/time of last attempted replication for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG   *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns date/time as STRING. If unsuccessful, returns empty STRING.

DirectPartnerSyncFlags()

Purpose: Returns a comma delimited string of returned flags for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

DRS_WRIT_REP
DRS_INIT_SYNC
DRS_PER_SYNC
DRS_MAIL_REP
DRS_FULL_SYNC_IN_PROGRESS
DRS_FULL_SYNC_PACKET
DRS_DISABLE_AUTO_SYNC
DRS_DISABLE_PERIODIC_SYNC
DRS_USE_COMPRESSION
DRS_NEVER_NOTIFY

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns flags as STRING. If unsuccessful, returns empty STRING.

ReplPartnerGuid()

Purpose: Returns the GUID for the replication partner requested in an enumeration of partners returned from the GetReplicationUSNState() or GetReplicationUSNStateEx() function.

Parameters:
LONG *Index*

Explanations:
Index

The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the GUID as STRING. If unsuccessful, returns empty STRING.

ReplPartnerName()

Purpose: Returns the name of the replication partner in an enumeration of partners returned from the GetReplicationUSNState() or GetReplicationUSNStateEx() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the name as STRING. If unsuccessful, returns empty STRING.

ReplPartnerUSN()

Purpose: Returns the USN of the replication partner in an enumeration of partners returned from the GetReplicationUSNState() or GetReplicationUSNStateEx() function.

Parameters:

LONG	*Index*

Explanations:
Index
The item number in the enumerated list of partners. If nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the USN as VARIANT. If unsuccessful, returns -1 as VARIANT.

DsGetDcName()

Type: Function.

Purpose: DsGetDcName is the new Windows 2000 API that allows finding Windows NT 4.0 and Windows 2000 domain controllers with varying requirements and much more flexibility.

Parameters:
STRING *DomainName*
STRING *DomainController*
INTEGER *UseFlags*

Explanations:
DomainName (optional)
A string that specifies the name of the domain to query. This name can either be a DNS-style name (for example, Microsoft.com.) or a flat-style name (for example, Microsoft). If a DNS-style name is specified, the name may be specified with or without a trailing period.
DomainController (optional)
A string that specifies the name of the remote server to process this function. Leaving this blank specifies that the local machine processes the request.
UseFlags (optional)
0 (zero) or no value = Flags will not be used
1= specifies that the property SetDsGetDcNameFlags() has a series of strings separated by commas and enclosed in quotes that specifies the pre-requisites for the returned domain controller.

Return: If successful, returns 0 (zero) as LONG. If unsuccessful, returns -1 as LONG.

Comments: When calling this function, the domain name or domain controller must be specified (or both), but not specifying either is invalid.

DsGetSiteName()

Type: Function.

Purpose: DsGetSiteName is the new Windows 2000 API that allows finding out the site of the computer queried for.

Parameters:
STRING *ComputerName*

Explanations:
ComputerName (optional)
The name of the computer to determine the site for (remotes the API).

Return: If successful, returns Site Name as STRING. If unsuccessful, returns empty STRING.

GetServerFromGuid()

Type: Function.

Purpose: Resolves a server name to passed in GUID.

Parameters:
STRING *SearchServer*
STRING *RequestedGuid*
INTEGER *UseCreds*

Explanations:
SearchServer
FQDN or NetBIOS name of server to use as source of information.
RequestedGuid
GUID in format (000000-0000-0000-0000-00000000) of server object requested.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns server as STRING in the format "Site\Server". If unsuccessful, returns empty STRING.

GetObjectFromGuid()

Type: Function.

Purpose: Resolves an object in Active Directory™ from a passed-in parameter in GUID.

Parameters:
STRING *SearchServer*
STRING *RequestedGuid*
INTEGER *UseCreds*

Explanations:
SearchServer
FQDN or NetBIOS name of server to use as source of information.
RequestedGuid
GUID in format (00000000-0000-0000-0000-000000000000) of object requested.
UseCreds (optional)
0 (zero) or no value= use credentials of currently logged-on user.
1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns object distinguished name as STRING. If unsuccessful, returns empty STRING.

GetConfigurationNamingContext()

Type: Function.

Purpose: Determines the Configuration Naming context for a given domain controller.

Parameters:
STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:
ServerName
  FQDN or NetBIOS name of server to use as source of information.
UseCreds (optional)
  0 (zero) or no value= use credentials of currently logged-on user.
  1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns configuration naming context as STRING. If unsuccessful, returns empty STRING.

GetDefaultNamingContext()

Type: Function.

Purpose: Determines the default Naming context for a given domain controller.

Parameters:
STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:
ServerName
  FQDN or NetBIOS name of server to use as source of information.
UseCreds (optional)
  0 (zero) or no value= use credentials of currently logged-on user.
  1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns default naming context as STRING. If unsuccessful, returns empty STRING.

AddPerformanceCounter()

Type: Function
'
Purpose: Adds a counter to the list of counters to be queried when GetPeformanceData() is called.

Parameters:
STRING	*CounterName*

Explanations:
CounterName
Name of the counter in the format (for example): \NTDS\DRA Sync Full Remaining Objects.

Return: If successful, returns 0 (zero). If unsuccessful, returns 1 (one).

ClearPerformanceCounters()

Type: Function

Purpose: Clears the list of performance counters used in GetPeformanceData() calls.

Parameters: None.

Explanations:

Return: If successful, returns 0 (zero). If unsuccessful, returns 1 (one).

DirectPartnerObjectGuid()

Purpose: Returns the object GUID for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:
Index
The item number in the enumerated list of partners - if nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the object GUID as STRING. If unsuccessful, returns empty STRING.

DirectPartnerFailReasonText()

Purpose: Returns the text of the error message for the failure for the replication partner requested in an enumeration of partners returned from the GetDirectPartners() or GetDirectPartnersEx() function.

Parameters:
LONG *Index*

Explanations:

Index

The item number in the enumerated list of partners - if nothing is passed in or the list has not been enumerated, an error is recorded.

Return: If successful, returns the fail reason text as STRING. If unsuccessful, returns empty STRING.

ConvertErrorMsg()

Type: Function

Purpose: Converts an error number to message text.

Parameters:

STRING	*ErrorNumber*
INTEGER	*ErrorType*

Explanations:

ErrorNumber

The number of the system/networking error to convert (for example, 1722).

Return: If successful, returns the error text as STRING. If unsuccessful, returns empty STRING.

GetIPConfiguration()

Type: Function.

Purpose: Determines the IP Configuration (IP address, DNS servers, and so on) of the server specified

Parameters:

STRING	*ServerName*
INTEGER	*UseCreds*

Explanations:

ServerName

FQDN or NetBIOS name of server to use as source of information.

UseCreds (optional)

0 (zero) or no value= use credentials of currently logged-on user.

1 = use credentials supplied from SetUserCredentials().

Return: If successful, returns a STRING separated by line feeds with all applicable information. If unsuccessful, returns empty STRING.