enaio®

OPTIMAL SYSTEMS

Software Documentation

enaio® client – Programming Reference

Version 10.0

All software products as well as all related extension programs and additional functions are registered and/or in-use trademarks of OPTIMAL SYSTEMS GmbH, Berlin or its subsidiaries. They may only be used according to a valid licensing agreement. The software as well as related documentation are protected by German and international copyright law. Unauthorized duplication and sales is plagiarism and subject to criminal prosecution. All rights reserved, including reproduction, transmission, translation, and storage with/on all kinds of media. For all preconfigured test scenarios or demo presentations: All company and person names which occur in examples (screenshots) are fictional. Any resemblance to existing companies or persons is purely coincidental and unintentional.

Copyright 1992 – 2021 by OPTIMAL SYSTEMS GmbH
Cicerostraße 26
D-10709 Berlin

14.07.2021
Version 10.0

Contents

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Introduction

enaio® provides a COM interface to communicate with the client. This interface has been around since optimal_AS® 3.x. In early versions of optimal_AS® 3.x, the COM interface co-existed with the DDE interface, and before that, communication with the client was only possible with DDE.

COM stands for **C**omponent **O**bject **M**odel, which is a model introduced by Microsoft to allow communication between Windows applications. Frequently, the term **OLE** is also used, standing for **O**bject **L**inking and **E**mbedding. In this document the term **COM** is used exclusively. In this handbook you will find source code examples, information on the transition from the DDE interface of enaio® to the COM interface and a description of all available COM commands.

Handoff Files

Files are passed to the COM commands for insertion and querying. Files which hand over insert commands (**InsertIntoArchive**, **InsertIntoRegister**, **InsertIntoDocument**) contain the indexing of the objects to be created. Files used for request commands (**StartArchiveRequest**, **StartRegRequest**, **StartDocRequest**) contain request information i.e. the wanted indexing. Both file types additionally contain labels of the requested / to be inserted objects.

A handoff file for the **InsertIntoArchive** command may have the following structure:

```
[EINFÜGEN]
SCHRANK=Schrankname
FIELD1=field_value#1
FIELD2=field_value#2
...
FIELDn=field_value#n
```

A handoff file for the **StartArchiveRequest** command may have the following structure:

```
[ANFRAGE]
SCHRANK = cabinet name
KLAUSEL1=cabinet_name@field_name1=field_value1
KLAUSEL2= cabinet_name@field_name2=field_value2
...
...
KLAUSELn= ...
DATENFELDER=0(1)
DATAHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
```

Note: Handoff files differ from the COM commands. Please note the spelling of the COM commands.

Internal Names

The object names contained in the handoff files can be replaced by internal names. Thus percent sign needs to be placed before and after the internal name.

Example of a query file with internal names:

```
[ANFRAGE]
SCHRANK = %internal_cabinet_name%
KLAUSEL1=cabinet_name@%internal_field_name1%=field_value1
KLAUSEL2= %internal_cabinet_name%@field_name2=field_value2
```

Examples for an expression with internal object names:

```
AUSDRUCK1=%InboundDocument%@Date1^6^10.03.2000
```

Example for a logical expression:

```
KLAUSEL1=%InboundDocument%@%Created%=10.03.1997
```

Logical Expressions and Expressions

Query files which are passed to the commands **StartArchiveRequest**, **StartRegRequest,** and **StartDocRequest** result in a hit list of enaio® objects. These query files can contain logical expressions and expressions used to query against the indexing of objects.

Logical Expressions

Logical expressions follow this syntax:

```
KLAUSEL1=Object@Field=Value
```

A query lets you pass more than one logical expression. To do so, logical expressions need to be numbered sequentially, e.g.:

```
KLAUSEL1=...
KLAUSEL2=...
...
KLAUSEL#N#=
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

The logical linking of logical expressions is the operator **AND**, i.e. additional logical expressions allowed, limiting the hit list.

Example for a logical expression:

```
KLAUSEL1=InboundDocument@Created=10.03.1997
```

In this case **InboundDocument** is the object name and **template** is the name of a field of the object **InboundDocument**. All inbound documents with the value **10.03.1997** in the field **Created** are returned.

Note: Internal object names and field names can also be used.

Expressions

Expressions have a similar syntax. The following forms are possible:

1. Form `AUSDRUCK1=Object@DBField^OP^Value`
2. Form `AUSDRUCK1=Object@FieldNo^OP^Value`
3. Form `AUSDRUCK1=Object@DBField^OP^Value~BoolOP~FieldNo^OP^Value`

Object stands for the object name, **DBField** for the column name of the requested field in the database, **OP** stands for the relational operator, **Value** for the field value. **BoolOP** is a logical link of the different relational expressions. ^ separates relational values from operators and ~ separates Boolean expressions from Boolean operators.

A query lets you pass more than one logical expression. To do so, logical expressions need to be numbered sequentially, e.g.:

```
AUSDRUCK1=...
AUSDRUCK2=...
...
AUSDRUCKN=
```

The logical linking of expressions is the operator **AND**, i.e. additional expressions allowed, limiting the hit list.

Example 1 of an expression:

```
AUSDRUCK1=InboundDocument@Date1^6^10.03.2000
```

As in the logical expression, **inbound document** indicates the object name. **Date1** is the database column name of the field **Created**. The column name can be determined using the commands **GetObjectFields** and **GetObjectFieldsEx**. All inbound documents are returned that contain a value of greater or equal value to **10.03.1997** in the database field **Date1**.

Example 2 of an expression:

```
AUSDRUCK1=Account@1100^3^4711
```

All **Account** objects with an object ID of less than 4711 are returned.

Example 3 for an expression:

```
AUSDRUCK1=Account@1100^3^4711~0~1100^4^0815
```

All **Account** objects with an object ID between 4711 and 0815 are returned. Complex expressions with Boolean operators must be grouped with **(**and**)** to assure the unique meaning of the expression.

Examples for a location preset

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Example: Document in folder

```
AUSDRUCK1=Account@1130^1^815~0~1133^1^0
```

Using this expression in a document query only returns documents from directly under the folder level. **1130^1^815** defines the folder with the ID 815 as the parent folder. **1133^1^0** specifies that the searched registers have no parent register.

Example: Register in folder

```
AUSDRUCK1=Register@1121^1^815~0~1122^1^0
```

Using this expression in a register query only returns registers from directly under the folder level. **1121^1^815** defines the folder with the ID 815 as the parent folder. **1122^1^0** specifies that the searched registers have no parent register.

Example: 'register in register'

```
AUSDRUCK1=Register@1122^1^9911
```

This expression defines the parent register for a register query. Important: if the parent register is of a different type, the expression will contain the register type of the requested register as an object. This is a list of **field names/field numbers**, operators (**OP**) and Boolean operators (**BoolOP**)

Field name/field no.

Field name	Field no.	Description
Folders		
STAMM_ID	1000	Folder index
STAMM_TIME	1001	Creation time
STAMM_LINKS	1002	Number of links to this folder
Objects		
OBJECT_ID	1100	Document index
OBJECT_COUNT	1101	Number of document files
OBJECT_FLAGS	1102	Archiving status: 0 = archived 1 = archivable 2 = not archivable 4 = error in pages 8 = no pages 16 = archived but no slides 32 = not archived and not visible for archivist
OBJECT_AVID	1103	Name of archivist
OBJECT_AVDATE	1104	Archiving date
OBJECT_CRID	1105	Name of creator
OBJECT_CRDATE	1106	Attachment date
OBJECT_TIME	1107	Time stamp (date and time) of creation
OBJECT_MAIN	1108	Main document type
OBJECT_CO	1109	Secondary document type
OBJECT_MEDDOCID	1110	Media index of document
OBJECT_MEDDIAID	1111	Media index of slide
OBJECT_MEDDOCNA	1112	Path to document
OBJECT_MEDDIANA	1113	Path to slide
OBJECT_LINKS	1114	Number of links to this document
OBJECT_VERID	1115	Index of the original variant. Special values: 0=no variants, 1=this is the original document
OBJECT_LOCKUSER	1116	Index of the user who locked the document. Special values: 0=not locked, 1=swapped
Registers		
REG_ID	1120	Index of register
REG_STAID	1121	Index of the folder in which the register is located
REG_PARID	1122	Index of the register in which the register is located
Folder-Document relation		
SDSTA_ID	1130	Index of the folder in which the document is located
SDOBJ_ID	1131	Document index
SDOBJTYPE	1132	Object type (main type/sub type) of the document
SDREG_ID	1133	Index of the register in which the document is located
SDDEL	1134	Delete flag
SDTIME	1135	Insertion time
Portfolio Document Relation		
MDDEL	1140	Delete flag
MDTIME	1141	Creation time
MDMAP_ID	1142	Portfolio ID
MDSTA_ID	1143	Folder ID
MDOBJ_ID	1144	ID of the object
MDOBJTYPE	1145	Object type
MDMOD	1146	Main object type (if without type)
MDIN	1147	Entry time (not used)
MDOUT	1148	Exit time (not used)
MDCOUNT	1149	Number of pages (if without type)
MDSEND	1150	Sender (not used)
Portfolio		
MAPDEL	1160	Delete flag
MAPTIME	1161	Creation time
MAP_ID	1162	Portfolio ID
MAPCR_ID	1163	Name of creator
MAPCRDATE	1164	Attachment date
MAPRE_ID	1165	Name of recipient
MAPTHEME	1166	Theme of portfolio
MAPTYPE	1167	Portfolio type (not used)
User		
USER_ID	1170	User ID
USER_SUPER	1171	0=no supervisor; 1=supervisor (since 3.00 SP02 this is a combination of the rights flags -> see new rights system)
USER_USER	1172	User name
USER_PASSWORD	1173	Encoded user password
USER_STATION	1174	Workstation name

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

USER_LOGIN	1175	Login time stamp

OP List

OP	SQL operator
1	= (for exact searches, e.g. ^1^899999) LIKE (when searching for strings with placeholders, e.g. ^1^*.pdf)
2	!=
3	<
4	<<New configuration name>>
5	<=
6	>=
7	In
8	Ex

BoolOP List

BoolOP	SQL operator
0	AND
1	OR
2	NOT

Note: Internal object names can also be used.

Wildcard in Query Files

These wildcards can be used in query files:

Placeholder	Reference
#COMPUTER-GUID#	GUID of the logged-on user's computer
#COMPUTER-NAME#	Name of the logged-in user's computer
#COMPUTER-IP#	IP address of the logged-on user's computer
#ANLEGER#	Link to the basic parameter field 'Creator'
#ANLEGEDATUM#	Link to the basic parameter field 'Date of creation'
#ARCHIVAR#	Link to the basic parameter field 'Archivist'
#ARCHIVIERUNGSDATUM#	Link to the basic parameter field 'Archiving date'
#USER#	Name of the logged-on user
#OWNER#	Link to the basic parameter field 'Owner' which contains the owner's GUID
#DATE#	current date

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Source Code Examples

VB and VBA

There are two equivalent values to address the enaio® COM object under VB and VBA.

If enaio® client is processing time-consuming operations, the client application may not react to parallel queries using the COM interface as promptly as one would expect. The 1044 (WM_USER + 20) Windows message with WPARAM 30 (SendMessage API function) may be used to check the client application's system load beforehand. The value 1 is returned if the client load is high; otherwise the return value is 0. It is recommended to run the client load check before COM functions are called and, if the check returns 1, to take actions to reduce the load.

The following examples were created in VB version 6 and Office 97 VBA.

1. **Dimensioning a variable as 'New optimal_AS.Application'**.
   The enaio® object can only be addressed in such a way if the reference pointing to it was activated in the VB project. An important advantage of this method is that the enaio® COM object is known to the Intellisense technology of VB and VBA including all parameters, and that the Intellisense lists are displayed. In the following example, license registration is performed for the COLD module, and, based on the file **InsInA.txt,** a cabinet is created and immediately deleted again. At the end of the **Main** routine an error handling routine can be found which returns the error text along with the COM error number of enaio®.

```
Dim MyAX As New optimal_AS.Application
Dim HelpInt as Integer
Dim InsertFile as String, DeleteString as String, ErrorString as String
Dim AxID as Long, AxObjectType as Long
Sub Main()
 HelpInt = COMLicLogin("COL")
 If HelpInt <> 0 Then Goto FehlerBehandlung
 InsertFile = App.Path & "\InsInA.txt"
 HelpInt = COMInsertArc(InsertFile)
 If HelpInt <> 0 Then Goto FehlerBehandlung
 DeleteString = CStr(AxID) & "," & CStr(AxObjectType)
 HelpInt = COMDelArc(DeleteString)
 If HelpInt <> 0 Then Goto FehlerBehandlung
 Exit Sub
Fehlerbehandlung:
 ErrorString = MyAX.GetLastError
 msgbox ErrorString
 End
End Sub

Function COMLicLogin(LicStr as String) as Integer
 COMLicLogin = MyAX.LicLogin(LicStr)
End Function

Function COMInsertArc(File as String) as Integer
 COMInsertArc = MyAX.InsertIntoArchive(File, AxID, AxObjectType)
End Function

Function COMDelArc(DelStr as String) as Integer
 COMDelArc = MyAX.DeleteFromArchive(DelStr)
End Function
```

2. **Creating an object with the 'CreateObject' command**.

   This method can also be used with VB script; not the first method as project references are required which are not possible. The disadvantage of this method is that VB and VBA do not provide IntelliSense support.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

```
Dim MyAX As Object
....
rest like in the previous example
....
Sub Main()
     Set MyAX = CreateObject("Optimal_AS.Application")
     ....
     Rest wie im vorigen Beispiel
     ....
end Sub
Function COMLicLogin(LicStr as String)
     COMLicLogin = MyAX.LicLogin(LicStr)
End Function
     ....
     Rest wie im vorigen Beispiel
     ....
```

Note: As there are already many VB projects with COM integration, the enaio® COM object is still called **optimal_AS** for compatibility reasons. The **CreateObject** method is recommended when creating an optimal_AS object. Internally VB saves the GUID of the registered object. If you work with project references and import a new version of enaio®, it may happen that the new version creates a new object GUID. As a result, the object cannot be created and the program quits with an error message.

In order to use **IntelliSense** technology in VB projects, you can use both methods during development, i.e. project reference with **New** dimensioning of the object, and creating the object with **CreateObject** in the source code. Having finished the development, comment out the **New** dimensioning.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Object Main Types

For some COM commands a document main type must be specified. The following settings are possible.

Main type no.	Main type name
1	X-Document (grayscale image)
2	D-Document (B/W image)
3	P-Document (color image)
4	W-Document (Windows document)
5	M-Document (video document)
6	E-Document (E-mail)
7	XML-Document (XML)
8	Container document

All COM Commands

ActivateApp

Definition	ActivateApp (short nShow)		
Type	Method		
Description	Displays the application window or hides it		
Parameter	**nShow**	**1**	Show
		0	Hide
Return value			
Comment			
Options			
Example			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

AdjustRetention

Definition	AdjustRetention	(long	lObjectID,
		long	lObjectType,
		Variant	varRetentionDate)

Type Method

Description Adjusts the retention date to the scheduled retention date.

Parameters

lObjectID	object ID
lObjectType	object type
varRetentionDate	the adjusted retention date is returned in the format **YYYY/MM/DD**

Return value

0	retention date successfully adjusted
-1	retention date could not be adjusted
-7	object type unknown
-22	an object with the specified ID does not exist
-38	invalid document type (a folder or register type was specified)
-40	invalid parameter passed
-112	the specified object has not been archived (yet).

Comment This function can only be used to change already archived documents. Error texts can be determined using GetLastError().

Options

Example

```
Dim a As Object
Dim retDate

Set a = CreateObject("optimal_as.application")

a.AdjustRetention lObjectID, lObjectType, retDate
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

AppBrowserSession

Definition

```
[AppBrowserSession]
URL=file:///C:/test1.html
MODE=0
TIMER=30
```

Type

Description

Provides a Chromium browser session which can be used as a connection between the client and a Web server. In the client you can then trigger actions using the JavaScript interface of the browser. If a URL is configured, it is already loaded when starting the client. Use the as.cfg to configure the browser session.

Parameters

URL	URL to load. If this is not specified, no action is triggered. Default: not specified.
MODE	Specifies the situation in which to load the given URL. MODE=1 -> When switching context MODE=0 -> No action
TIMER	Specifies (in seconds) when to load the given URL. Valid values: 0 to 3600. Default: TIMER=0 -> No timer.

Return value

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ApplicationLogin

Definition	ApplicationLogin (String strUser)
Type	Method
Description	Allows a new user to log in.
Parameter	**strUser** string with user name and password, separated by a '@'
Return value	**1** no error
Comment	If the **DIALOG** string is passed, in any case the login dialog will open. This is only available if password verification is not handled with **Novell NetWare** but with a user dialog of the archiving system.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CalcFileDigest

Definition CalcFileDigest (String FileName, VARIANT* vRetDigest)

Type Method

Description Determines the hash value (digest) of the specified file

Parameters

FileName	the file name
vRetDigest	the determined hash value (digest)

Return value

0	no error
-1	hash value could not be determined
-32	the specified file does not exist

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CheckInDocument

Definition	CheckInDocument	(long long String	lDocID, lDocType, strSourcePath)

Type Method

Description Checks in again a document that was checked out for editing.

Parameter

lDocID	document ID
lDocType	document type
strSourcePath	path (without file name) to the file to be checked in

Return value

0	no error
-7	unknown document type
-53	document was not checked out
-54	document was not checked out for external editing
-55	file could not be copied to cache directory
-56	document name could not be determined in cache
-57	source file does not exist

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CheckLicence

Definition CheckLicence (String strModuleName)

Type Method

Description Verifies if a specific module is licensed for the current workstation

Parameter **strModuleName** Name of the module

Return value

0	is licensed
-1	is not licensed

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CheckObjectAccess

Definition

CheckObjectAccess (long lObjectID,
long lObjectType,
long lDesiredAccess,
long lFlags,
short nShowError)

Type

Method

Description

Verifies access rights for the specified object

Parameter

lObjectID object ID

lObjectType object type

lDesiredAccess
- **0** read index data
- **1** write index data
- **2** delete object
- **3** output object (e.g. print)
- **4** write object
- **5** check-out status
- **6** archive status
- **7** indicates whether the object is in the workflow tray (return value =**1**) or not (**0**)
- **8** indicates whether the object was marked for deletion. Return: **0**, object not marked for deletion
  1 object marked for deletion
  -22 unknown object.

lFlags since 4.20 SpII the value is set to **1**, in order to request the correct user rights, even when the security system is disregarded for this object.

nShowError if not equal to **0** an error dialog is displayed

Return value
- **0** user has no access right
- **1** user has access right
- **-1** unknown right
- **-7** document type unknown
- **-22** unknown object
- **-46** unknown user
- **-51** document contains no files
- **-52** object already archived
- **-58** document already loaned
- **-59** the document was checked out by another user for editing
- **-60** object is not a W-Document

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CheckOutDocument

Definition	CheckOutDocument	(long	lDocID,
		long	lDocType,
		String	strPath,
		Variant*	varRetDocName)

Type Method

Description Checks out a document for external editing.

Parameter

lDocID	document ID
lDocType	document type
strPath	Path to where the document will be checked out
varRetDocName	here, the full path and file name for the checked out document is displayed

Return value

0	no error
-1	the document could not be checked out
-7	unknown document type
-51	document contains no files
-52	object already archived
-58	the document was already checked out externally
-59	the document was checked out by another user

Error text can be determined using **GetLastError()**.

Comment

Options

Example

Notes: The checked out file must not be renamed.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ClearSignatureProperties

Definition	ClearSignatureProperties()
Type	Method
Description	Deletes all properties that were previously defined with **SetSignatureProperty**.
Parameters	
Return value	
Comment	It is recommended that this function is executed right after a digital signature.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CloseAllWindows

Definition	CloseAllWindows ()
Type	Method
Description	Closes all child windows of the enaio® client.
Parameter	
Return value	
Comment	
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CloseObjectID

Definition CloseObjectID (long lObjectID,
long lObjectType)

Type Method

Description Closes a window opened by **OpenObjectID**.

Parameter
lObjectID object ID
lObjectType object type

Return value
0 no error
-11 object ID invalid
Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ConvertImage

Definition	ConvertImage	(String	strSource,
		String	strDestination,
		short	nFormat,
		short	nCompression,
		short	nBitsPerPixel,
		short	nFlags)

Type Method

Description This function converts an image file into the specified format.

Parameter

strSource	source file
strDestination	destination file
nFormat	destination format, see GetImageType
nCompression	compression method or loss factor for JPEG format, 1 recommended
nBitsPerPixel	color depth 1, 2, 4, 8, 16, 24, or 0 for color depth of the source file
nFlags	not used

Return value

0	file was successfully converted
<> 0	error

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CopyObject

Definition	CopyObject	(long	lObjectID,
		long	lObjectType,
		long	lFolderID,
		long	lRegisterID,
		short	nFlags,
		Variant	varRetObjectID)

Type Method

Description Creates a copy of an object including multi-fields and document.

Parameter

lObjectID	object ID of the object to be copied
lObjectType	object type of the object to be copied
lFolderID	ID of the destination folder; set the value to 0 to copy a folder.
lRegisterID	ID of the destination register, 0, unless to be copied into a register
nFlags	**0** copy indexing and document **1** copy indexing only
varRetObjectID	the ID of the new object is returned here

Return value

0	no error
-11	document ID unknown
-22	object type unknown
-23	destination register ID unknown
-47	user has no write access or is not allowed to create new objects.
-61	no definite assignment of the object possible.

Error text can be determined using **GetLastError()**.

Comment Creates a copy of an object including multi-fields and document. If the object is part of the version administration, the current version will be copied. This function can also be used to copy folders and registers. In this case the folder or register contents are not copied.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CreateDocumentLink

Definition

```
CreateDocumentLink (long lObjectID,
                    long lObjectType,
                    long lTargetID,
                    long lTargetType)
```

Type

Method

Description

Creates a new reference to a document

Parameters

lObjectID	object index
lObjectType	object type
lTargetID	index of the destination folder/register.
lTargetType	object type of the destination folder/register.

Return value

-1	reference could not be created.
-3	cabinet unknown
-5	register unknown
-7	document type unknown
-13	register identification unknown (if destination type is a register)
-14	folder identification unknown (if destination type is a folder)
-20	specified object type is not a document type
-22	unknown object type
-33	document type invalid (does not match cabinet)
-89	object cannot be created in the destination folder/register.

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CreateMimeFile

Definition

```
CreateMimeFile  (String   strFrom,
                 String   strTo,
                 String   strCC,
                 String   strBCC,
                 String   strSubject,
                 String   strBody,
                 String   strAttachments,
                 String   strCreationTime,
                 Variant* varRetFilename)
```

Type Method

Description Creates a Mime-file

Parameters

strFrom	sender address
strTo	recipient
strCC	CC
strBCC	BCC
strSubject	subject
strBody	text
strAttachments	attached files separated by a pipe character
strCreationTime	creation time in the format: day.month.year hour:minute
varRetFilename	full path and file name of created Mime file

Return value

0	no error
-1	Mime file could not be created

Comment If required, the created file can be deleted by the requesting application.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

DecodeIMAPFile

Definition	DecodeIMAPFile	(String	strFilename,
		Variant	strMailDate,
		Variant	strFrom,
		Variant	strTo,
		Variant	strCC,
		Variant	strBCC,
		Variant	strSubject,
		Variant	strBody,
		Variant	strAttachments)

Type Method

Description Decodes an IMAP file and returns its content.

Parameters

strFilename	full path and file name of the IMAP File
strMailDate	The sending date of the e-mail is returned here.
strFrom	here the name of the sender is returned.
strTo	the recipient list (separated by semicolon) is returned here.
strCC	the CC list (separated by semicolon) is returned here.
strBCC	the BCC list (separated by semicolon) is returned here.
strSubject	the subject is returned here.
strBody	the e-mail content is returned here.
strAttachments	the attachments are returned here, separated by semicolon

Return value

0	if no error
-1	if the file could not be decoded

Comment

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

CreateNewDocShare

Definition

CreateNewDocShare (long lIdent,
long lType,
string Rights
string Users
string Info)

Type

Method

Description

Creates new shared version of the specified document for the specified users

Parameter

lIdent document ID
lType document type
Rights preset rights (RWXU)
Users user IDs of the users for whom the specified document is to be shared, separated by semicolons
Info info text for the document share

Return value

0 "New document share(s) created."
-125 "A modal dialog is open."
-130 "Specified DMS object type unknown."
-131 "Specified DMS object type is not a document."
-132 "Only 'RXWU' rights are permitted."
-133 "Specified document does not exist or was deleted."
-134 "Document shares not possible."
-135 "The required system role to share documents does not exist."
-136 "At least one specified user does not exist."
-137 "Canceled by user."
-138 "Server is reporting an error: "

Exact error text can be determined using **GetLastError()**.

Comment

The parameters 'Rights,' 'Users,' and 'Info' can be empty.

Example

```
dim Application : set Application =
createobject("optimal_AS.application")
Dim sRet


Application.ActivateApp 1

sRet = Application.CreateNewDocShare(590, 262146, "WX", "416;15475",
"Aufruf aus COM-Methode")

if (sRet <> 0) then
	MsgBox Application.GetLastError() & " (" & sRet & ")"
else
	MsgBox "Neue Freigabe(n) angelegt!"
end If

set Application = nothing
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

DeleteFromArchive

Definition DeleteFromArchive (String strParam)

Type Method

Description Deletes the specified folder or the specified document.

Parameters

strParam string with object index and object type or file name of a file with the following structure:*

```
Objectindex1,Objecttype1\r\n
Objectindex2,Objecttype2\r\n
...
Objektindexn,Objekttypn\r\n
```

* **\r\n** stands for a line break, **CR** and **LF**.

Return value

0	no error
-1	handoff string incorrect
-11	invalid document identifier
-41	handoff string empty
-44	the logged-in user is not authorized to delete at least one of the document types in the folder
-45	The logged-in user has no access to one or more files in the requested document type.
-69	the specified handoff file is empty.
-77	the server cannot delete the specified object (further information may be apparent from the server log)
-78	The document is being used by one or more workflow processes and cannot be deleted.

Error text can be determined using **GetLastError()**.

Comment This function deletes the specified object **without prompting** if the user who is logged in to the archive system has sufficient rights to delete the object.
It is also possible when handing over a string with object ID and type to delete several objects with one request. The string needs to have the following structure.

```
object ID1,object type1 object ID2,object type2 object IDn object typen
```

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

DoPrefetch

Definition DoPrefetch (Variant lObjectID,
Variant lObjectType)

Type Method

Description Performs a prefetch for the given object.

Parameters
lObjectID object ID as variant
lObjectType object type as variant

Return value
0 no error
<> 0 error
Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ExecuteRequest

Definition ExecuteRequest (String strRequestName)

Type Method

Description Performs a saved query.

Parameter **strRequestName** name of the saved query

Return value
0 no error
-1 no saved queries found
-2 a query with the specified name does not exist
Error text can be determined using **GetLastError()**.

Comment Displayed by the enaio® client.

If the saved query has variables, they can be specified in the following form:

```
VAR1=Content1;VAR2=Content2;...VARn=Contentn
```

Values of static variables can be placed in a request too:

```
STAT1=Content1;STAT2=Content2;...STATn=Contentn
```

A saved query with variables opens a request page if not all variables have been specified in the request; specified fields are already filled in.

In any case the saved query can be 'forced' to display the request form by using the 'OPT1=1' switch.

Options **OPT1=1**
Forces the search form to be opened

Examples Request for the saved query 'Test' which contains static variables:

```
ExecuteRequest „test STAT1=Hallo;VAR1=Bert*“
```

Request for the saved query 'Test' which contains static variables and that opens the search form in any case:

```
ExecuteRequest „test OPT1=1;STAT1=Hello;VAR1=Bert*“
```

All fields assigned with variables are then filled with their corresponding contents.

Request for the saved query 'Test' with variables handoff

```
Test VAR1=199*;VAR2=Bert*
```

Note: The request data is not verified. SQL errors may occur if e.g. a date is expected and text was specified.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ExecuteRequestEx

Definition	ExecuteRequestEx	(String String	strRequestName, strParams)
Type	Method		
Description	As with **ExecuteRequest**, but the parameters for the saved query need to be specified additionally.		
Parameters	**strRequestName** **StrParams**	name of the saved query parameters for this query (see **ExecuteRequest**)	
Return value	**0** no error **-1** no saved queries found **-2** a query with the specified name does not exist Error text can be determined using **GetLastError()**.		
Comment	In contrast to **ExecuteRequest** this function can also process saved queries with names containing spaces. See also **ExecuteRequest()**		
Options			
Example			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

FindObjectType

Definition	FindObjectType(long lObjectID)
Type	Method
Description	Determines the object type corresponding with a specified object ID
Parameters	**lObjectID** object ID for which the object type is to be determined
Return value	**String** with object type **Empty string** if no object type could be determined.
Comment	In rare cases it may occur that the object type cannot be determined unambiguously. In such case the result string will contain all object types consecutively separated by a comma.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

FindObjectTypeEx

Definition

```
FindObjectTypeEx (long     lObjectID
                  long     lType,
                  VARIANT* vRetObjTypes)
```

Type

Method

Description

Determines the object type corresponding with a specified object index

Parameters

lObjectID	object ID for which the object type is to be determined
lType	type of the object type to be determined: **0** document **1** folder **2** register
vRetObjTypes	the determined object type is returned here

Return value

0	no error
-26	object index invalid
-40	specified type invalid

Comment

By specifying the type this function allows quicker finding the object type. In rare cases it may occur that the object type cannot be determined unambiguously. In such case the result string will contain all object types consecutively separated by a comma.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

FreeDocument

Definition	FreeDocument (long lObjectID)
Type	Method
Description	Unblocks the specified W-Document.
Parameters	**lObjectID** index of the object to be unblocked
Return value	**>0** number of documents that could not be checked in **0** no error **-1** document could not be checked in by the archive server **-2** document locked
Comment	If a **0** is specified for the object ID, all locked documents are unlocked.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

FreeDocumentEx

Definition	FreeDocumentEx (String strDocuments)
Type	Method
Description	Loans the specified documents with an additional thread.
Parameters	**strDocuments** string with IDs of the objects to be shared, separated by semicolons.
Return value	0
Comment	This function opens a thread to loan out objects i.e. it does not wait until all objects were loaned out. If a 0 is specified for the object ID, all locked documents are unlocked.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GenerateColdFiles

Definition	GenerateColdFiles	(String	strFile1,
		String	strFile2,
		String	strToken,
		VARIANT*	vRetFileList)

Type Method

Description Creates image objects from ASCII-COLD import files.

Parameters

strFile1	either a data file or a control file
strFile2	either a data file or a control file
strToken	separator for file list
vRetFileList	list of file names of the created images, separated by the characters specified by strToken.

Return value

0	no error
-83	subdirectory for the COLD files could not be created.
-84	image could not be created
-85	file could not be moved to the destination directory.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GenerateOSFile

Definition	GenerateOSFile	(long long	lObjectID, lObjectType)
Type	Method		
Description	Creates an OS file from the specified parameters.		
Parameters	**lObjectID** object ID **lObjectType** object type		
Return value	**Empty string** if an error occurred **file name** of the created OS-file Error text can be determined using **GetLastError()**.		
Comment	If required, the created OS-file will be deleted by the requesting program.		
Options			
Example			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetActiveDocument

Definition	GetActiveDocument	(String	strDocName,
		BOOL	bOpen,
		VARIANT	vRetVal)

Type Method

Description Delivers the COM object with the optimal_AS: Active Document.

Parameters

strDocName	name of the active document as specified in *as.cfg.*	
bOpen	**TRUE**	if the document has to be opened and is not open already.
vRetVal	here the error number is returned	

Return value

0	no error
-70	no active document with this name could be found
-72	no optimal_AS: Active Document

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetAllArchives

Definition GetAllArchives()

Type Method

Description Identifies all cabinets

Parameters

Return value String with all available cabinets

Comment The return string has the following format:

```
SCHRANK1,OBJEKTTYP1;SCHRANK2,OBJEKTTYP2;... ; SCHRANKn,OBJEKTTYPn
```

Options

Example Source code example:

```
Helpstr=MyAX.GetAllArchives()
```

Helpstr can contain the following string, for example:

```
Customer,1;Patient,2
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetAllOpenFolders

Definition GetAllOpenFolders()

Type Method

Description Provides a list of the IDs and types of all open folders. If registers within the folders are open, their IDs and types will also be provided.

Parameter

Return value string with a list of the IDs and types in the following form:

```
Ordner-o.Register-ID,Ordner-bzw.Registertyp;Ordner-o.Register-
ID,Ordner-bzw.Registertyp;.....
```

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetCurrentResultList

Definition GetCurrentResultList (VARIANT* pstrItems)

Type Method

Description Returns the items of the current hit list.

Parameters

pstrItems here the indexes and object types of the objects displayed in the current hit list are being returned.

```
ObjectID1,ObjecttypeI1; ObjectID2,ObjecttypeI2; …;
ObjectIDn,ObjecttypeIn
```

Return value If higher than **0**, the return value corresponds to the number of returned objects. A return value lower than **0** corresponds to an error.
-40 incorrect input parameter
Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetCurrentSelection

Definition GetCurrentSelection(VARIANT* varResult)

Type Method

Description provides the IDs and object types selected in the current hit list

Parameters **varResult** here the indexes and object types are returned in the following form:

```
ObjectID1,Objecttype1;ObjectID2,Objecttype2;.....
```

Return value The return value corresponds to the number of selected objects.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetDataID

Definition

GetDataID (long lObjectID,
long lObjectType,
short nMode,
short bWriteToFile)

Type

Method

Description

Returns object data, i.e. field names and values

Parameters

lObjectID index of the object to be opened
lObjectType object type
nMode controls the return value.

0	Returns field names and field values
1	Returns basic parameters if it is not a folder or register
2	Returns names and values of multi-fields
10	Corresponds to the value **0**, but returns the internal field names
12	Corresponds to the value **2**, but returns the internal field names
bWriteToFile	If the value is **0**, the object data is returned in a string. If it is **1**, it is written to a file and the file name is returned.

Return value

Result string or **File name**.
Empty string if error
Error text can be determined using **GetLastError()**.

Comment

Options

Example

For nMode = 0:

```
Fieldlabel = Fieldcontent\r\n
Fieldlabel = Fieldcontent\r\n
```

For nMode = 1:

```
ZEITSTEMPEL = FieldContent\r\n
ANLEGER = Fieldcontent\r\n
ARCHIVAR = FieldContent\r\n
ANGELEGT = Fieldcontent\r\n
ARCHIVIERT = Fieldcontent\r\n
```

For nMode = 2 multi-fields of an object:*

```
Mehrfachfeldname=Seitennummer,Werte\r\n
Mehrfachfeldname=Seitennummer,Werte\r\n
```

* **\r\n** stands for a line break, **CR** and **LF**.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetDescription

Definition	GetDescription	(String	strShortName,
		long	lObjectType,
		String	strFieldName)

Type Method

Description Provides the description text from a list field or structure tree for a specified short form.

Parameters

strShortName	short form for which the description text will be returned.
lObjectType	object type
strFieldName	name of the list or structure tree field

Return value

String	with the description,
Empty string	if no description is available or errors occurred.

Error text can be determined using **GetLastError()**.

Comment This function is only valid for list and structure tree fields.

Options

Example The field 'month' has a list with the following entries:

```
01|January
02|February
03|March
```

Source code example

```
HelpStr=MyAX.GetDescription(„02“, 65535, „Month“)
```

HelpStr now has the value 'February'

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetDocTypesFromArchive

Definition GetDocTypesFromArchive (String strSchrankName)

Type Method

Description Determines all document types belonging to a cabinet.

Parameters **strSchrankName** string with the cabinet label

Return value String with the cabinet's document types.

Comment The return string has the following format:

```
DOCUMENT1,OBJECTTYPE1;DOCUMENT2,OBJECTTYPE2 ...
DOCUMENTn,OBJECTTYPEn
```

Options

Example Source code example:

```
Helpstr=MyAX.GetDocTypesFromArchive
```

Helpstr may contain the following string:

```
IncomingDocument,131085;OutgoingDocument,262159;Attribute,131086
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetEnvironment

Definition GetEnvironment (short nEnvType)

Type Method

Description Determines settings from the archive system.

Parameters **nEnvType** number of the setting to be determined

Return value String with the required setting

Comment Possible values for nEnvType:

0	determines osGetTmpDir
1	determines osGetAppCWD
2	determines osGetCfgFileName
3	determines osGetUserName
4	determines osGetHomeCWD
5	determines osGetIniFileName
6	determines the application name (in the case of an OEM version, the OEM name of the client)
7	determines the ID of the current user
8	determines osGetStationNumber
9	determines osGetLocalWorkDir
10	determines FileVersion of the client
11	determines all group names of the current user. If the user is part of more than one group, the group names are separated by semicolons.
12	not used
13	determines a list of the document indexes newly created in the current session. If more than one documents were created, they are listed consecutively separated by commas. E.g. "1234,1235,1236". If no documents had been created the return value contains „EMPTY“
14	determines the full user name.
15	determines the maximum allowed size of text notes.
16	determines the e-mail address of the logged-in user, if it is known by the system.
17	determines the comment field for the logged-in user.
18	determines the GUID of the logged-in user.
19	determines GUID of the department of the logged in user.
20	determines the number of times a user has failed to log in since the last successful login.
21	determines the time from which the current user's account is valid.
22	determines the time until which the current user's account is valid.
23	determines osGetDocLockStation.
24	determines the GUI language set in the client (here the extension of the TRA language file is returned: "en" for English, "fra" for French, and "" for German (default).
25	determines the ID of the object definition language set in the client: e.g. 7-German, 9-English.

Options

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetFilesFromID

Definition	GetFilesFromID	(long	lObjectID,
		long	lObjectType,
		String	strToken)

Type Method

Description Determines the image files belonging to the object.
If the images are archived, they are first loaded into the cache.

Parameters

lObjectID	index of the object
lObjectType	object type
strToken	separator

Error text can be determined using **GetLastError()**.

Return value

String	with file name and path
Empty string	if an error occurred

Error text can be determined using **GetLastError()**.

Comment If multiple pages are part of a document, the file names are listed consecutively separated by a space in the string. If a separator was specified, file names are separated by this character. This makes sense as file names can contain spaces.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetFilesFromIDEx

Definition	GetFilesFromIDEx	(long	lObjectID,
		long	lObjectType,
		String	strToken,
		BOOL	bWriteProtected,
		VARIANT	vFileNames)

Type Method

Description Determines the files corresponding with an object.

Parameters

lObjectID	object ID
lObjectType	object type
strToken	separator used to separate the names in the return string
bWriteProtected	**TRUE** retrieve document with write protection
vFileNames	here the file names are returned

Return value

0	no error
-7	document type unknown
-15	document ID unknown
-51	document has no files

Error text can be determined using **GetLastError()**.

Comment In contrast to **GetFilesFromID()**, it is possible to specify whether the files are to be checked out or retrieved with write protection.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetImageType

Definition GetImageType (String strSource)

Type Method

Description Specifies the image type of the given file.

Parameters **strSource** full path and file name of the image file

Return value

> 0	file type – see comment
< 0	error
-20009	incorrect file format

Comment The following file types are defined:

PCX(1), GIF(2), TIF(3), TGA(4), CMP(5), BMP(6), FROM_BUFFER(7), BITMAP(9), JFIF(10), JTIF(11), BIN(12), HANDLE(13), OS2(14), WMF(15), EPS(16), TIFLZW(17), LEAD(20), LEAD1JFIF(21), LEAD1JTIF(22), LEAD2JFIF(23), LEAD2JTIF(24), CCITT(25), LEAD1BIT(26), CCITT_GROUP3_1DIM(27), CCITT_GROUP3_2DIM(28), CCITT_GROUP4(29), LEAD_NOLOSS(30), FILE_CALS(50), MAC(51), IMG(52), MSP(53), WPG(54), RAS(55), PCT(56), PCD(57), DXF(58), AVI(59), WAV(60), FLI(61), CGM(62), EPSTIFF(63), EPSWMF(64), CMPNOLOSS(65), FAX_G3_1D(66), FAX_G3_2D(67), FAX_G4(68), WFX_G3_1D(69), WFX_G4(70), ICA_G3_1D(71), ICA_G3_2D(72), ICA_G4(73), OS2_2(74), PNG(75), PSD(76), RAWICA_G3_1D(77), RAWICA_G3_2D(78), RAWICA_G4(79), FPX(80), FPX_SINGLE_COLOR(81), FPX_JPEG(82), FPX_JPEG_QFACTOR(83), BMP_RLE(84), TIF_CMYK(85), TIFLZW_CMYK(86), TIF_PACKBITS(87), TIF_PACKBITS_CMYK(88), DICOM_GRAY(89), DICOM_COLOR(90), WIN_ICO(91), WIN_CUR(92), TIF_YCC(93), TIFLZW_YCC(94), TIF_PACKBITS_YCC(95), EXIF(96), EXIF_YCC(97), EXIF_JPEG(98), AWD(99), FASTEST(100), EXIF_JPEG_411(101), PBM_ASCII(102), PBM_BINARY(103), PGM_ASCII(104), PGM_BINARY(105), PPM_ASCII(106), PPM_BINARY(107), CUT(108), XPM(109), XBM(110), IFF_ILBM(111), IFF_CAT(112), XWD(113), CLP(114), JBIG(115), EMF(116), ICA_IBM_MMR(117), RAWICA_IBM_MMR(118), ANI(119), ANI_RLE(120), LASERDATA(121), INTERGRAPH_RLE(122), INTERGRAPH_VECTOR(123), DWG(124), DICOM_RLE_GRAY(125), DICOM_RLE_COLOR(126), DICOM_JPEG_GRAY(127), DICOM_JPEG_COLOR(128), CALS4(129), CALS2(130), CALS3(131), XWD10(132), XWD11(133), FLC(134), KDC(135), DRW(136), PLT(137), TIF_CMP(138), TIF_JBIG(139), TIF_DXF(140), TIF_UNKNOWN(141), SGI(142), SGI_RLE(143), VECTOR_DUMP(144), DWF(145), MPEG1(243), MPEG2(246), JPGOS(1000) (by OPTIMAL SYSTEMS encrypted JPG-file), TIFOS(1001) (by OPTIMAL SYSTEMS encrypted TIFF-file)

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetLastError

Definition	GetLastError()
Type	Method
Description	Provides the error text of the last occurred error.
Parameters	none
Return value	**Error text** as string.
Comment	This function must be called right after the error to avoid receiving the text of a different error.
Options	None
Example	In case of errors, start an error handling routine and find the error text.

```
On error goto ErrorHandling
...
...
Fehlerbehandlung:
ErrStr = MyAX.GetLastError()
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetMainType

Definition	GetMainType	(long long	lObjectID, lObjectType)
Type	Method		
Description	Determines the main type of an object. In case of a multi media document type the actual main type of the document is provided.		

Parameters

lObjectID	object ID
lObjectType	object type

Return value

> = 0	main type of the object
-22	object type unknown
-26	object index unknown

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetObjectFields

Definition GetObjectFields (String strObjectName, long lObjectType, short nMode)

Type Method

Description Determines all fields of an object.
Types and lengths of the fields are also determined.
The object type can be a folder, register, or document.

Parameters

strObjectName	not used.
lObjectType	object type
nMode	mode (see comment)

Return value

String	with field labels
Empty string	if an error occurred.

Error text can be determined using **GetLastError()**.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Comment The return string has the following format:
For nMode = 0:

```
FieldIdentifier1\FieldType1\FieldLength1 TAB
FieldIdentifier2\FieldType2\FieldLength2...
```

For nMode = 1

```
FieldIdentifier1\FieldType1\FeldLength1\TableName.FieldName1 TAB
FieldIdentifier2\FieldType2\FeldLength2\TableName.FieldName2...
```

Possible field types:

Type	Description	Database
A	all characters without numbers	CHAR
X	all characters	CHAR
Z	only numbers	CHAR
S	male/female	CHAR
Q	yes/no	CHAR
G	Uppercase letters	CHAR
P	Patient type	CHAR
T	Left/right	CHAR
L	all characters	CHAR
M	all characters	CHAR
D	Date	DATE
9	Numbers	INTEGER
I	Full text index	INTEGER
#	Decimal numbers	DECIMAL
1	Radio button	SHORT
0	Check boxes	SHORT

Field lengths refer to the number of characters that can be entered. In case of decimal numbers, the capacity of the decimal field consists of two numbers before and after the decimal point (field length – 2).

Options

Example Source code example 1:

```
Helpstr=MyAX.GetObjectFields(“D-Reports”,131073,0)
```

Helpstr may contain the following string, for example:

```
Berichtstyp\X\30                    Bemerkung\X\50
```

Source code example 2:

```
Helpstr=MyAX.GetObjectFields(“D-Berichte”,131073,1)
```

Helpstr may contain the following string, for example:

```
Berichtstyp\X\30\object1.feld1    Bemerkung\X\50\object1.feld2
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetObjectFieldsEx

Definition GetObjectFieldsEx (String strObjectName, long lObjectType, short nMode)

Type Method

Description Determines all fields of an object. This function corresponds to the **GetObjectFields()** function. The only difference is that additionally the catalog type is determined.

Types and lengths of the fields are also determined.

The object type can be a folder, register, or document.

Parameters

strObjectName	not used
lObjectType	object type
nMode	mode (see comment)

Return value

String	with field labels
Empty string	if an error occurred.

Error text can be determined using **GetLastError()**.

Comment The return string has the following format:

For nMode = 0:

```
Feldbezeichner1\Feldtyp1\Feldlänge1\Katalogtyp TAB
Feldbezeichner2\Feldtyp2\Feldlänge2\Katalogtyp...
```

For nMode = 1

```
Feldbezeichner1\Feldtyp1\Feldlänge1\Katalogtyp\
Tabellenname.Feldname1 TAB
Feldbezeichner2\Feldtyp2\Feldlänge2\Katalogtyp\
Tabellenname.Feldname2...
```

For nMode = 2

```
Feldbezeichner1\Feldposition in der Objektdefinition\Feldtyp1\
Feldlänge1\Katalogtyp\Tabellenname.Feldname1 TAB
Fieldlabel2\Feldposition in der Objektdefinition\Feldtyp2\
Fieldlength2\Catalogtype\Tablename.Fieldname2...
```

For nMode = 3

```
Feldbezeichner1\Feldposition in der Objektdefinition\Feldtyp1\
Feldlänge1\Katalogtyp\Tabellenname.Feldname1 \Interner Feldname1TAB
Feldbezeichner2\Feldposition in der Objektdefinition\Feldtyp2\
Feldlänge2\Katalogtyp\Tabellenname.Feldname2\Interner Feldname2...
```

For nMode = 3

```
Feldbezeichner1\Feldposition in der Objektdefinition\Feldtyp1\
Feldlänge1\Katalogtyp\Tabellenname.Feldname1 \Interner
Feldname1\Flags\Flags1\Flags2TAB
Feldbezeichner2\Feldposition in der Objektdefinition\Feldtyp2\
Feldlänge2\Katalogtyp\Tabellenname.Feldname2\Interner
Feldname2\Flags\Flags1\Flags2...
```

Possible field types:

Type	Description	Database
A	all characters without numbers	CHAR

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

X	all characters	CHAR
Z	only numbers	CHAR
S	male/female	CHAR
Q	yes/no	CHAR
G	Uppercase letters	CHAR
P	Patient type	CHAR
T	Left/right	CHAR
L	all characters	CHAR
M	all characters	CHAR
D	Date	DATE
9	Numbers	INTEGER
I	Full text index	INTEGER
#	Decimal numbers	DECIMAL
1	Radio button	SHORT
0	Check boxes	SHORT
K	Button	FROM 3.50.619 !!!

Field lengths refer to the number of characters that can be entered. In case of decimal numbers, the capacity of the decimal field consists of two numbers before and after the decimal point (field length – 2).

Possible values for catalog type:

0 no catalog
1 list
2 tree
3 hierarchy
4 database
5 structure tree
6 AddOn
7 full text

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Options

Example

Source code example 1:

```
Helpstr=MyAX.GetObjectFields(“D-Reports”,131073,0)
```

Helpstr may contain the following string, for example:

```
Berichtstyp\X\30\0                    Bemerkung\X\50\1
```

GetObjectName

Definition	GetObjectName	(long VARIANT*	lObjectID, strReturnObjectName)
Type	Method		
Description	Provides the object name for the given object ID		
Parameters	**lObjectID** **strReturnObjectName**	object ID the name of the object is returned here	
Return value	**0** no error **-1** error Error text can be determined using **GetLastError()**.		
Comment			
Options			
Example			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetObjectNameEx

Definition	GetObjectNameEx	(long	lObjectID,
		long	lType,
		short	nMode,
		VARIANT*	strReturnObjectName)

Type Method

Description Provides the object name for the given object ID

Parameters

lObjectID	object ID
lType	type of the object to be determined **-1** type unknown **0** the object ID is from a document **1** the object ID is from a folder **2** the object ID is from a register
nMode	type of result to be delivered **0** object type name **1** internal object type name **2** table name of the object type **3** UID of the object type
strReturnObjectName	the name of the object is returned here

Return value

0 no error
-1 error
Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetObjectPath

Definition GetObjectPath (long lObjectID,
long lObjectType)

Type Method

Description Determines the location path of an object.
Determines the folder and all registers.
The object type can be a folder, register, or document.

Parameters
lObjectID object ID
lObjectType object type

Return value
String with the object path.
Empty string if an error occurred.
Error text can be determined using **GetLastError()**.

Comment The return string has the following format:

```
CABINETINDEX,CABINETTYP;REGISTERINDEX1,REGISTERTYPE;REGISTERINDEX2,R
EGISTERTYPE;...;REGISTERINDEXn,REGISTERTYPE
```

Register information is only returned if the object is inside a register.
If the searched object is inside a register that is contained inside another register the path will look like this:
folder ID, folder type, ID and type of the register that the object is located in, ID and type of the register that the register is located in etc.

Example:
An object is inside a register named 'REGISTER3' which itself is inside a register named REGISTER2 which again is inside a register named REGISTER1. The object path will look like this:

```
folder ID,folder type;REGISTER3 ID,REGISTER3 type;REGISTER2
ID,REGISTER2 type;REGISTER1 ID,REGISTER1 type
```

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetObjectPathEx

Definition

GetObjectPathEx (long lObjectID,
long lObjectType,
VARIANT* vRetObjectPath)

Type

Method

Description

Determines like GetObjectPath the location path to an object. The fnuction also determines the locations of all reference documents if any exist.
Determines the folder and all registers.
The object type can be a folder, register, or document.

Parameters

lObjectID object ID
lObjectType object type
vRetObjectPath the path is returned here.

Return value

-3 cabinet unknown
-5 register unknown
-22 document type unknown
Error text can be determined using **GetLastError()**.

Comment

The return string has the following format:

```
CABINETINDEX,CABINETTYP;REGISTERINDEX1,REGISTERTYPE;REGISTERINDEX2,R
EGISTERTYPE;...;REGISTERINDEXn,REGISTERTYPE
```

Register information is returned if the object is inside a register. The reference document paths are separated by a | (pipe) character.

If the searched object is inside a register that is contained inside another register the path will look like this:
folder ID, folder type, ID and type of the register that the object is located in, ID and type of the register that the register is located in etc.

Example:
An object is inside a register named 'REGISTER3' which itself is inside a register named REGISTER2 which again is inside a register named REGISTER1. The object path will look like this:

```
folder ID,folder type;REGISTER3 ID,REGISTER3 type;REGISTER2
ID,REGISTER2 type;REGISTER1 ID,REGISTER1 type
```

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetObjectTypeImage

Definition

```
GetObjectTypeImage (long    lObjectID,
                    long    lObjectType,
                    String  strImageFormat,
                    Variant vRetImagePath)
```

Type

Method

Description

Returns the icon of an object type as an image file. Currently, the file is only available in GIF format.

Parameters

lObjectID	ID of the object (can be 0 if it not an object with an icon catalog)
lObjectType	object type
strImageFormat	currently not used
vRetImagePath	the full path to the image file is returned here

Return value

0	image file successfully transmitted
-1	image file could not be determined

Comment

The error texts cannot be determined with GetLastError().

Options

Example

```
Dim a As Object
Dim vRetImagePath as Variant

Set a = CreateObject("optimal_as.application")

a.GetObjectTypeImage lObjectID, lObjectType, “”, vRetImagePath
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetRegTypeFromArchive

Definition GetRegTypeFromArchive (String strSchrankName)

Type Method

Description Determines all registers with name and object type belonging to a cabinet.

Parameters **strSchrankName** string with the cabinet label

Return value Semicolon-separated string with registername,registertype.

Comment The return string has the following format:

```
REGISTERNAME1,OBJECTTYPE1;REGISTERNAME2,OBJECTTYPE2
```

Options

Example Source code example:

```
sRet=MyAX.GetRegTypeFromArchive(sArchiveName)
```

sRet may contain the following string:

```
RegisterName1,6488064;RegisterName2,6488065
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetRelReferenceObject

Definition

GetRelReferenceObject(VARIANT* vpObjectID,
VARIANT* vpObjectType)

Type

Method

Description

Returns the reference object specified in the system for working with relations.

Parameters

vpObjectID	the object ID of the reference object is returned here
vpObjectType	reference object type is returned here

Return value

0	no error
-1	no reference object known

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetResultFields

Definition

```
GetResultFields   (String   strObjectType,
                   long     lObjectType,
                   long     lMode,
                   String   strDelimiter,
                   VARIANT* pstrFields)
```

Type

Method

Description

Returns the configured fields of the current users for hit lists of the indicated type.

Parameters

strObjectType	internal name of the object type (only evaluated if ID=-1 is passed)
lObjectType	object type
lMode	type of the result to be delivered
strDelimiter	separator used to separate the field name in the return string. If there is no separator, an „;“ will be set as default.
pstrFields	user-configured fields are returned in the following form here (see comments):

```
Field1; Field2; …; Fieldn
```

Return value

0	no error
-7	object type unknown
-40	incorrect input parameter

Error text can be determined using **GetLastError()**.

Comment

lMode = 0 returns display names of the fields
lMode = 1 returns internal names of the fields
lMode = 2 returns the OSGUIDs of the fields

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetSignatureProperty

Definition	String GetSignatureProperty (String strPropertyName)	
Type	Method	
Description	Returns a property that was previously defined with SetSignatureProperty.	
Parameters	**strPropertyName**	name of the property
Return value	Value of the property.	
Comment		
Options		
Example		

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetSelectedObject

Definition GetSelectedObject (Variant strSelectedObjects)

Type Method

Description Returns objects from a hit list selected with **SelectObject**.

Parameters

strSelectedObjects here objects are returned in the following form:

```
ObjectID1,Objecttype1;...,...;ObjectIDn,Objecttypen
```

Return value

3	objects were selected, selection finished
2	selection canceled
1	no object selected yet
0	SelectObject was not executed
-3	cabinet unknown
-5	register unknown
-26	object ID unknown

Comment If this function is used in a loop, make sure that window messages are processed; otherwise no object can be selected from the hit list.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetSignDocumentResult

Definition GetSignDocumentResult (Variant vRetSignText)

Type Method

Description Returns the result of the last digital signature of a document.

Parameters **vRetSignText** the chosen signature text is returned here

Return value

0	signature successful
1	signature canceled
100	digital signature has not been started yet
101	digital signature still in progress
-1	error while initializing the digital signature
-2	no signature texts were configured
-77	signature model has not been initialized
-79	the file to be signed does not exist

Comment The error texts cannot be determined with GetLastError().

Options

Example

```
Dim a As Object
Set a = CreateObject("optimal_as.application")

a.signdocumentex docID, docType, False, signText

abort = False
Do While abort = False
    c = a.getsigndocumentresult(signText)
    If c <> 101 Then
        abort = True
    End If
    DoEvents
Loop
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetWDocPattern

Definition

```
GetWDocPattern (long    lObjectType,
                String  strPatternName,
                String  strPath,
                VARIANT* strEditorName,
                VARIANT* strFileName)
```

Type

Method

Description

Copies the template file for the given object type into the specified directory

Parameters

lObjectType	desired object type, must be a W-Document
strPatternName	alias name of the template
strPath	destination path
strEditorName	the name of the editing application for this template is returned here
strFileName	the full path and name of the copied template is returned here

Return value

0	no error
-7	document type unknown
-35	no templates defined for this type
-36	no template found with this name
-37	template could not be retrieved
-60	object is not a W-Document
-104	The entered alias name exists more than one time.
-105	The specified namespace was not found.

Error text can be determined using **GetLastError()**.

Comment

If *as.cfg* contains the entry STOREATSERVER=1, the template file is transferred by the archive server, otherwise it is copied from the template directory.

Options

The name of the template can also be entered along with the intended namespace, separated by '::'.

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GetWDocPatternNames

Definition	GetWDocPatternNames	(long	lObjectType,
		String	strToken,
		VARIANT*	strPatternNames)

Type Method

Description Provides all alias names of the W-templates that are linked to this document type.

Parameters

lObjectType	desired object type, must be a W-Document
strToken	separator used to separate the names in the return string
strPatternName	the names are returned here

Return value

0	no error
-7	document type unknown
-35	no templates defined for this type
-60	object is not a W-Document

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GoToDocPage

Definition GoToDocPage (long lObjectID,
long lObjectType,
long lPageNum)

Type Method

Description Navigates in an open document to the specified page.

Parameters

lObjectID	object index
lObjectType	object type
lPageNum	page number

Return value

0	no error
-7	document type unknown
-26	document index unknown
-91	document type is not supported
-92	document not open
-93	document contains no pages

Error text can be determined using **GetLastError()**.

Comment The function only supports document types with the main types **1** X = grayscale
2 D = black/white
3 P = color

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Options

Example

InfoWindow

Definition InfoWindow

Type Property

Description Provides an object to control the info window.

Parameters

Return value an object to control the info window

Comment In order to use the info window within an event, it is not necessary to get the object over this interface as it can be accessed in any event under the name 'InfoWindow'.

Options

Example

```
Dim a As Object

Set a = CreateObject("optimal_as.application")
a.InfoWindow.URL = “www.google.de”
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

InsertFileList

Definition	InsertFileList	(short	nMainType,
		String	strFileName,
		long*	lReturnObjectID,
		long*	lReturnObjectType)

Type Method

Description Inserts one or several files (depending on main type) to the archive as new document.

Parameter

nMainType	main type
strFileName	file containing the names of the files to be inserted
lReturnObjectID	the object index is returned here
lReturnObjectType	the object type is returned here

Return value

0	no error
-1	insert failed
-62	a data sheet is currently open.
-69	Handoff file is empty.

Exact error text can be determined using **GetLastError()**.

Comment Possible main types are described in the chapter **Introduction**.

The file names in the handoff file must be separated with **CR/LF**. The files must be located in the path determined with **GetEnvironment**(0). (Note: If this function, depending on computer configuration, returns short file names, data paths thus must also be entered with short file names in the handoff file.)

In the enaio® client the user can define in which cabinet or folder to insert the document. Furthermore it is possible to define the document type and to insert into the filing tray. More than one file in the handoff file are allowed for the main types 1, 2 and 3 . For all other main types it must be exactly one file.

If an additional file called **FILEDATA.TXT** is specified in the handoff file, index data for the initial indexing of the new document can be entered there. (Note: This file must be inside the working directory **GetEnvironment**(9) and must be listed as the last file in the handoff file, after the document files.) In **FILEDATA.TXT** the index fields can be specified either via their name, or via their internal name bracketed by '%' characters. Usually an indexing form is displayed to the user and he can edit the form. But if additionally the parameter **ShowNoDialog**=1 is shown in this file, then no indexing file will be shown to the user in the enaio® client. I.e. the new document is inserted, but users cannot edit its index data.

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

There are two ways to insert one or more files as documents into the archive:

1. Files and potential **FILEDATA.TXT** information in a string

```
[FILELIST]
#0000=C:\DOKUM~1\user\LOKALE~1\Temp\DOC0000.TIF
#0001=C:\DOKUM~1\user\LOKALE~1\Temp\DOC0001.TIF
#0002=C:\DOKUM~1\user\LOKALE~1\Temp\DOC0002.TIF
…
#000n=C:\DOKUM~1\user\LOKALE~1\Temp\DOC000n.TIF
[DATA]
FIELD0=Titel=Expose
FIELD1=Annotation=Added per script
FIELD2=%Author%=Jon Doe
```

2. **FILEDATA.TXT** in the file list:

```
[FILELIST]
#0000=C:\DOKUM~1\user\LOKALE~1\Temp\DOC0000.TIF
#0001=C:\DOKUM~1\user\LOKALE~1\Temp\DOC0001.TIF
#0002=C:\DOKUM~1\user\LOKALE~1\Temp\DOC0002.TIF
…
#000n=C:\DOKUM~1\user\LOKALE~1\Temp\DOC000n.TIF
```

Options

Example

Example of the handoff file

```
C:\DOKUME~1\user\LOKALE~1\Temp\DOC000000.TIF
C:\DOKUME~1\user\LOKALE~1\Temp\OSTEMP\FILEDATA.TXT
```

Example of a possible FILEDATA.TXT

```
[DATA]
FIELD0=Titel=Expose
FIELD1=Annotation= Added per script
FIELD2=%Author%=Jon Doe
…
ShowNoDialog=1
```

In the FILEDATA.TXT file a location can be specified now.

```
[PREDEFTARGET]
DOKUMENTINTERN
DOKUMENTTYP
SCHRANKINTERN
SCHRANK
SCHRANKIDENT
REGISTERINTERN
REGISTER
REGISTERIDENT
ABLAGE
```

The following rules must be followed:

- A document type must be specified.
- A cabinet type must be specified.
- A register type can be specified.
- If there is no register type, a cabinet ID must be specified.
- If there is a register type, a register ID must be specified. In this case, the cabinet ID is not necessary.
- If a register type is specified, it must belong to the given cabinet.
- The specified document type must correspond to the cabinet.
- It must be possible to create a document of the specified document type at the location (register or cabinet).

Example:

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

```
[PREDEFTARGET]
DOKUMENTTYP=Report
SCHRANK=Patient
REGISTER=Hospital stay
REGISTERIDENT=123456
```

If the specification in the DATA section meet all requirements, a document will be created accordingly at the specified location.

Alternatively the document can be created in the user's filing tray.

Example:

```
[PREDEFTARGET]
DOKUMENTTYP=Report
SCHRANK=Patient
ABLAGE=1
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

InsertFileListS

Corresponds to 'InsertFileList' and can be used from VBScript.

Definition	InsertFileList	(short	nMainType,
		String	strFileName,
		VARIANT*	varReturnObjectID,
		VARIANT*	varReturnObjectType)

InsertIntoArchive

Definition

```
InsertIntoArchive (String strFilename,
                   long   *lpReturnObjectID,
                   long   *lpReturnObjectType)
```

Type

Method

Description

Allows to insert a new folder into a cabinet.

Parameters

strFilename	handoff file (for structure see comment)
***lpReturnObjectID**	the object index is returned here
***lpReturnObjectType**	here the object type is returned

Return value

0	if no error
-1	folder insertion failed
-30	one or more mandatory fields not filled out
-24	field names could not be resolved
-28	invalid field value for a given field
-31	data type of entered value is not compatible with the data type of the associated field.
-32	handoff file does not exist
-47	no write permission for this object
-64	server error occurred

Error text can be determined using **GetLastError()**.

Comment

The handoff file has the following structure:

```
[EINFÜGEN]
SCHRANK=
FIELD1=
FIELD2=
...
FIELDn=
```

The following lines must be inserted in order to define values for the table controls, where the separator character equals chr(17).

```
[Table@TABLENAME]
Line0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Line1={Column1Row2(Separator)Column2Row2}
```

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options

PFLICHTFELDER=[0;1]
Mandatory fields are verified for the value 1.

VORBELEGUNG=[0;1]
fills in preset values fields which have not been defined explicitly. Default: 0

CHECKKEYFIELDS=[0;1]
For the value 0 the key field check is deactivated. Default: 1

Example

Inserting a folder into the cabinet 'Customer'

```
[EINFÜGEN]
SCHRANK=Kunde
FIELD1=Class=Customer
FELD2=Vorlage=10.03.1997
FIELD3=optimal_AS=1
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Note: Key fields are not verified here. The numbering must not contain gaps. Entries after a gap are ignored.

InsertIntoArchiveS

Corresponds to 'InsertIntoArchive' and can be used from VBScript.

Definition	InsertIntoArchiveS	(String	strFilename,
		VARIANT*	varReturnObjectID,
		VARIANT*	varReturnObjectType)

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

InsertIntoDocument

Definition InsertIntoDocument (String strFilename,
long *lpReturnObjectID,
long *lpReturnObjectType)

Type Method

Description Allows to insert a new document into a cabinet.

Parameters

strFilename	handoff file (see comment for structure)
***lpReturnObjectID**	object index is returned
***lpReturnObjectType**	object type is returned

Return value

0	if no error
-1	inserting document failed
-2	no folder specified
-3	folder unknown
-6	no document type specified
-7	document type unknown
-9	Document type does not belong to specified folder
-10	folder identifier missing
-24	field names could not be resolved
-28	invalid field value for a given field
-30	one or more mandatory fields not filled out
-31	data type of entered value is not compatible with the data type of the associated field.
-32	handoff file does not exist
-47	no write permission for this object
-64	server error occurred
-89	The object must not be created in destination folder/register.

Error text can be determined using **GetLastError()**.

Comment The handoff file has the following structure:

```
[EINFÜGEN]
SCHRANK=
DOKUMENT=
SCHRANK-ID=
REGISTER-ID=
MAINTYPE=
FIELD1=
FIELD2=
...
FIELDn=
DATEI1=
DATEI2=
...
DATEIn=
```

The following lines must be inserted in order to define values for the table controls, where the separator character equals chr(17).

```
[Table@TABLENAME]
Line0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Line1={Column1Row2(Separator)Column2Row2}
```

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Note: The main type of the destination document must be defined with **MAINTYPE=#** where # stands for the main type no. . Possible main types are described in the chapter **Introduction**.

It is not recommended that this function is used to create links to documents which are subject to the variant administration.

Options

PFLICHTFELDER=[0;1]
Mandatory fields are verified for the value 1.

SHOWTEMPLATES=[0;1]
The template dialog for W_Documents is displayed if the value is 1 . The selected template is inserted. The FILE1 entry then does not need to be inserted. If no template is selected a reference is created.

ENABLEOPTIONS=[0;1]
If the value is 1, the options of the template dialog are activated.

ARCHIVIERBAR=[0;1]
Sets the archiving status if files are specified.

DATEILÖSCHEN=[0;1]
Deletes the specified source files. Default is 0.

VORBELEGUNG=[0;1]
fills in preset values fields which have not been defined explicitly. Default: 0

CHECKKEYFIELDS=[0;1]
For the value 0 the key field check is deactivated. Default: 1

FOREIGN-ID=DocumentID
SYSTEM-ID=
Both of these parameters can be used to create multi-cabinet-spanning links. To do so, set the FOREIGN ID as the document ID you want to link to and the SYSTEM ID to zero and define the CABINET ID.

To attach multiple parameters new sections have to be inserted into the file. These sections start with **MULTI_** followed by the internal field label of the multi-field (e.g. MULTI_FIELD1). They are then followed by the data which will be inserted into the multiple parameters, e.g.:

```
[MULTI_FIELD1]
Data1=Seitennummer,Text
Data2=Seitennummer,Text
```

Example

Inserting into the 'Customer' document, 'inbound document'.

```
[EINFÜGEN]
SCHRANK=Kunde
REGISTER=Register
DOKUMENT=InboundDocument
SCHRANK-ID=4711
REGISTER-ID=0
FIELD1=Class=Customer
FELD2=Vorlage=10.03.1997
FILE1=c:\temp\image.tif
[MULTI_FIELD1]
DATA1=1,Peter
DATA2=2,Hans
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Creating a multi-cabinet spanning link:

```
[EINFÜGEN]
SCHRANK=Kunde
DOCUMENT=Documentation
SCHRANK-ID=6296
SYSTEM-ID=0
FOREIGN-ID=6305
```

Note: Key fields are not verified here. The numbering must not contain gaps. Entries after a gap are ignored. Specifying a cabinet ID can be skipped if the register ID is specified. If no register ID is specified (REGISTER ID=0) the document is created at the highest level. If a register ID is specified the document is created in the specified register.

InsertIntoDocumentS

Corresponds to 'InsertIntoDocument' and can be used from VBScript.

Definition	InsertIntoDocumentS	(String	strFilename,
		VARIANT*	varReturnObjectID,
		VARIANT*	varReturnObjectType)

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

InsertIntoRegister

Definition InsertIntoRegister (String strFilename,
long *lpReturnObjectID,
long *lpReturnObjectType)

Type Method

Description Inserting a new register

Parameters

strFilename	handoff file (structure see comment)
***lpReturnObjectID**	here the object index is returned
***lpReturnObjectType**	here the object type is returned.

Return value

0	if no error
-1	inserting register failed
-2	no folder specified
-3	cabinet unknown
-4	no register specified
-5	register unknown
-8	register does not belong to the folder
-24	field names could not be resolved
-28	invalid field value for a given field
-30	one or more mandatory fields not filled out
-31	data type of entered value is not compatible with the data type of the associated field.
-32	handoff file does not exist
-47	no write permission for this object
-64	server error occurred
-89	The object must not be created in destination folder/register.

Error text can be determined using **GetLastError()**.

Comment The handoff file has the following structure:

```
[EINFÜGEN]
SCHRANK=
REGISTER=
SCHRANK-ID=
REGISTER-ID=
FIELD1=
FIELD2=
...
...
FIELDn=
```

The following lines must be inserted in order to define values for the table controls, where the separator character equals chr(17).

```
[Table@TABLENAME]
Line0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Line1={Column1Row2(Separator)Column2Row2}
```

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options **PFLICHTFELDER=[0;1]**

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Mandatory fields are verified for the value 1.

VORBELEGUNG=[0;1]
fills in preset values fields which have not been defined explicitly. Default: 0

CHECKKEYFIELDS=[0;1]
For the value 0 the key field check is deactivated. Default: 1

Example Inserting into the 'Customer' register **register**.

```
[EINFÜGEN]
SCHRANK=Kunde
REGISTER=Register
SCHRANK-ID=4711
REGISTER-ID=0
FELD1=Typ=Brief
FIELD2=Text=Test
```

Note: Key fields are not verified here. The numbering must not contain gaps. Entries after a gap are ignored.

The entry *MANDATORYFIELDS* =1 ensures that mandatory fields are verified. The default value is 0.

The entry VORBELEGUNG=1 fills in preset values fields which have not been defined explicitly. Default: 0

Specifying a cabinet ID can be skipped if the register ID is specified. If no register ID is specified (REGISTER ID=0) the register is created at the highest level. If a register ID is specified the register is created in the specified register. If the cabinet ID and the register ID are specified, then it is additionally checked whether the specified register is located in the given folder.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

InsertIntoRegisterS

Corresponds to 'InsertIntoRegister' and can be used from VBScript.

Definition	InsertIntoRegisterS	(String	strFilename,
		VARIANT*	varReturnObjectID,
		VARIANT*	varReturnObjectType)

InsertNewDMSObject

Definition	InsertNewDMSObject	(long	lType,
		long	lPosIdent,
		String	strFilename,
		VARIANT*	newObjIdent)

Type Method

Description Allows a new object – folder, register, or document – to be inserted by opening an index data form in enaio® client.

Parameter
lType DMS object type of the new object
lPosIdent ID of the new object's location, not evaluated for folders.
strFilename handoff file with index data field presets, as well as the name and path in the case of document files.
newObjIdent ID of the new object

Return value

-120 specified DMS object type unknown.
-121 cannot determine location for new object creation.
-122 no new object of the specified type can be created at the specified location.
-123 input canceled.
-124 the index write permission required to create a new object of the specified type has not been granted.
-125 a modal dialog is open.
-126 object creation already in progress.
-127 the DMS object type to be created is not present in the cabinet for the location.

The following return values will create a document without pages:

1 the object write permission required to save files with the specified type has not been granted.
2 at least one of the specified files does not exist.
3 at least one specified file does not match the document type to be created.
4 too many files specified for the document type to be created.

Error text can be determined using **GetLastError()**.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Comment

Preset index data are optional. They can be specified as follows:

```
[Data]
Feld0=Feldname=Wert
Feld1=%Feldname%=Wert
…
```

Internal names for fields are bracketed by percentage characters.

Files can be specified as follows:

```
[Files]
File0=C:\tmp\File1.JPG
File1=C:\tmp\File2.JPG
…
```

If `DeleteAfterInsert=1` is used, the files will be deleted after they have been transferred.
Without files, the appropriate module to create files will be opened in enaio® client after the indexing.
If `OnlyIndexData=1` is used, no module will be opened – instead, a document without pages will be created. Similarly, no module will be opened if the document type has no pages.
If `ShowNoDialog=1` is used, no index data form will be opened – instead, the object will be created without user action.

Options

-

Example

-

LicLogin

Definition

LicLogin (strModulName)

Type

Method

Description

Licenses the specified module.

Parameters

strModulName name of the module to be licensed

Return value

0 if the module could be licensed

-1 general error in function
>0 license error number from the application server

Comment If the specified module has already been licensed for the current workstation,
this license will be used.

Options

Example

LicLogout

Definition LicLogout (strModulName)

Type Method

Description Unblocks the license for the specified module.

Parameters **strModulName** name of the module to be unblocked

Return value **0** if the license could be unblocked
-1 if the license could not be unblocked

Comment Only modules previously licensed with **LicLogin** can be unblocked with LicLogout

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

LinkDocuments

Definition

```
LinkDocuments (long lObjectID1,
               long lObjectType1,
               long lObjectID2,
               long lObjectType2)
```

Type

Method

Description

Creates a link between two documents.

Parameters

lObjectID1	ID of the first document
lObjectType1	type of the first document
lObjectID2	ID of the second document
lObjectType2	type of the second document

Return value

0	no error
-20	one of the object types is not a document type, the erroneous type can be found in the error text
-29	one or both object IDs are unknown
-34	this link already exists
-86	object indexes are identical
-87	objects of the type portfolio are not allowed
-88	objects from the filing tray are not allowed

Error text can be determined using **GetLastError()**.

Comment

The link between two documents is saved in their notes.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

MergeArchives

Definition	MergeArchives	(long	lSourceID,
		long	lTargetID,
		long	lObjectType)

Type Method

Description Combines the contents of two folders.

Parameters

lSourceID	source folder
lTargetID	destination folder
lObjectType	object type

Return value

0	no error
-22	object type unknown
-47	no right to create objects.
-80	source folder unknown
-81	destination folder unknown
-82	updating database failed

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

MoveObject

Definition	MoveObject	(long	lObjectID,
		long	lObjectType,
		long	lFolderID,
		long	lRegisterID)

Type Method

Description Moves an object (register or document).

Parameters

lObjectID	ID of the object to be moved
lObjectType	corresponding object type
lFolderID	folder ID of the destination folder.
lRegisterID	register ID of the destination register

Return value

0	no error
-5	register type unknown (register ID belongs to wrong object type)
-7	document type unknown
-13	register ID unknown
-14	folder ID unknown
-29	object ID invalid
-68	folders cannot be moved.
-82	updating database failed
-90	the specified object is a reference copy
539	violation of object type relation

Error text can be determined using **GetLastError()**.

Comment If a register ID > 0 is specified, the folder ID is ignored. If register ID and folder ID = 0, the object is moved to the root.

Reference documents cannot be moved.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OleDdeRequest

Definition	OleDdeRequest	(String String	strFunctionName, strParameter)
Type	Method		
Description	This is a legacy function and is no longer supported.		
Parameters	**strFunctionName** **strParameter**	name of the function parameter as string	
Return value			
Comment	COM functions must be called directly instead of using this function.		
Options			
Example			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenAboDialog

Definition

```
OpenAboDialog (long   lHwnd,
               long   lObjectID,
               long   lObjectType,
               long   lFlags,
               String strInfoText)
```

Type

Method

Description

Displays the subscription dialog.

Parameters

lHwnd	window handle
lObjectID	document ID
lObjectType	document type
lFlags	see comment
strInfoText	info text, entered into the dialog's info field

Return value

1	dialog canceled by user
0	no error
-7	object type unknown
-26	lObjectID is unknown
-62	a dialog is already open

Error text can be determined using **GetLastError()**.

Comment

By setting flags certain check boxes can be set in the dialog. These flags can be combined.

1	checkbox document changed
2	checkbox index data changed
4	checkbox object created
8	checkbox object deleted

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenDataDlg

Definition	OpenDataDlg	(long long short	lObjectID, lObjectType, nMode)

Type Method

Description Opens the data sheet of the specified object.

Parameters

lObjectID	object index
lObectType	object type
nMode	editing mode (see comment)

Return value

0	no error
-7	object unknown
-11	invalid document identifier (= 0)

Error text can be determined using **GetLastError()**.

Comment

Displayed by the enaio® client.
Possible values for nMode are:

0	the data sheet is displayed read-only as a modal window
1	the data sheet is opened for editing as a modal window
2	the data sheet is opened for editing as a modeless window
3	the data sheet is displayed read-only as a modeless window

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenObjectID

Definition

OpenObjectID (long lObjectID,
long lObjectType,
short nMode)

Type

Method

Description

Opens the specified object.

If the object is a **document**, then the image is displayed and/or the corresponding document is loaded (W-Document). If the document has no images, the data sheet is opened.

If the object is a **folder** or **register** then the corresponding folder is opened.

Parameters

lObjectID	index of the object to be opened
lObjectType	object type
nMode	Sets for W-Documents (and only those) if the document will be opened as read-only or for editing. If the parameter is **0**, the document will be opened as usual as read-only. If the mode **= 1**, the document will be opened for editing. nMode = **2**, If there are multiple document variants, instead of the active variant the document with the specified ID will open in read-only mode. nMode = **3**, If there are multiple document variants, instead of the active variant the document with the specified ID will open for editing. read-only mode.

Return value

0	no error
-11	if lObjectID = 0

Error text can be determined using **GetLastError()**.

Comment

Displayed by the enaio® client.

Options

Example

A W-Document will be opened as read-only:

```
intRet = MyAX.OpenObjectID(lngID, lngType, 0)
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenResultList

Definition OpenResultList (String strIDList, String strTitle)

Type Method

Description Opens a hit list with the passed objects.

Parameters

strIDList	object list in the following form: ID,Objecttype;ID1,Objecttype1…
strTitle	window title

Return value

0	no error
-1	hit list could not be created

Comment

Options

Example

```
Dim a As Object
Set a = CreateObject("optimal_as.application")
a.OpenResultList "873722,131089", "test"
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenObjectIDEx

Definition

```
OpenObjectIDEx (long  lObjectID,
                long  lObjectType,
                BOOL  bWriteProtected,
                BOOL  bActivateIfOpen)
```

Type

Method

Description

Opens the specified object. If the object is a document, the image is displayed an/or the corresponding document is loaded (W-Document). If the document has no images, the data sheet is opened. If the object is a cabinet or register then the corresponding folder is opened.

Parameters

lObjectID	object ID
lObjectType	object type
bWriteProtected	**TRUE**, if the document will be opened as read-only
bActivateIfOpen	**TRUE**, if an already open window of this type will be activated

Return value

0	no error
-11	object ID invalid

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenURL

Definition	OpenURL	(String	strURL,
		BOOL	bShowAddressBar,
		BOOL	bOpenNewWindow)
Type	Method		
Description	Opens an URL.		

Parameters

strURL	the address to be opened
bShowAddressBar	**FALSE** if the address bar should be hidden
bOpenNewWindow	**TRUE** if the address should be opened in a new window

Return value

0	no error
-1	URL window could not be opened

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

OpenWorkItem

Definition	OpenWorkItem (String strWorkItemID)
Type	Method
Description	Opens a process step in the inbox of the logged-on user.
Parameters	**strWorkItemID** ID of the process step to be opened.
Return value	**0** no error **-113** workflow module is not available **-114** Id of the process step not found in the inbox **-115** internal error while opening the process step Error text can be determined using **GetLastError()**.
Comment	This functions exclusively opens process steps which are located in the inbox of the currently logged-on user.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

PrintDocumentID

Definition PrintDocumentID (String strParam)

Type Method

Description Prints the given objects if the objects are of the type 'document'. Except for documents of the W type.

Parameters **strParam** string with the following content:

```
INDEX,OBJECTTYPE
```

Return value

0	no error
-1	not a document type object
-7	document type unknown
-40	parameter to be passed wrong

Error text can be determined using **GetLastError()**.

Comment The print dialog of enaio® client will open.

More than one object can be specified. The parameter string has the following form:

```
INDEX1,OBJECTTYPE ... INDEXn,OBJECTTYPE
```

The object information are separated with spaces. Please note that only one object type can be used.

The file name can also be passed.
The file would have the following structure:*

```
INDEX1,OBJECTTYPE\r\n
...
...
INDEXn,OBJECTTYPE\r\n
```

* **\r\n** stands for a line break, **CR** and **LF**.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

RefreshFolderWindow

Definition

```
RefreshFolderWindow(long   lObjectID,
                    long   lObjectType,
                    short  lObjectID2Select)
```

Type

Method

Description

Updates all folder and register windows open in the client and specified by ID and type. If this window is the active one, a new object can be selected by specifying ID2Select.

Parameters

lObjectID	object index
lObjectType	object type
lObjectID2Select	object which will be selected (**0** if you want to keep the previous selection)

Return value

Comment

This function updates all open folder windows of the specified folder/register. A new selection on the object handed off to the function will only take effect if the window is the client's active window.

Window updates may take a while, depending on the system. During this time the client will be temporarily unavailable. For that reason, consider carefully when to carry out this function.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ScanDocument

Definition	ScanDocument	(long	lObjectID,
		long	lObjectType,
		short	nMode)

Type Method

Description Allows images to be attached to an existing document.

Parameters

lObjectID	object index
lObjectType	object type
nMode	mode (see comment)

Return value

0	no error
-11	invalid document identifier
-19	object is not assigned to a folder
-20	specified object type is not a document type.
-21	document already archived

Error text can be determined using **GetLastError()**

Comment If nMode has the value 1, the index form is opened before the scan window. If this form if left using 'Cancel', the scan window will not be opened.

The scan dialog of the client will be used.

This function cannot be used to change already archived documents.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ShowVariantsDialog

Definition

```
ShowVariantsDialog(long     lObjectID,
                   long     lObjectType,
                   BOOL     bCreateSingleVariant,
                   BOOL     bAllowDragDrop,
                   BOOL     bOpenAfterCreation,
                   BOOL     bSetNewVersionActive,
                   long     lHwnd,
                   VARIANT  vRetNewObjectID)
```

Type

Method

Description

Shows the variant administration dialog.

Parameters

lObjectID	document ID
lObjectType	document type
bCreateSingleVariant	specifies whether the user is only allowed to create a single variant
bAllowDragDrop	specifies whether drag and drop is allowed in this dialog
bOpenAfterCreation	specifies whether the new variant is opened immediately
bSetNewVersionActive	specifies whether the new variant is marked as active
lHwnd	window handle
vRetNewObjectID	returns IDs of new created variants separated with semicolons

Return value

1	dialog canceled by user
0	no error
-7	object type unknown
-26	lObjectID is unknown
-60	object not of type W-Document
-62	a dialog is already open
-93	object contains no pages

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SelectObject

Definition	SelectObject	(long	lObjectID,
		long	lObjectType,
		BOOL	bMultiSelect
		String	strTitle)

Type Method

Description Opens the hit list for object selection.

Parameters

lObjectID	folder or register ID
lObjectType	object type of the folder or register
bMultiSelect	**TRUE** for multiple selection
strTitle	window title

Return value

1	selection was started
-25	object is not a folder or register
-26	object ID invalid
-27	a hit list is already open for object selection

Error text can be determined using **GetLastError()**.

Comment The open hit list has two new buttons in the toolbar to select the selected objects or to cancel the selection.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SendMail

Definition	SendMail	(String strFileList, short nDelete)	
Type	Method		
Description	Opens a form in order to send an e-mail		
Parameters	**strFileList**	file list attached to the e-mail	
	nDelete	**0** if the files should not be deleted, otherwise **1**	
Return value			
Comment			
Options			
Example			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SendMailMapi

Definition	SendMailMapi	(String	strFrom,
		String	strTo,
		String	strCc,
		String	strBcc,
		String	strSubject,
		String	strBody,
		Strin	strFiles,
		long	lReserved,
		short	nCompress)

Type Method

Description Sends the e-mail without a confirmation dialog

Parameters

strFrom	sender
strTo	recipient
strCc	copy for
strBcc	blind copy for
strSubject	subject
strBody	text
strFiles	list of files to be attached, separated by commas
lReserved	not used
nCompress	not used

Return value

0	no error
<> 0	error

Error text can be determined using **GetLastError()**.

Comment If the key **SendMail=OUTLOOK** is entered in the *as.cfg* in the [CLIENT] section,
the Outlook e-mail window will open before the e-mail is sent.
Hence the enaio® Outlook add-ins can be used.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SetPlannedRetention

Definition

```
SetPlannedRetention (long    lObjectID,
                     long    lObjectType,
                     String  strRetentionDate)
```

Type

Method

Description

Sets the scheduled retention date for a document which will be archived.

Parameters

lObjectID	object ID
lObjectType	object type
strRetentionDate	retention date with the format **YYYY/MM/DD**

Return value

0	retention date successfully set
-1	retention date could not be set
-7	object type unknown
-22	an object with the specified ID does not exist
-38	invalid document type (a folder or register type was specified)
-40	invalid parameter passed

Comment

Error texts can be determined using GetLastError().

Options

Example

```
Dim a As Object

Set a = CreateObject("optimal_as.application")

a.SetPlannedRetention lObjectID, lObjectType, "12.10.2010"
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SetRelReferenceObject

Definition	SetRelReferenceObject (long long	lObjectID, lObjectType)
Type	Method	
Description	Sets the reference object specified in the system for working with relations to new values.	
Parameters	**lObjectID** **lObjectType**	object ID of the reference object type of the reference object
Return value	**0** no error	
Comment		
Options		
Example		

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SetResultListSelection

Definition	SetResultListSelection (String strObjectIDs)
Type	Method
Description	Selects objects in the active hit list.
Parameters	**strObjectIDs** IDs of the objects to be selected, separated by commas
Return value	Number of selected objects
Comment	The selection is only performed in the currently active window.
Options	
Example	

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SetSignatureProperty

Definition `SetSignatureProperty` (String strPropertyName, String strValue)

Type Method

Description Sets the property for the digital signature of a document.

Parameters

strPropertyName	name of the property
strValue	value of the property

Return value

Comment This function must be used immediately before executing SignDocument, SignDocumentEx, or SignDocumentList in order to define specific properties of the signature, e.g. the signature text.

The following properties can be set:

SIGTYP[0-n]	value=signature type
SIGTEXT[0-n]	value=signature text corresponding to the signature type
DEFAULTTYP	value=0-n, preset number
CAPTION	value=heading of the signature dialog
LOCATION	value=signature location
CERTFLAGS	value=possible values for certificate type filters: 64=allowed(ball pen signature) 128=digital signature (pencil signature) 192=both
ISSUERFILTER	Value=filter for certificates to be shown This refers to the issuer of a certificate.

Options

Example

```
Set a = createobject(“optimal_as.application”)
a.SetSignatureProperty "SIGTYP0", "Notice "
a.SetSignatureProperty "SIGTEXT0", "The content of the following …"
a.SetSignatureProperty "SIGTYP1", "Content correct"
a.SetSignatureProperty "SIGTEXT1", "The content of the following…"
a.SetSignatureProperty "DEFAULTTYP", "1"

a.SetSignatureProperty "CAPTION", "Electronic signature "
a.SetSignatureProperty "LOCATION", "Berlin "

a.SetSignatureProperty "CERTFLAGS", "192"
a.SetSignatureProperty "ISSUERFILTER", "OPTIMAL SYSTEMS"

a.SignDocument 4711,65554
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SignDocument

Definition	SignDocument	(long long	lObjectID, lObjectType)
Type	Method		
Description	Opens the digital signature dialog of a document.		
Parameters	**lObjectID** ID of the document to be signed **lObjectType** type of the document to be signed		
Return value	No return value		
Comment	The function calls the signature in an own thread. To obtain the return value the function GetSignDocumentResult can be used.		
Options			
Example			

```
Dim a As Object
Set a = CreateObject("optimal_as.application")

a.signdocument docID, docType

abort = False
Do While abort = False
    c = a.getsigndocumentresult()
    If c <> 101 Then
        abort = True
    End If
    DoEvents
Loop
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

SignDocumentEx

Definition	SignDocumentEx	(long	lObjectID,
		long	lObjectType,
		BOOL	bWaitForCompletion,
		Variant	vRetSignText)

Type Method

Description Opens the digital signature dialog of a document.

Parameters

lObjectID	ID of the document to be signed
lObjectType	type of the document to be signed
bWaitForCompletion	specifies whether the function will wait for the signature to be completed.
vRetSignText	the chosen signature text is returned here

Return value 0 signature successful, launched successfully
See **GetSignDocumentResult**

Comment The function calls the signature in an own thread. To obtain the return value the function GetSignDocumentResult can be used.

Options

Example

```
Dim a As Object
Set a = CreateObject("optimal_as.application")

a.signdocumentex docID, docType, False, signText

abort = False
Do While abort = False
    c = a.getsigndocumentresult(signText)
    If c <> 101 Then
        abort = True
    End If
    DoEvents
Loop
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

StartArchiveRequest

Definition StartArchiveRequest (String strFileName)

Type Starts a cabinet query

Description Provides the error text of the last occurred error.

Parameters **strFileName** name of the query file

Return value *file name* or **READY**, if QUERYWINDOW <> 0.
Empty string, if an error occurred.
Error text can be determined using **GetLastError()**.

Comment The query file must have the following structure:

```
[ANFRAGE]
SCHRANK =
CLAUSE1=
CLAUSE2=
...
...
CLAUSEn=
DATENFELDER=0(1)
DATAHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
VOLLTEXT=
```

For DATENFELDER=1 additionally a section called: [ANFRAGEFELDER] can be defined. It allows you to specify all fields to be displayed in the hit list of the file; thus only these fields will be queried.

Example:
The document type 'Krankendossier' has 3 fields: **form type**, **topic**, and **comment**. If only the values for **topic** and **comment** are meant to be queried, the section [QUERYFIELDS] would look like this:

```
[QUERYFIELDS]
Field0=topic
Field1=comment
```

The numbering of the fields must be continuous! This functionality also applies to the functions **StartRegRequest** and **StartDocRequest**.

If line breaks are contained in the returned field values, they are replaced with ASCII-character no. 17; so the programmer can decide whether to reinsert the line breaks when outputting the values in order to assure a proper output. The return file has the following structure:

```
INDEX1,OBJECTTYPEE
INDEX2,OBJECTTYPEE
...
...
INDEXn,OBJECTTYPE
```

An empty file is a valid result.

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options **DATENFELDER**:

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

If this option is switched on with DATAFIELDS=1, in addition to index and object type all object data is written into the file. The single columns are separated with the special character <TAB>.

A line in the result table would look like this:

```
1234,131071<TAB>Heiner<TAB>Lauterbach<TAB>Actor<CR><LF>
```

DATENHEADER*:*
If this option is switched on with DATAHEADER=1, then the first line of the result file will contain the column headings. The first line in the result table would look like this:

```
OSID,OSTYPE<TAB>First name<TAB>Last name<TAB>Occupation<CR><LF>
```

ANFRAGEFENSTER:
If a value for query window was specified, the following actions can be caused:
1 The query window will open if a query result is available. If no query result is available, nothing happens.
2 The query window will open if a query result is available. If no query result is available, a message will be displayed informing the user that the query did not return any results.

AUTOSTERN:
Specifies automatic adding of asterisks to query values.
Possible values:
0 autoasterisk setting as in the client
1 autoasterisk enabled
2 switch off autoasterisk.
3 autoasterisk at the end
33 autoasterisk at the front
35 autoasterisk at the front and end

Example Query file for a query in the cabinet **Customer**

```
[ANFRAGE]
SCHRANK=Kunde
CLAUSE1=Customer@Class=Customer
CLAUSE2=Customer@Vorlage=10.03.1997
CLAUSE3=Customer@optimal_AS=1
DATAHEADER=1
DATAFIELDS=1
```

Note: Logical expressions must be numbered sequentially. If there is a sequence gap, the following logical expressions will be ignored. Cabinet and field identifiers must exactly match the identifiers in the object definition.

The request data is not verified. SQL errors may occur if e.g. a date is expected and text was specified.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

StartDocRequest

Definition StartDocRequest (String strFileName)

Type Method

Description Starts a document query.

Parameters **strFileName** name of the query file

Return value ***File name*** or **READY**, if ANFRAGEFENSTER <> 0
Empty string, if an error occurred
Error text can be determined using **GetLastError()**.

Comment The query file must have the following structure:

```
[ANFRAGE]
SCHRANK=
DOKUMENT=
CLAUSE1=
CLAUSE2=
...
...
CLAUSEn=
DATENFELDER=0(1)
DATAHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
VOLLTEXT=
```

In order to create a register hit list, the register name must be specified. See also 'StartRegRequest'

The return file has the following structure:

```
INDEX1,OBJECTTYPEE
INDEX2,OBJECTTYPEE
...
...
INDEXn,OBJECTTYPE
```

An empty file is a valid result.

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options **DATENFELDER**:
If this option is switched on with DATAFIELDS=1, in addition to index and object type all object data is written into the file. The single columns are separated with the special character <TAB>. A line in the result table would look like this:

```
1234,131071<TAB>Heiner<TAB>Lauterbach<TAB>Actor<CR><LF>
```

DATENHEADER*:*
If this option is switched on with DATAHEADER=1, then the first line of the result file will contain the column headings. The first line in the result table would look like this:

```
OSID,OSTYPE<TAB>First name<TAB>Last name<TAB>Occupation<CR><LF>
```

ANFRAGEFENSTER:
If a value for query window was specified, the following actions can be caused:

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

1 The query window will open if a query result is available. If no query result is available, nothing happens.
2 The query window will open if a query result is available. If no query result is available a message will be displayed informing, that the query did not return any results.

AUTOSTERN:
Specifies automatic adding of asterisks to query values.
0 autoasterisk setting as in the client
1 autoasterisk enabled
2 switch off autoasterisk
3 autoasterisk at the end
33 autoasterisk at the front
35 autoasterisk at the front and end

TYP=2(0,1)
Specifies, what type of hit list will be created.
0 folder hit list
1 register hit list
2 document hit list
If the type is not specified, a document hit list will be created.

LOKALESUCHE=0,1,2
Specifies, if documents without register reference will be included in the query. If this value is 2, the user setting is applied to the client application.
Furthermore it can be specified in as.cfg, if this value (the setting in the client) will be automatically included in the query file. To do so the following needs to be added in the CLIENT section:

```
AUTOLOCALSEARCH=1
```

Example

Query file for a query for the document type **InboundDocument** in the cabinet **Customer**

```
[ANFRAGE]
SCHRANK=Kunde
REGISTER=Register
DOKUMENT=InboundDocument
CLAUSE1=Customer@Name=Müller
CLAUSE2= Register@Type=job
CLAUSE3=Eingangsbeleg@Vorlage=10.03.1997
DATAHEADER=1
DATAFIELDS=1
```

Note: Clauses must be numbered sequentially. If there is a sequence gap, the following logical expressions will be ignored.

Cabinet and field identifiers must exactly match the identifiers in the object definition.

The request data is not verified. SQL errors may occur if e.g. a date is expected and text was specified.

See also: StartArchiveRequest

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

StartRegRequest

Definition StartRegRequest (String strFileName)

Type Method

Description Starts a register query.

Parameters **strFileName** name of the query file

Return value **File name** or **READY**, if ANFRAGEFENSTER <> 0
Empty string, if an error occurred.
Error text can be determined using **GetLastError()**.

Comment The query file must have the following structure:

```
[ANFRAGE]
SCHRANK=
REGISTER=
CLAUSE1=
CLAUSE2=
...
...
.CLAUSEn=
DATENFELDER=0(1)
DATAHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
VOLLTEXT=
```

The return file has the following structure:

```
INDEX1,OBJECTTYPEE
INDEX2,OBJECTTYPEE
.
.
.
INDEXn,OBJECTTYPE
```

An empty file is a valid result.

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options **DATENFELDER**:
If this option is switched on with DATAFIELDS=1, in addition to index and object type all object data is written into the file. The single columns are separated with the special character <TAB>. A line in the result table would look like this:

```
1234,131071<TAB>Heiner<TAB>Lauterbach<TAB>Actor<CR><LF>
```

DATENHEADER:
If this option is switched on with DATAHEADER=1, then the first line of the result file will contain the column headings. The first line in the result table would look like this:

```
OSID,OSTYPE<TAB>First name<TAB>Last name<TAB>Occupation<CR><LF>
```

ANFRAGEFENSTER:
If a value for query window was specified, the following actions can be caused:
1 The query window will open if a query result is available. If no query result is available, nothing happens.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

2 The query window will open if a query result is available. If no query result is available, a message will be displayed informing the user that the query did not return any results.

AUTOSTERN:
Specifies automatic adding of asterisks to query values.
0 autoasterisk setting as in the client
1 autoasterisk enabled
2 switch off autoasterisk
3 autoasterisk at the end
33 autoasterisk at the front
35 autoasterisk at the front and end

Example Query file for a query in the **Register** customer register

```
[ANFRAGE]
SCHRANK=Kunde
REGISTER=Register
CLAUSE1=Customer@Name=Müller
CLAUSE2=Register@Typ=Auftrag
CLAUSE3=Customer@optimal_AS=1
DATAHEADER=1
DATAFIELDS=1
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Note: Logical expressions must be numbered sequentially. If there is a sequence gap, the following logical expressions will be ignored. Cabinet and field identifiers must exactly match the identifiers in the object definition.

The request data is not verified. SQL errors may occur if e.g. a date is expected and text was specified.

See also: StartArchiveRequest

StartClientRequest

Definition StartClientRequest (string request)

Type Method

Description Starts the formulated client query.
Full text hit lists are shown with facets.

Parameter

request	Query in JSON format

Comment JSON description

request	start object
objecttype	UINT, object type of the query
fulltext	string, full text query string
fields	array, JSON elements of type 'field'
field	consists of:
iname	string, internal name of the queried field
value	string, value of the queried field

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Return value

0	query executed
-125	a modal dialog is open.
-140	JSON has no member 'request'
-141	member 'request' has no member 'objecttype'
-143	member 'objecttype' has unexpected data type
-144	member 'objecttype' is invalid
-145	specified object type is unknown
-146	specified object type is not in full text index
-147	member 'fulltext' has unexpected data type
-148	member 'fulltext' has no value
-149	JSON contains parser errors
-150	member 'fields' is not an array
-151	array member 'iname' or 'value' is missing
-152	array member 'iname' or 'value' has no value
-153	specified field not found

Exact error text can be determined using **GetLastError()**.

Example

```
Dim ax
Dim ret
Dim req
set ax = CreateObject("optimal_AS.Application")

'10.1.4.159#4000
req = "{""request"": {""objecttype"": 393226,""fulltext"":
""optimal"",""fields"": [{""iname"": ""MAIL_FROM"",""value"":
""multi*""}]}}"
ret = ax.StartClientRequest(req)

if ret <> 0 then
    MsgBox ax.GetLastError() & " (" & ret & ")"
else
    ax.ActivateApp 1
end If

set ax = nothinget Application = nothing
```

StoreNotice

Definition	StoreNotice	(long	lObjectID,
		long	lObjectType,
		String	strFileName)

Type Method

Description Saves a note for an existing object (folder, register, document).

Parameters

lObjectID	object ID
lObjectType	object type
strFileName	file name of the note with 'txt' as file extension

Return value

0	no error
-1	no destination object specified
-3	cabinet unknown
-7	document type unknown
-14	folder identifier unknown
-15	document identifier unknown
-32	note file does not exist
-41	file name empty

Error text can be determined using **GetLastError()**.

Comment The note must be provided as an ASCII text file with 'txt' as file extension. Independently of the success or failure of the function, the given file will not be deleted by the archiving system. If it will be no longer required after the function call, it is recommended to have it deleted by the calling application.

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break. If, instead of a file, a string is passed and notes are stored as files on the server, the client itself creates the file.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

TransformXML

Definition	TransformXML	(String	strXMLFile,
		String	strXSLFile,
		VARIANT	varRetString)

Type Method

Description Converts an XML file using the XSL file into the specified format.

Parameters

strXMLFile	full path and file name of the XML file
strXSLFile	full path and file name of the XSL file
varRetString	the converted object is returned here as a string

Return value

0	no error
-32	one of the two source files does not exist
-100	files could not be converted
-101	XML file could not be loaded
-102	XSL file could not be loaded
-103	XML object could not be created

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

UndoCheckOut

Definition	UndoCheckOut	(long long	lDocID, lDocType)
Type	Method		
Description	Undoes the check-out procedure for actual user.		

Parameters

lDocID	document ID
lDocType	document type

Return value

0	no error
-1	undo failed
-7	unknown document type
-53	document was not checked out
-56	document name could not be determined in cache

Error text can be determined using **GetLastError()**.

Comment

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

UnlinkDocuments

Definition

```
UnlinkDocuments   (long  lObjectID1,
                   long  lObjectType1,
                   long  lObjectID2,
                   long  lObjectType2 )
```

Type

Method

Description

Deletes a link between two documents.

Parameters

lObjectID1	ID of the first document
lObjectType1	type of the first document
lObjectID2	ID of the second document
lObjectType2	type of the second document

Return value

0	no error
-29	one or both object IDs are unknown
-86	object indexes are identical
-88	objects from the filing tray are not allowed

Error text can be determined using **GetLastError()**.

Comment

This function is available from 4.00 SPII. The link between two documents is saved in their notes.

Options

Example

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

UpdateArchiveData

Definition

UpdateArchiveData (String strFilename,
long *lpReturnObjectID,
long *lpReturnObjectType)

Type

Method

Description

Performs an update of the folder data.

Parameters

strFilename	handoff file (for structure see comment)
***lpReturnObjectID**	the object index is returned here if insert is successful
***lpReturnObjectType**	the object type is returned here if insert is successful.

Return value

0	if no error
-3	cabinet unknown
-10	folder identifier missing
-14	folder identifier unknown
-18	update failed
-24	field names could not be resolved
-28	invalid field value for a given field
-30	one or more mandatory fields not filled out
-31	data type of entered value is not compatible with the data type of the associated field.
-32	handoff file does not exist
-47	no write permission for this object
-64	server error occurred

Error text can be determined using **GetLastError()**.

Comment

The handoff file has the following structure:

```
[AKTUALISIEREN]
SCHRANK=
SCHRANK-ID=
FIELD1=
FIELD2=
...
...
FIELDn=
Mode=1
```

If mode = 1, all non-specified fields are not reset. See annotation 'WARNING'

The following lines must be inserted in order to update the table controls, where the separator character equals chr(17).

```
[Table@TABLENAME]
Mode=0[1] 0 = Tabelle leeren, 1 = Tabelle anhängen
Line0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Line1={Column1Row2(Separator)Column2Row2}
```

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options

PFLICHTFELDER=[0;1]
Mandatory fields are verified for the value 1.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Example Updating the 'Customer' folder

```
[AKTUALISIEREN]
SCHRANK=Kunde
SCHRANK-ID=4711
FIELD1=Class=Customer
FELD2=Vorlage=10.03.1997
FELD3=optimal_AS=0
```

Note: If possible, all fields should be specified for every update. Fields that are not specified will have no content after the update. Preset fields are disregarded. Key fields are not verified!

UpdateArchiveDataS

Corresponds to 'UpdateArchiveData' and can be used from VBScript.

Definition

```
UpdateArchiveDataS (String    strFilename,
                    VARIANT*  varReturnObjectID,
                    VARIANT*  varReturnObjectType)
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

UpdateDocFileList

Definition UpdateDocFileList (String strFileName)

Type Method

Description Allows the file list of a document to be changed.

Parameters **strFileName** file name (for structure see comment)

Return value

0	no error
-2	no cabinet specified
-3	cabinet unknown
-6	no document type specified
-7	document type unknown
-9	document type does not belong to the specified cabinet
-11	document identifier missing
-21	document already archived
-32	handoff file does not exist
-41	file name not specified
-64	server error occurred

Error text can be determined using **GetLastError()**.

Comment The handoff file has the following format:

```
[DATEILISTE]
SCHRANK=
DOKUMENT=
DOKUMENT-ID=
MAINTYPE=
DATEI1=
DATEI2=
   .
   .
DATEIn=
ARCHIVIERBAR=[0;1]
```

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Note: The main type of the destination document must be defined with **MAINTYPE=#** where # stands for the main type no. . Possible main types are described in the chapter **Introduction**.

Options

ARCHIVIERBAR=[0;1]
The archive status of the document can be changed here.

0	not archivable
1	archivable

1 is the default value

DATEILÖSCHEN=[0;1]
Deletes the specified source files.
0 is the default value

Example Source code example:

```
HelpStr=MyAX. UpdateDocFileList(strFile)
```

Note: The given file names must not be identical with the file names that have already been assigned to the document. Otherwise data loss will occur.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

The file list can only be changed if the document has not been archived yet!

UpdateDocShare

Definition	UpdateDocShare	(long	lIdent,
		long	lType,
		long	IUser)

Type Method

Description Edits the document share of the specified document for the specified user

Parameter

lIdent document ID
lType document type
Rights preset rights (RWXU)
IUser user ID of the user of the document share

Return value

0 "Document share edited or editing canceled."
-125 "A modal dialog is open."
-130 "Specified DMS object type unknown."
-131 "Specified DMS object type is not a document."
-132 "Only 'RXWU' rights are permitted."
-133 "Specified document does not exist or was deleted."
-134 "Document shares not possible."
-135 "The required system role to share documents does not exist."
-136 "Specified user does not exist."
-139 "Specified object was not shared by logged-in user."

Exact error text can be determined using **GetLastError()**.

Comment -

Example

```
dim Application : set Application =
createobject("optimal_AS.application")
Dim sRet


Application.ActivateApp 1

sRet = Application.UpdateDocShare(590, 262146, 15475)

if (sRet <> 0) then
       MsgBox Application.GetLastError() & " (" & sRet & ")"
else
       MsgBox "Document share edited or editing canceled."
end If

set Application = nothing
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

UpdateDocumentData

Definition UpdateDocumentData (String strFilename)

Type Method

Description Performs an update of the document data.

Parameters **strFilename** handoff file (for structure see comment)

Return value

0	if no error
-2	no folder specified
-3	folder unknown
-4	no register specified
-5	register unknown
-7	unknown document type
-8	register does not belong to the specified folder
-11	document identifier missing
-13	register identifier unknown
-16	update failed
-24	field names could not be resolved
-28	invalid field value for a given field
-30	one or more mandatory fields not filled out
-31	data type of entered value is not compatible with the data type of the associated field.
-32	handoff file does not exist
-47	no write permission for this object
-64	server error occurred

Error text can be determined using **GetLastError()**.

Comment The handoff file has the following structure:

```
[AKTUALISIEREN]
SCHRANK=
DOKUMENT=
DOKUMENT-ID=
FIELD1=
FIELD2=
      .
FIELDn=
Mode=1
```

If mode = 1, all non-specified fields are not reset. See also the **note** on UpdateArchiveData
The following lines must be inserted in order to update the table controls, where the separator character equals chr(17).

```
[Table@TABLENAME]
Mode=0[1] 0 = Tabelle leeren, 1 = Tabelle anhängen
Line0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Line1={Column1Row2(Separator)Column2Row2}
```

Example:

```
[Tabelle@Protokoll]
Mode=1
Line0={01.02.2013(Separator)Mustermann}
Line1={04.02.2013(Separator)Müller}
Line2={05.02.2013(Separator)Meyer}
```

The main type of documents which are part of the document history must not be changed.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options

PFLICHTFELDER=[0;1]
Mandatory fields are verified for the value 1.

REFRESHMULTIFIELDS=[0;1]
If the value is 1, all entries of the multiple fields, for which new values in the section MULTI_... were defined, are deleted. If the handoff file contains the value REFRESHMULTIFIELDS=1 and a MULTI_Field1 section was defined, then all entries are deleted for this field and values entered in this section are inserted.
The database name of the multiple field stands for the update of multiple fields MULTI_... for ...

```
DATA1=Seitennummer,Wert
DATAn=Seitennummer,Wert
```

ARCHIVIERBAR = [0;1]
here the archive status of the document can be changed.
0 not archivable
1 archivable

SHOWTEMPLATES=[0;1]
The template dialog for W_Documents is displayed if the value is 1 . The selected template is inserted.

ENABLEOPTIONS=[0;1]
If the value is 1, the options of the template dialog are activated.

Example

E.g. Updating the **inbound document** customer document

```
[AKTUALISIEREN]
SCHRANK=Kunde
REGISTER=Register
DOKUMENT-ID=4713
FELD1=Typ=Brief
FIELD2=Text=Hallo
[MULTI_FIELD1]
DATA1=1,Peter
DATA2=2,Hans
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

UpdateRegisterData

Definition UpdateRegisterData (String strFilename)

Type Method

Description Performs an update of the register data.

Parameters **strFilename** handoff file (for structure see comment)

Return value

0	if no error
-2	no folder specified
-3	folder unknown
-4	no register specified
-5	register unknown
-8	register does not belong to the specified folder
-12	register identifier missing
-13	register identifier unknown
-16	update failed
-24	field names could not be resolved
-28	invalid field value for a given field
-30	one or more mandatory fields not filled out
-31	data type of entered value is not compatible with the data type of the associated field.
-32	handoff file does not exist
-47	no write permission for this object
-64	server error occurred

Error text can be determined using **GetLastError()**.

Comment The handoff file has the following structure:

```
[AKTUALISIEREN]
SCHRANK=
REGISTER=
REGISTER-ID=
FIELD1=
FIELD2=
      .
      .
FIELDn=
Mode=1
```

If mode = 1, all non-specified fields are not reset. See also the note on UpdateArchiveData

The following lines must be inserted in order to update the table controls, where the separator character equals chr(17).

```
[Table@TABLENAME]
Mode=0[1] 0 = Tabelle leeren, 1 = Tabelle anhängen
Line0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Line1={Column1Row2(Separator)Column2Row2}
```

The **strFileName** parameter value can be a file name or instead a string containing the content of the file which would be passed. If file content is passed as a string, each line must end with a line break.

Options **PFLICHTFELDER=[0;1]**
Mandatory fields are verified for the value 1.

Example Customer register update

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

```
[AKTUALISIEREN]
SCHRANK=Kunde
REGISTER=Register
REGISTER-ID=4711
FELD1=Typ=Brief
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

COM Interface of the Preview Window/Dashlets

Introduction

The details preview, content preview, and dashlets of the client can be addressed and controlled using a COM interface. This interface can either be accessed over the 'application' interface or used directly as long as this happens within an event. Within events this object can be addressed under the name 'InfoWindow'.

All COM Commands

Caption

Definition	String caption
Type	Property (read/write)
Description	Sets the window title or returns it.
Parameter	-
Return value	-
Example	

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.Caption = “This is the caption”
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Visible

Definition	Boolean visible
Type	Property (read/write)
Description	Sets the visibility of the window
Parameter	-
Return value	-
Example	

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.Visible = false ‘makes the window invisible
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

URL

Definition String URL

Type Property (read/write)

Description Sets the URL to be displayed.

Parameter -

Return value -

Example

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.Url = „www.google.de“
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Closeable

Definition Boolean closeable

Type Property (read/write)

Description Specifies if the window can be closed by the user.

Parameter -

Return value -

Example

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.Closeable = false
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

HtmlDocument

Definition	Object HtmlDocument
Type	Property (read)
Description	Provides the HTML-DOM document which is currently displayed.
Parameter	-
Return value	-
Comment	The HTML document interface is described on MSDN.
Example	

```
Dim a as object
Set a = createobject(“optimal_as.application”)
Set doc = a.InfoWindow.HtmlDocument
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

EnableContextMenu

Definition Boolean EnableContextMenu

Type Property (read/write)

Description Specifies if the context menu can be displayed

Parameter -

Return value -

Example

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.EnableContextMenu = false
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Refresh

Definition	Refresh()
Type	Method
Description	Updates the view in the window
Parameter	-
Return value	-
Example	

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.Refresh
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GoBack

Definition	GoBack()
Type	Method
Description	Navigates to the previous URL.
Parameter	-
Return value	-
Example	

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.URL = “www.google.de“
a.InfoWindow.URL = “www.test.de”
a.InfoWindow.GoBack
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

GoForward

Definition GoForward()

Type Method

Description Navigates in the history to the next URL.

Parameter -

Return value -

Example

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.URL = “www.google.de“
a.InfoWindow.URL = “www.test.de”
a.InfoWindow.GoBack
a.InfoWindow.GoForward
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

ShowHtml

Definition	ShowHtml(String strHtml)
Type	Method
Description	Displays the passed HTML string.
Parameter	-
Return value	-
Example	

```
Dim a as object
Set a = createobject(“optimal_as.application”)
a.InfoWindow.ShowHtml “<html></head><body>Test</body></html>”
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Script Interface of the Dashlets

Introduction

This interface allows you to access enaio® client functionality from preview windows/dashlets, for example to refresh a hit list or open a data sheet. All functions can be used in multiple iterations.

Functions to Browse Across Documents

osjxCanNextDoc

Definition	Boolean osjxCanNextDoc	
Type	Property (read/write)	
Description	Indicates whether there is a next document in the current hit list.	
Parameter	-	
Return value	**1**	There is a next document in the current hit list.
	0	There is no next document in the current hit list.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxCanPrevDoc

Definition	Boolean osjxCanPrevDoc
Type	Property (read/write)
Description	Indicates whether there is a previous document in the current hit list.
Parameter	-
Return value	**1** There is a previous document in the current hit list. **0** There is no previous document in the current hit list.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxNextDoc

Definition	osjxNextDoc()
Type	Method
Description	Sets the focus on the next document.
Parameter	-
Return value	-

osjxPrevDoc

Definition	osjxPrevDoc()
Type	Method
Description	Sets the focus on the previous document.
Parameter	-
Return value	-

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Functions to Control Client Windows

osjxOpenDataSheet

Definition	osjxOpenDataSheet	(long BOOL	lObjectID, bReadOnly)
Type	Method		
Description	Opens the data sheet of a specified object.		
Parameters	**lObjectID** object ID **bReadOnly** **TRUE** if the document is to be shown read-only.		
Return value	-		
Comment	-		

osjxOpenObject

Definition	osjxOpenObject(long lObjectID)
Type	Method
Description	Opens the document or the folder of an object.
Parameters	**lObjectID** object ID
Return value	-

Comment	-

osjxOpenLocation

Definition	osjxOpenLocation(long lObjectID)
Type	Method
Description	Opens the location of a selected object.
Parameters	**lObjectID** object ID
Return value	-
Comment	-

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxOpenLocationsAndLinks

Definition	osjxOpenLocationsAndLinks(long lObjectID)
Type	Method
Description	Opens links and references of an object.
Parameters	**lObjectID** object ID
Return value	-

osjxOpenObjectHistory

Definition	osjxOpenObjectHistory(long lObjectID)
Type	Method
Description	Opens the editing history of an object.
Parameters	**lObjectID** object ID
Return value	-

osjxAddSignature

Definition	osjxAddSignature(long lObjectID)
Type	Method
Description	Adds an electronic signature.
Parameters	**lObjectID** object ID
Return value	-

osjxPrintObject

Definition	osjxPrintObject()
Type	Method
Description	Prints a document.
Parameter	-
Return value	-

osjxOpenObjectRemarks

Definition	osjxOpenObjectRemarks(long lObjectID)
Type	Method
Description	Opens the notes of an object.
Parameters	**lObjectID** object ID
Return value	-

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxAddFollowUp

Definition	osjxAddFollowUp(long lObjectID)
Type	Method
Description	Adds a follow-up.
Parameters	**lObjectID** object ID
Return value	-

osjxAddSubscribe

Definition	osjxAddSubscribe(long lObjectID)
Type	Method
Description	Adds a subscription.
Parameters	**lObjectID** object ID
Return value	-

osjxStartWorkflow

Definition	osjxStartWorkflow	(long String	lObjectID, strWorkflowModell)

Type	Method
Description	Starts a workflow with a given object, based on the model name.
Parameters	**lObjectID** object ID **strWorkflowModell** Name of the workflow model
Return value	-

osjxGetSelectedObjects

Definition	osjxGetSelectedObjects()
Type	Method
Description	Finds desired objects in a hit list.
Parameter	-
Return value	List of selected objects: `Obj0-ID,Obj0-Type;…;Obj(n)-ID,Obj(n)-Type`

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxRefreshObjectInLists

Definition	osjxRefreshObjectInLists (long lObjectID)
Type	Method
Description	Updates the specified DMS object in all open lists.
Parameter	**lObjectID** object ID
Return value	-
Comment	-

Example	if (window.osClient)
	{
	window.osClient.osjxRefreshObjectInLists(312);
	}
	else
	{
	alert("window.osClient not exist");
	}

osjxByteArrayToFile

Definition	osjxByteArrayToFile(JaveScriptArray, sting sDateiname)
Type	Method
Description	Takes a JavaScript array with integer values and writes it to a file.

Parameter	**JavaScript array** **sFileName**
Return value	-
Comment	If the file name is specified without a unique path, a 'Save as' dialog will be opened.

Example	`if (window.osClient)`
	`{`
	`var arr = new Array(10);`
	
	`arr[0] = 0;`
	`arr[1] = 1;`
	`arr[2] = 2;`
	`arr[3] = 3;`
	`arr[4] = 4;`
	
	`window.osClient.osjxByteArrayToFile(arr, "datei.pdf");`
	`}`
	`else`
	`{`
	`alert("window.osClient not exist");`
	`}`

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxURLDownloadToFile

Definition	osjxURLDownloadToFile(string sDateiUrl, sting sDateiname)
Type	Method
Description	Takes a file URL and writes the contents of the file URL to a file.
Parameter	**sFileUrl** **sFileName**
Return value	-
Comment	If the file name is specified without a unique path, a 'Save as' dialog will be opened.

Example	`if (window.osClient)`
	`{`
	`window.osClient.osjxURLDownloadToFile`
	`("http://www.url.de/bild.jpg", "urlbild.jpg");`
	`}`
	`else`
	`{`
	`alert("window.osClient not exist");`
	`}`

osjxOpenResultList

Definition	osjxOpenResultList(sting sTitel, JSON-String)
Type	Method

Description	Opens a hit list with the specified title and the specified DMS objects.
Parameter	**sTitle** **JSON string**
Return value	-

Example	JSON string:
	`{`
	`"title": "meine Ordner-Trefferliste",`
	`"hits": [{`
	`"id": "411",`
	`"type": "8"`
	`},`
	`{`
	`"id": "577",`
	`"type": "8"`
	`}]`
	`}`
	
	Call:
	`if (window.osClient)`
	`{`
	`window.osClient.osjxOpenResultList("{\"title\":`
	`\"Trefferliste\",\"hits\":[{\"id\":\"411\",\"type\":\"`
	`8\"},{\"id\":577,\"type\":\"8\"}]}");`
	`}`
	`else`
	`{`
	`alert("window.osClient not exist");`
	`}`

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Functions for Dashlet Control

Dashlets can be identified in three ways:

1. No parameter: The current dashlet
2. Dashlet title: The title given when setting up the dashlet in enaio® enterprise-manager. The system-side dashlets 'DocumentViewer' and 'DetailsViewer' can be accessed via the titles 'OSDOCUMENTVIEWER' and 'OSDETAILVIEWER.'
3. Dashlet number (1 – 10)

This function parameter is henceforth referred to as 'dashletID.'

osjxOpenChromeDEVTools

Definition	osjxOpenChromeDEVTools()
Type	Method
Description	Opens DEV tools of the Chrome browser.
Parameter	-
Return value	-

osjxCloseChromeDEVTools

Definition	osjxCloseChromeDEVTools()
Type	Method
Description	If the DEV tools of the Chrome browser are opened, they can be closed using this method.
Parameter	-
Return value	-

osjxIsDashletVisible

Definition	osjxIsDashletVisible(dashletID)
Type	Method
Description	Determines if the specified dashlet is visible.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed.
Return value	**1** dashlet is visible **0** dashlet is not visible

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxSetDashletVisible

Definition	osjxSetDashletVisible(dashletID, bool bVisible)
Type	Method
Description	Enables/disables visibility of the specified dashlet. If it is not visible, it can also no longer be accessed via the 'View' ribbon in enaio® client.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed. **bVisible** 0 = not visible / 1 = visible
Return value	-

osjxIsDashletEnabled

Definition	osjxIsDashletEnabled(dashletID)
Type	Method
Description	Determines if the specified dashlet is enabled.

Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed.
Return value	**1** dashlet is enabled **0** dashlet is not enabled

osjxSetDashletEnabled

Definition	osjxSetDashletEnabled(dashletID, bool bEnabled)
Type	Method
Description	Sets the specified dashlet to enabled or disabled If it is disabled, 'ContextChanges' are ignored.
Parameter	**dashletID** (see above) Dashlet name as a string, or dashlet number. If unspecified, the current dashlet will be addressed. **bEnabled** 0 = disabled / 1 = enabled
Return value	-

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxGetDashletURL

Definition	osjxGetDashletURL(dashletID)
Type	Method
Description	Determines the URL of the specified dashlet.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed.
Return value	URL of the dashlet as a string

osjxSetDashletURL

Definition	osjxSetDashletURL(dashletID, string sUrl)
Type	Method
Description	Sets the URL of the specified dashlet. If no URL is specified, the originally configured URL is used.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed. **sUrl** URL of the dashlet as a string
Return value	-

osjxGetDashletPaneState

Definition	osjxGetDashletPaneState(dashletID)
Type	Method
Description	Determines the display status of the specified dashlet pane.
Parameter	**dashletID** (see above) Dashlet name as a string, or dashlet number. If unspecified, the current dashlet will be addressed.
Return value	0: unknown 1: closed 2: hidden 3: floating

osjxSetDashletPaneState

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Definition	osjxSetDashletPaneState(dashletID, long lStatus)
Type	Method
Description	Sets the display status of the specified dashlet pane. If the specified dashlet is set to invisible using 'osjxSetDashletVisible', the call is ignored.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed. **lStatus**: 0: unknown 1: closed 2: hidden 3: floating
Return value	-

osjxSetDashletCaption

Definition	osjxSetDashletCaption(dashletID, string sTitle)
Type	Method
Description	Specifies a title for a dashlet.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed. **sTitle**
Return value	-
Example	`if (window.osClient)` `{`

```
window.osClient.osjxSetDashletCaption
("OSDETAILVIEWER", "Neuer Titel");
}
else
{
alert("window.osClient not exist");
}
```

osjxGetDashletCaption

Definition	osjxGetDashletCaption(dashletID)
Type	Method
Description	Determines the title of a dashlet.
Parameter	**dashletID** (see above) Dashlet title as a string, or dashlet number. If unspecified, the current dashlet will be addressed.
Return value	Title of the dashlet.
Example	

```
if (window.osClient)
{
var State =
window.osClient.osjxGetDashletCaption("OSDETAILVIEWER"
);

alert(State);
}
else
{
alert("window.osClient not exist");
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

osjxGetEnvironment

Determines the values known from the COM function 'GetEnvironment'.

Appendix: Configuring the enaio® Printers

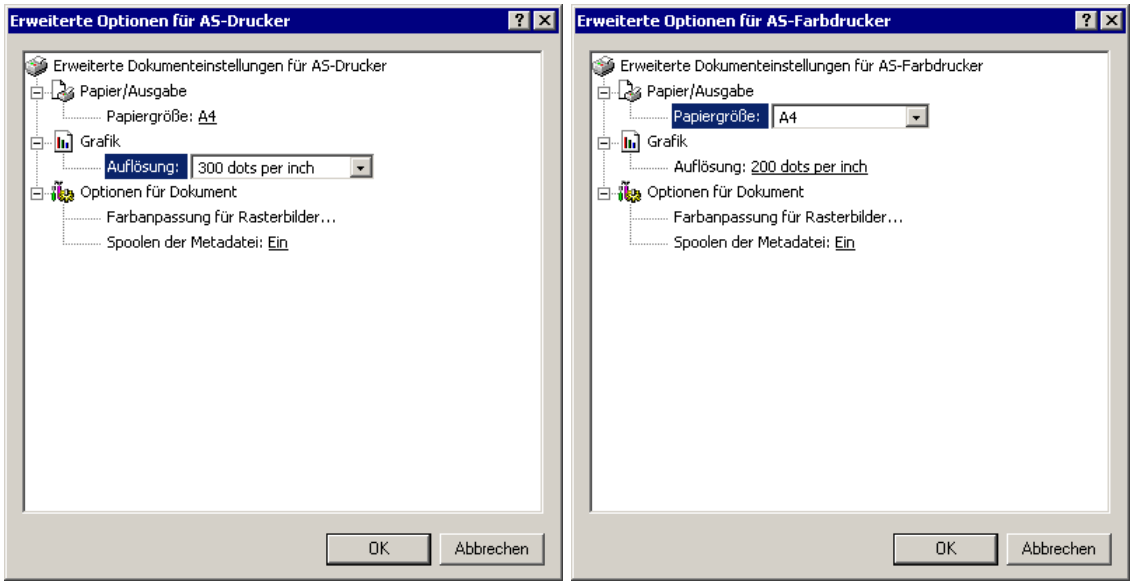
Introduction

When installing enaio® client, optionally two printer drivers can be installed at the workstation:

§ OS-printer `axprint.dll` for black/white printing

§ the OS color printer `axcprint.dll` for color printing

If you have not installed these printer drivers, you can obtain a printer driver setup upon request.
Using these printer drivers, image files can be created from any application with printing abilities and passed on to enaio® client. enaio® client opens the corresponding dialogs in which the user can specify a location, document type, and indexing. It is required to have enaio® client launched. If it has not been launched an information dialog will open. The user can then launch enaio® client and continue the process, or cancel printing.
The printer drivers can also be used to save printing data as image or PDF-files in the file system. The configuration file `asprint.ini` will be required.
This configuration file must be created by the corresponding application and saved in the Windows directory. Usually, after printing the configuration file must be deleted.
Paper size and resolution are not configured in the configuration file but as usual in the properties dialogs of the printer drivers.



Note: The OS color printer must not be set to grayscale printing. This setting will cause errors.

The Configuration File

The printer drivers locate the configuration file `asprint.ini` in the Windows directory before creating any files.
Only if the file does not exist, is an attempt made to pass on image files to enaio® client. If the file is found, the configuration data will be read.
You can create the file with any text editor. It must be labeled `asprint.ini` and saved in the Windows directory.
Enter the section label `[ASPRINT]` in the first row of the file. Followed by the parameters:

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Parameters	Possible values	Description
FILE=	<Pfad\Dateiname>.000	Enter path, file name and extension. The printer drivers always use an automatic hexadecimal extension if several pages are created as single files. The first page has the extension '000'.
	<Pfad\Dateiname>.tif	If you specify the extension 'tif', the AS-printer will use this extension if all pages are saved as one multipage TIFF G4 file. Using MULTIPAGE=1 creates one file for each print job. Otherwise automatically the extension 000
	<Pfad\Dateiname>.pdf	If you specify the extension 'pdf', the OS-printer and OS color printer will use this extension if all pages are saved as one PDF file.
HINTERGRUND1=	<Pfad\Dateiname>.tif	The OS-printer can include background images. The files must be in TIFF G4 format and it is recommended that they have the same resolution as the one that is defined for the OS printer. This parameter defines the background image for the first page.
HINTERGRUND2=	<Dateiname>.tif	This parameter defines the background image for all following pages.
MULTIFILE=	0	If there are already files with the same label, they are not overwritten. The print job will be canceled. The result code '-8 – file already exists' is inserted into the log file.
	1	Depending on the parameter 'Multipage' files with the same label are not overwritten, but their extension is incremented or the file is extended.
MULTIPAGE=	0	A file is created for each page. The extension starts as hexadecimal '000' and is incremented. Using the parameter 'MULTIFILE=1' the printer drivers determine if files of the same name already exist and in such case continue with the numbering of the extensions. With parameter 'MULTIFILE=0' the print job is canceled if files with the same extension already exist.
	1	The OS-printer creates a multipage TIFF G4 file. If this file already exists, new pages are added. If the parameter has the value '1', the file is created with a PDF header and can be opened in a PDF viewer. The OS color printer creates a PDF-file with all pages if the 'PDF' parameter has the value '1'. If this file already exists, new pages are added. If the 'PDF' parameter has the value '0', a JPEG file is created for each page. The extension starts as hexadecimal '000' and is incremented.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Parameters	Possible values	Description
PDF=	0	The OS-printer creates image files in TIFF G4 format. The OS color printer creates color images in JPEG-format. The default setting is '0'.
	1	The created image files get a PDF-header and can be opened in a PDF viewer.
GRAY=	0	Default setting of the OS color printer, color images are created.
	1	The OS color printer creates grayscale images in JPEG-format.
COMPRESSION=	1-100	The level of compression can be adjusted for the OS color printer. A value of '100' denotes maximum compression. The default setting is '1' – minimal compression.
EXE=	<Pfad\Programm>	Path and label of a program that will be started after the print job. On launch, the parameters are passed to the program in quotation marks. § Path and label of the configuration file `asprint.ini` § Log entries (see Logging) In addition the user name is entered into the `asprint.ini`: USERNAME='Windows user name'
CITRIXMODE=	1	This entry is only required for a terminal server installation (see Terminal server)

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Example:

```
[ASPRINT]
FILE=C:\ASDRUCK\print.tif
BACKGROUND1=C:\ASDDRUCK\HG1.tif
HINTERGRUND2C:\ASDDRUCK\HGff.tif
MULTIFILE=1
MULTIPAGE=1
PDF=0
```

The OS-printer creates the file `print.tif`. The format is multipage TIFF G4. The file is extended in case of the following print jobs.

Logging

The enaio® printer drivers create a LOG file with a result number and a description. The LOG-file has the same label as the image file and the file extension 'LOG'. It can be opened with any editor.

Number	Description
1	"Pages created successfully."
-1	"Incorrect or invalid parameters."
-2	"Not enough memory."
-3	"Error while locking memory area."
-4	"Error creating a file."
-5	"File does not exist or could not be opened."
-6	"Error reading a file."
-7	"Error writing a file."
-8	"File already exists."

-9	"Incorrect or unsupported file format."
-10	"Incorrect or unsupported TIFF format."
-23	"Error creating a directory."
-24	"Invalid destination directory."
-25	"Invalid source directory."
-26	"Invalid temporary directory."
-27	"Invalid drive."
-28	"Not enough disk space."
-29	"Error changing directory."
-30	"Error deleting a directory."
-31	"Error determining file sizes in a directory."
-44	"Error while loading a DIB."
-45	"DIB is faulty or does not exist."
-46	"Error creating a bitmap."
-73	"No file name available for background bitmap."
-74	"File does not contain image data."
-78	"Unknown error."
-79	"File does not exist."
-80	"Source file could not be opened."
-81	"Destination file could not be opened."
-82	"Source file does not exist."
-83	"Destination file does not exist."
-85	"Error creating a slide."
-91	"Action canceled by user."

Example:

```
-8: file C:\ASDRUCK\print.000 already exists
```

If an application is run after printing, this content, the path, and the name will be passed to the configuration file `asprint.ini` as a start parameter.

Terminal server

When installing terminal servers both printer drivers require the configuration file `asprint.ini`. The path to the configuration file is entered in the registry under:

`HKEY_LOCAL_MASCHINE\SOFTWARE\Optimal Systems`

There create the key 'asprinter'. Enter 'directory' for string and as value a directory that will be accessible for the user.
Places the configuration file `asprint.ini` in this directory. The parameter 'CITRIXMODE=1' will be required.
Enter any file label for the 'file' parameter. Path and file name are not process, but possibly the file extension.
The printer drivers always save files according to the following schema:

- Inside the directory containing the configuration file for each user that prints a subdirectory is created in which the files are stored.
- The sub-directory is labeled with the Windows user name.
- The files are labeled with the Windows user name. The extension depends on the parameters.

The other parameters have the same functions in case of terminal server installations. Also a LOG file is created.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Appendix: Structure Tree Catalogs

File Conversion

The data of structure tree catalogs is administered with structure tree files. These files are created with the enaio® editor.
Since version 5.5 an API interface is available with the library `oxlist.dll` in order to convert ASCII text files into structure tree files and structure tree files into ASCII text files.
This allows data for structure tree catalogs to be created dynamically, to update and to align.
The library `oxlist.dll` is copied along with the also required library `oxmisc.dll` into the installation directories `clients\client32` and `clients\admin`.
The interface contains the two following functions which can e.g. be addressed using Visual Basic:

```
int WINAPI ConvertAsciiFileToTreeFile(LPCSTR lpAsciifile, LPCSTR lpDefinition, LPCSTR lpTreefile);

/*!
  The function generates a structure tree file from an ASCII file and an associated level
definition.

  @param LPCSTR lpAsciifile       - ASCII file that contains sorted short forms and corresponding
labels.  (short form and label must be separated by spaces, not by tabs)
  @param LPCSTR lpDefinition      – level definition, e.g. AA@A.9.AA
(A -> letters,numbers 9 -> numbers only, @ -> no separator, . -> separator)

  @param LPCSTR lpTreefile        - file name of the created structure tree file.

  @return int                             - 0 -> if successful, else -> error code
*/

int WINAPI ConvertTreeFileToAsciiFile(LPCSTR lpTreefile, LPCSTR lpAsciifile);

/*!
  The function generates an ASCII file from a structure tree file.

  @param LPCSTR lpTreefile        - file name of an existing structure tree file.
  @param LPCSTR lpAsciifile       - ASCII-file  to be created that contains sorted short forms and
corresponding labels.

  @return int                             - 0 -> if successful, else -> error code
*/
```

Example on how to use this functions in Visual Basic:

```
Private Declare Function ConvertTreeFileToAsciiFile Lib "oxlist.dll"
(ByVal lpTreefile As String, ByVal lpAsciifile As String) As Long
Private Declare Function ConvertAsciiFileToTreeFile Lib "oxlist.dll"
(ByVal lpAsciifile As String, ByVal lpDefinition As String,
ByVal lpTreefile As String) As Long


Private Sub Command1_Click()
lRet = ConvertTreeFileToAsciiFile
("C:\\Import\\Struktur.dat", "C:\\Import\\Struktur.txt")
End Sub

Private Sub Command2_Click()
lRet = ConvertAsciiFileToTreeFile
("C:\\Import\\Struktur.txt", "AA-99-AA", "C:\\Import\\Struktur1.dat")
End Sub
```

Structure of the text file

Specify a layer definition when creating a structure tree file in enaio® editor. When converting an ASCII text file into a structure tree file also specify this layer definition.
Within the ASCII file align short form and entry in each line.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

The short form must match the layer definition in terms of number of places and separator position.
Any separator character can be used in the syntax of the ASCII file, only the position is important.
In the example below instead of the separators '/' and '-' simply spaces were inserted.
Do not use a separator in between layers, like the '@' in the layer definition, also do not insert a separator in between layer short forms.
Short form and entry are separated by a space.
Add spaces for all short forms except the short forms of the last layer until the number of spaces matches the layer definition.

Example:

```
Level definition: 99/99-A

File content:
01      2001
01 01   January
01 01 F Family law
01 01 U Copyright law
01 01 V Contract law
01 02   February
01 02 F Family law
01 02 U Copyright law
01 02 V Contract law
01 03   March
01 03 F Family law
01 03 U Copyright law
01 03 V Contract law
02      2002
02 01   January
```

Illustration:
The number of positions according to the layer definition is seven, each short form has seven positions. Short forms of the first and second layer are complemented with spacing characters. The eighth position is a space and separates short form and entry. A tabulator is not allowed.
The third position is a separator, any separator can be inserted in the ASCII text file. The same applies to the sixth position.
Assignments must be in hierarchical order. An assignment for the second level must follow the corresponding assignment of the first level in the file.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN