enaio®

OPTIMAL SYSTEMS

Softwaredokumentation
enaio® client – Programmierreferenz

Version 10.0

09.05.2022
Version 10.0

Inhalt

Einführung

enaio® stellt eine COM-Schnittstelle zur Kommunikation mit dem Client zur Verfügung. Diese Schnittstelle besteht seit optimal_AS® 3.x. In den frühen Versionen von optimal_AS® 3.x bestand die COM-Schnittstelle parallel zur DDE-Schnittstelle, davor war die Kommunikation mit dem Client nur über DDE möglich.

COM steht für **C**omponent **O**bject **M**odel, es ist ein von Microsoft eingeführtes Modell zur Kommunikation zwischen Windowsanwendungen. Häufig wird dafür auch der Begriff **OLE** verwendet, das für **O**bject **L**inking and **E**mbedding steht. Hier wird ausschließlich der Begriff **COM** verwendet. In diesem Handbuch finden Sie Code-Beispiele, Hinweise für den Umstieg von der DDE-Schnittstelle von enaio® auf die COM-Schnittstelle und eine Beschreibung aller COM-Befehle, die zur Verfügung stehen.

Übergabedateien

Den COM-Befehlen für das Einfügen und Anfragen werden Dateien übergeben. Die Dateien, die Einfüge-Befehlen (**InsertIntoArchive**, **InsertIntoRegister**, **InsertIntoDocument**) übergeben werden, enthalten die Verschlagwortung der Objekte, die erzeugt werden sollen. Dateien für Anfragebefehle (**StartArchiveRequest**, **StartRegRequest**, **StartDocRequest**) enthalten Anfrageinformationen, also die gesuchte Verschlagwortung. Beide Dateitypen enthalten außerdem die Bezeichnung der angefragten bzw. einzufügenden Objekte.

Eine Übergabedatei für den Befehl **InsertIntoArchive** hat z.B. folgenden Aufbau:

```
[EINFÜGEN]
SCHRANK=Schrankname
FELD1=Feldwert#1
FELD2=Feldwert#2
...
FELDn=Feldwert#n
```

Eine Übergabedatei für den Befehl **StartArchiveRequest** hat z.B. folgenden Aufbau

```
[ANFRAGE]
SCHRANK = Schrankname
KLAUSEL1=Schrankname@Feldname1=Feldwert1
KLAUSEL2= Schrankname@Feldname2=Feldwert2
...
...
KLAUSELn= ...
DATENFELDER=0(1)
DATENHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
```

Hinweis: Die Übergabedateien unterscheiden sich für die einzelnen COM-Befehle. Bitte beachten Sie die Beschreibung der einzelnen COM-Befehle.

Interne Namen

Die in den Übergabedateien enthaltenen Objektnamen können durch interne Namen ersetzt werden. Dafür ist vor und nach dem internen Namen ein Prozentzeichen zu setzen.

Beispiel für eine Anfragedatei mit internen Namen:

```
[ANFRAGE]
SCHRANK = %interner_Schrankname%
KLAUSEL1=Schrankname@%interner_Feldname1%=Feldwert1
KLAUSEL2= %interner_Schrankname%@Feldname2=Feldwert2
```

Beispiel für einen Ausdruck mit internen Objektnamen:

```
AUSDRUCK1=%Eingangsbeleg%@Datum1^6^10.03.2000
```

Beispiel für eine Klausel:

```
KLAUSEL1=%Eingangsbeleg%@%Erstellt%=10.03.1997
```

Klauseln und Ausdrücke

Die Anfragedateien, die den Befehlen **StartArchiveRequest**, **StartRegRequest** und **StartDocRequest** übergeben werden, resultieren in einer Trefferliste mit enaio®-Objekten. Diese Anfragedateien können Anfrage-Klauseln oder -Ausdrücke enthalten, mit denen die Verschlagwortung der Objekte angefragt werden kann.

Klauseln

Klauseln haben die Syntax:

```
KLAUSEL1=Objekt@Feld=Wert
```

In einer Anfrage können mehrere Klauseln übergeben werden. Dazu sind die Klauseln fortlaufend nummeriert zu übergeben, also z.B.:

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

Die logische Verknüpfung von Klauseln ist das logische **UND**, d.h. dass die Trefferliste durch zusätzliche Klauseln eingeschränkt wird.

Beispiel für eine Klausel:

```
KLAUSEL1=Eingangsbeleg@Erstellt=10.03.1997
```

Dabei ist **Eingangsbeleg** der Objektname und **Vorlage** der Name eines Feldes des Objekts **Eingangsbeleg**. Zurückgeliefert werden alle Eingangsbelege, die im Feld **Erstellt** den Wert **10.03.1997** haben.

Hinweis: Es können auch interne Objekt- und Feldnamen verwendet werden.

Ausdrücke

Ausdrücke haben eine ähnliche Syntax. Folgende Formen sind möglich:

1. Form
```
AUSDRUCK1=Objekt@DBFeld^OP^Wert
```
2. Form
```
AUSDRUCK1=Objekt@Feldnr^OP^Wert
```
3. Form
```
AUSDRUCK1=Objekt@DBFeld^OP^Wert~BoolOP~Feldnr^OP^Wert
```

Objekt steht für den Objektnamen, **DBFeld** für den Spaltennamen des angefragten Felds in der Datenbank, **OP** steht für einen Vergleichsoperator, **Wert** für den Feldwert. **BoolOP** ist eine logische Verknüpfung der verschiedenen Vergleichsausdrücke. **^** trennt Vergleichswerte von Operatoren und **~** trennt Boolsche Ausdrücke von Boolschen Operatoren.

In einer Anfrage können mehrere Ausrücke übergeben werden. Dazu sind die Ausdrücke fortlaufend nummeriert zu übergeben, also z.B.:

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

Die logische Verknüpfung von Ausdrücken ist das logische **UND**, d.h. dass die Trefferliste durch zusätzliche Ausdrücke eingeschränkt wird.

Beispiel 1 für einen Ausdruck:

```
AUSDRUCK1=Eingangsbeleg@Datum1^6^10.03.2000
```

Wie in der Klausel bedeutet **Eingangsbeleg** den Objektname. **Datum1** ist der Datenbank-Spaltenname des Felds **Erstellt**. Der Spaltenname kann mit den Befehlen **GetObjectFields** und **GetObjectFieldsEx** ermittelt werden. Zurückgegeben werden alle Eingangsbelege, die im Datenbank-Feld **Datum1** einen Wert größer oder gleich **10.03.1997** haben.

Beispiel 2 für einen Ausdruck:

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

Zurückgegeben werden im Beispiel alle **Konto**-Objekte mit ObjectIDs kleiner als 4711.

Beispiel 3 für einen Ausdruck:

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

Zurückgegeben werden alle **Konto**-Objekte, deren ObjectIDs zwischen 4711 und 0815 liegen. Komplexe Ausdrücke mit Boolschen Operatoren sind mit (und) zu klammern, um die Eindeutigkeit des Ausdrucks sicherzustellen.

Beispiele zur Standortvorgabe

Beispiel: Dokument im Ordner

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

Mittels dieses Ausdrucks werden bei einer Dokumentenanfrage nur Dokumente direkt unterhalb der Ordnerebene zurückgegeben. Durch **1130^1^815** wird der Ordner mit der ID 815 als Parentordner festgelegt. **1133^1^0** gibt an, dass die gesuchten Register kein Parentregister besitzen.

Beispiel: Register im Ordner

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

Mittels dieses Ausdrucks werden bei einer Registeranfrage nur Register direkt unterhalb der Ordnerebene zurückgegeben. Durch **1121^1^815** wird der Ordner mit der ID 815 als Parentordner festgelegt. **1122^1^0** gibt an, dass die gesuchten Register kein Parentregister besitzen.

Beispiel: 'Register im Register'

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

Dieser Ausdruck legt bei einer Registeranfrage das Parentregister fest. Wichtig: Auch wenn das Parentregister von einem anderen Typ ist, wird in dem Ausdruck der Registertyp des angefragten Registers als Objekt angegeben.
Nachfolgend Zusammenstellungen der **Feldnamen/Feldnr**, Operatoren (**OP**) und Boolschen Operatoren (**BoolOP**)

Feldname/Feldnr.

Feldname	Feldnr	Beschreibung
Ordner		
STAMM_ID	1000	ID des Ordners
STAMM_TIME	1001	Zeitpunkt der Anlage
STAMM_LINKS	1002	Anzahl der Verknüpfungen mit diesem Ordner
Objekte		
OBJECT_ID	1100	ID des Dokuments
OBJECT_COUNT	1101	Anzahl der Dokumentdateien
OBJECT_FLAGS	1102	Archivierungsstatus: 0 = archiviert 1 = archivierbar 2 = nicht archivierbar 4 = Fehler in den Seiten 8 = Keine Seiten 16 = archiviert, Dia jedoch nicht 32 = nicht archiviert und für den Archivar nicht sichtbar
OBJECT_AVID	1103	Name des Archivars
OBJECT_AVDATE	1104	Datum der Archivierung
OBJECT_CRID	1105	Name des Anlegers
OBJECT_CRDATE	1106	Datum der Anlage
OBJECT_TIME	1107	Zeitstempel (Datum und Zeit) der Anlage
OBJECT_MAIN	1108	Haupttyp des Dokumentes
OBJECT_CO	1109	Nebentyp des Dokumentes
OBJECT_MEDDOCID	1110	Medien-Index des Dokuments
OBJECT_MEDDIAID	1111	Medien-Index des Dias
OBJECT_MEDDOCNA	1112	Pfad zum Dokument
OBJECT_MEDDIANA	1113	Pfad zum Dia
OBJECT_LINKS	1114	Anzahl der Verknüpfungen mit diesem Dokument
OBJECT_VERID	1115	ID der Originalvariante.Spezialwerte: 0=ohne Varianten, 1=das Dokument ist selbst das Original
OBJECT_LOCKUSER	1116	ID des Benutzers, welcher das Dokument gesperrt hat. Spezialwerte: 0=nicht gesperrt, 1=ausgelagert
Register		
REG_ID	1120	ID des Registers
REG_STAID	1121	ID des Ordners, in dem sich das Register befindet
REG_PARID	1122	ID des Registers, in dem sich das Register befindet
Ordner-Dokument-Relation		
SDSTA_ID	1130	ID des Ordners, in dem sich das Dokument befindet
SDOBJ_ID	1131	ID des Dokumentes
SDOBJTYPE	1132	Objekttyp (Haupttyp/Untertyp) des Dokumentes
SDREG_ID	1133	ID des Registers, in dem sich das Dokument befindet
SDDEL	1134	Löschflag
SDTIME	1135	Zeitpunkt des Einfügens
Mappe-Dokument-Relation		
MDDEL	1140	Löschflag
MDTIME	1141	Zeitpunkt der Anlage
MDMAP_ID	1142	ID der Mappe
MDSTA_ID	1143	Ordner-ID
MDOBJ_ID	1144	ID des Objekts
MDOBJTYPE	1145	Typ des Objekts
MDMOD	1146	Haupttyp des Objekts (wenn ohne Typ)
MDIN	1147	Zeitpunkt des Eingangs (unbenutzt)
MDOUT	1148	Ausgangszeitpunkt (unbenutzt)
MDCOUNT	1149	Anzahl der Seiten (wenn ohne Typ)
MDSEND	1150	Absender (unbenutzt)
Mappe		
MAPDEL	1160	Löschflag
MAPTIME	1161	Erstellungszeit
MAP_ID	1162	ID der Mappe
MAPCR_ID	1163	Name des Anlegers
MAPCRDATE	1164	Datum der Anlage
MAPRE_ID	1165	Name des Empfängers
MAPTHEME	1166	Thema der Mappe
MAPTYPE	1167	Typ der Mappe (unbenutzt)
Benutzer		
USER_ID	1170	ID des Benutzers
USER_SUPER	1171	0=kein Supervisor; 1=Supervisor (ab 3.00 SP02 ist das eine Kombination aus den Rechte-Flags -> siehe neues Rechtesystem)
USER_USER	1172	Name des Benutzers
USER_PASSWORD	1173	Kodiertes Passwort des Benutzers
USER_STATION	1174	Name der Arbeitsstation
USER_LOGIN	1175	Zeitstempel des Logins

OP-Liste

OP	SQL-Operator
1	= (bei exakter Suche, z.B. ^1^899999) LIKE (bei Suche nach Strings mit Platzhaltern, z.B. ^1^*.pdf)
2	!=
3	<
4	>
5	<=
6	>=
7	In
8	Ex

BoolOP-Liste

BoolOP	SQL-Operator
0	UND
1	ODER
2	NICHT

Hinweis: Es können auch interne Objektnamen verwendet werden.

Platzhalter in Anfragedateien

Folgende Platzhalter können in Anfragedateien verwendet werden:

Platzhalter	Bezug
#COMPUTER-GUID#	GUID des Computers des angemeldeten Benutzers
#COMPUTER-NAME#	Name des Computers des angemeldeten Benutzers
#COMPUTER-IP#	IP Adressen des Computers des angemeldeten Benutzers
#ANLEGER#	Verknüpfung mit Basisparameterfeld 'Anleger'
#ANLEGEDATUM#	Verknüpfung mit Basisparameterfeld 'Anlegedatum'
#ARCHIVAR#	Verknüpfung mit Basisparameterfeld 'Archivar'
#ARCHIVIERUNGSDATUM#	Verknüpfung mit Basisparameterfeld 'Archivierungsdatum'
#BENUTZER#	Name des angemeldeten Benutzers
#BESITZER#	Verknüpfung mit Basisparameterfeld 'Besitzer', welches die GUID des Besitzers enthält
#DATUM#	aktuelles Datum

Code-Beispiele

VB und VBA

Es gibt zwei gleichwertige Wege, um unter VB und VBA das enaio® COM-Objekt anzusprechen.

Ist enaio® client mit langwierigen Vorgängen beschäftigt, so reagiert er u.U. für den Konsumenten nicht erwartungsgemäß auf parallele Anfragen über die COM-Schnittstelle. Mit der Windows-Nachricht 1044 (WM_USER + 20) mit WPARAM 30 (API-Funktion SendMessage) kann im Vorfeld eine Prüfung durchgeführt werden. Ist der Client beschäftigt, wird eine 1 zurückgegeben, sonst eine 0. Es wird empfohlen diese Prüfung vor jedem Aufruf einer COM-Funktion durchzuführen und für den Fall 1 am Konsumenten zu reagieren.

Die folgenden Beispiele sind in VB Version 6 und Office-97-VBA erstellt.

1. **Dimensionierung einer Variablen als 'New optimal_AS.Application'.**
   Das enaio®-Objekt kann auf diesem Weg nur angesprochen werden, wenn im VB-Projekt der Verweis darauf aktiviert wurde. Ein wichtiger Vorteil dieser Verfahrensweise ist, dass das enaio® COM-Objekt der IntelliSense-Technik von VB und VBA mit allen Parametern bekannt ist und die IntelliSense-Listen angezeigt werden. Im folgenden Beispiel wird die Lizenzanmeldung für das COLD-Modul durchgeführt und anhand der Datei **InsInA.txt** ein Schrank angelegt, der gleich wieder gelöscht wird. Am Ende der **Main**-Routine findet sich eine Fehlerbehandlungsroutine, die den Fehlertext zu einer COM-Fehlernummer von enaio® ausgibt.

```
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. **Erzeugung des Objekts mit dem Befehl 'CreateObject'.**

   Diese Methode kann auch mit VB Script verwendet werden, die erste Methode dagegen nicht, da dabei Projektverweise erforderlich, aber nicht möglich sind. Nachteil dieser Methode ist, dass es keine IntelliSense-Unterstützung in VB und VBA gibt.

```
Dim MyAX As Object
....
Rest wie im vorigen Beispiel
....
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
    ....
```

Hinweis: Da es zahlreiche VB-Projekte mit COM-Anbindung gibt, heißt das enaio®-COM-Objekt aus Kompatibilitätsgründen weiterhin **optimal_AS.** Zur Erzeugung des optimal_AS-Objekts wird die **CreateObject**-Methode empfohlen. VB speichert intern die GUID des registrierten Objektes. Wenn Sie mit einem Projekt-Verweis arbeiten und eine neue Version von enaio® einspielen, kann es vorkommen, das mit der neuen Version eine neue Objekt-GUID erzeugt wird. Folge ist, dass das Objekt nicht erzeugt werden kann und Ihr Programm mit einer Fehlermeldung beendet wird.

Um in VB-Projekten im Genuss der **IntelliSense**-Technik zu bleiben, können Sie während der Entwicklung beide Methoden anwenden, d.h. Projektverweis mit **New**-Dimensionierung des Objekts sowie im Code die Erzeugung des Objekts mit **CreateObject**. Nach Abschluss der Entwicklung kommentieren Sie dann die **New**-Dimensionierung aus.

Objekt-Haupttypen

Für einige COM-Befehle muss ein Dokumenten-Haupttyp angegeben werden. Folgende Angaben sind möglich.

Haupttyp-Nr.	Haupttyp-Name
1	X-Dokument (Graustufenbild)
2	D-Dokument (S/W-Bild)
3	P-Dokument (Farbbild)
4	W-Dokument (Windows-Dokument)
5	M-Dokument (Video-Dokument)
6	E-Dokument (E-Mail)
7	XML-Dokument (XML)
8	Container Dokument

Alle COM-Befehle

ActivateApp

Definition	ActivateApp	(short nShow)	
Typ	Methode		
Beschreibung	Zeigt das Anwendungsfenster oder versteckt es		
Parameter	**nShow**	**1**	Anzeigen
		0	Verstecken
Rückgabewert			
Bemerkung			
Optionen			
Beispiel			

AdjustRetention

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

Typ Methode

Beschreibung Passt das Retention-Datum an das geplante Retention-Datum an.

Parameter

lObjectID	ID des Objekts
lObjectType	Typ des Objekts
varRetentionDate	das angepasste Retention-Datum im Format **DD.MM.YYYY** als Rückgabewert

Rückgabewert

0	Retention-Datum erfolgreich angepasst
-1	Retention-Datum konnte nicht angepasst werden
-7	Objekttyp unbekannt
-22	Objekt mit gegebener ID existiert nicht
-38	Ungültiger Dokumenttyp (es wurde ein Ordner oder Registertyp angegeben)
-40	Ungültiger Parameter wurde übergeben
-112	Das angegebene Objekt wurde (noch) nicht archiviert.

Bemerkung Die Funktion kann nur für bereits archivierte Dokumente verwendet werden. Die Fehlertexte können mit GetLastError() ermittelt werden.

Optionen

Beispiel

```
Dim a As Object
Dim retDate

Set a = CreateObject("optimal_as.application")

a.AdjustRetention lObjectID, lObjectType, retDate
```

AppBrowserSession

Definition

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

Typ

Beschreibung

Stellt eine Chromium-Browser Session zur Verfügung, die als Verbindung zwischen dem Client und einem Webserver dient. Mit dieser können im Client über die JavaScript-Schnittstelle des Browsers Aktionen angestoßen werden. Ist eine URL konfiguriert, wird diese bereits beim Start des Clients einmal geladen. Konfiguriert wird die Browser Session über die as.cfg.

Parameter

URL	Die aufzurufende URL. Ist diese nicht angegeben, wird keine Aktion angestoßen. Standard: nicht angegeben.
MODE	Situation, bei der die angegebene URL geladen wird. MODE=1 -> Beim Kontextwechsel MODE=0 -> Keine Aktion
TIMER	Gibt an, wann (in Sekunden) die angegebene URL geladen wird. Gültige Werte: 0 bis 3600. Standard: TIMER=0 -> kein Timer.

Rückgabewert

Bemerkung

Optionen

Beispiel

ApplicationLogin

Definition	ApplicationLogin (String strUser)
Typ	Methode
Beschreibung	Ermöglicht das Einloggen eines neuen Benutzers.
Parameter	**strUser** String mit Benutzername und Passwort, getrennt durch ein '@'
Rückgabewert	**1** kein Fehler **0** Fehler, Einloggen nicht möglich **-119** Einloggen zurzeit nicht möglich, da der Client gerade gestartet wird
Bemerkung	Wenn der String **DIALOG** übergeben wird, dann wird in jedem Fall der Login-Dialog geöffnet. Dies steht nur zur Verfügung, wenn die Passwortverifizierung nicht über **Novell NetWare**, sondern über den Benutzerdialog von enaio abgewickelt wird. Beim Fehler **-119** sollte in einer Warteschleife der Aufruf der Funktion nach kurzer Zeit erneut versucht werden, da der Client sich gerade in seiner initialen Startphase befindet und sich nicht ummelden kann.
Optionen	
Beispiel	

CalcFileDigest

Definition CalcFileDigest (String FileName, VARIANT* vRetDigest)

Typ Methode

Beschreibung Ermittelt den Hash-Wert (Digest) der angegebenen Datei

Parameter

FileName	Der Dateiname
vRetDigest	Der ermittelte Hash-Wert (Digest)

Rückgabewert

0	kein Fehler
-1	Hash-Wert konnte nicht ermittelt werden
-32	Die angegebene Datei existiert nicht

Bemerkung

Optionen

Beispiel

CheckInDocument

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

Typ Methode

Beschreibung Checkt ein für die externe Bearbeitung ausgechecktes Dokument wieder ein.

Parameter

lDocID	Dokumenten-ID
lDocType	Dokumententyp
strSourcePath	Pfad(ohne Dateiname) zur einzucheckenden Datei

Rückgabewert

0	kein Fehler
-7	Unbekannter Dokumenttyp
-53	Dokument wurde nicht ausgecheckt
-54	Dokument wurde nicht für externe Bearbeitung ausgecheckt
-55	Datei konnte nicht in Cache-Verzeichnis kopiert werden
-56	Name des Dokuments im Cache konnte nicht ermittelt werden
-57	Quelldatei existiert nicht

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

CheckLicence

Definition	CheckLicence	(String strModuleName)
Typ	Methode	
Beschreibung	Ermittelt, ob ein bestimmtes Modul für den aktuellen Arbeitsplatz lizensiert ist	
Parameter	**strModuleName**	Name des Moduls
Rückgabewert	**0** ist lizensiert **-1** ist nicht lizensiert	
Bemerkung		
Optionen		
Beispiel		

CheckObjectAccess

Definition

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

Typ

Methode

Beschreibung

Prüft Zugriffsrechte für das angegebene Objekt

Parameter

lObjectID Objekt-ID

lObjectType Objekttyp

lDesiredAccess

- **0** Indexdaten lesen
- **1** Indexdaten schreiben
- **2** Objekt löschen
- **3** Objekt ausgeben (z.B. Drucken)
- **4** Objekt schreiben
- **5** Auscheckstatus
- **6** Archivstatus
- **7** gibt an, ob das Objekt in der Workflowablage liegt (Rückgabewert =**1**) oder nicht (**0**)
- **8** gibt an, ob das Objekt zum Löschen vorgemerkt wurde.
  Rückgabe: **0**, Objekt nicht zum Löschen vorgemerkt
  1, Objekt zum Löschen vorgemerkt
  -22 Unbekanntes Objekt.

lFlags ab 4.20 SpII wird dieser Wert auf **1** gesetzt, so werden die richtigen Benutzerrechte angefragt, auch wenn das Sicherheitssystem für dieses Objekt nicht beachtet werden soll.

nShowError wenn ungleich **0**, wird ggf. eine Fehlerbox angezeigt

Rückgabewert

- **0** Benutzer hat kein Zugriffsrecht
- **1** Benutzer hat Zugriffsrecht
- **-1** Unbekanntes Recht
- **-7** Dokumenttyp unbekannt
- **-22** Unbekanntes Objekt
- **-46** Unbekannter Benutzer
- **-51** Dokument enthält keine Dateien
- **-52** Objekt bereits archiviert
- **-58** Dokument bereits ausgelagert
- **-59** Dokument von anderem Benutzer ausgecheckt
- **-60** Objekt ist kein W-Dokument

Bemerkung

Optionen

Beispiel

CheckOutDocument

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

Typ Methode

Beschreibung Checkt ein Dokument für die externe Bearbeitung aus.

Parameter

lDocID	Dokumenten-ID
lDocType	Dokumententyp
strPath	Pfad in den das Dokument ausgecheckt werden soll
varRetDocName	hier wird der vollständige Pfad und Dateiname des ausgecheckten Dokuments geliefert

Rückgabewert

0	kein Fehler
-1	Dokument konnte nicht ausgecheckt werden
-7	unbekannter Dokumenttyp
-51	Dokument enthält keine Dateien
-52	Objekt bereits archiviert
-58	Dokument bereits extern ausgecheckt
-59	Dokument von anderem Benutzer ausgecheckt

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

Hinweise: Die ausgecheckte Datei darf nicht umbenannt werden.

ClearSignatureProperties

Definition	ClearSignatureProperties()
Typ	Methode
Beschreibung	Löscht alle Eigenschaften, die zuvor mit **SetSignatureProperty** gesetzt wurden
Parameter	
Rückgabewert	
Bemerkung	Diese Funktion sollte unmittelbar nach einer digitalen Signatur aufgerufen werden.
Optionen	
Beispiel	

CloseAllWindows

Definition	CloseAllWindows ()
Typ	Methode
Beschreibung	Schließt alle Child-Fenster von enaio®client.
Parameter	
Rückgabewert	
Bemerkung	
Optionen	
Beispiel	

CloseObjectID

Definition	CloseObjectID (long lObjectID, long lObjectType)
Typ	Methode
Beschreibung	Schließt ein mit **OpenObjectID** geöffnetes Fenster.
Parameter	**lObjectID** Objekt-ID **lObjectType** Objekttyp
Rückgabewert	**0** kein Fehler **-11** Objekt-ID ungültig Der Fehlertext kann mit **GetLastError()** ermittelt werden.
Bemerkung	
Optionen	
Beispiel	

ConvertImage

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

Typ Methode

Beschreibung Die Funktion konvertiert eine Bilddatei ins angegebene Format.

Parameter

strSource	Quelldatei
strDestination	Zieldatei
nFormat	Zielformat, siehe GetImageType
nCompression	Kompressionsart oder Verlustfaktor für JPEG-Format, 1 empfohlen
nBitsPerPixel	Farbtiefe 1, 2, 4, 8, 16, 24 oder 0 für Farbtiefe der Quelldatei
nFlags	unbenutzt

Rückgabewert

0	Datei wurde erfolgreich konvertiert
<> 0	Fehler

Bemerkung

Optionen

Beispiel

CopyObject

Definition

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

Typ

Methode

Beschreibung

Erstellt eine Kopie eines Objektes incl. Mehrfachfelder und Dokument.

Parameter

lObjectID	Objekt-ID des zu kopierenden Objekts
lObjectType	Objekttyp des zu kopierenden Objekts
lFolderID	ID des Zielordners, zum Kopieren eines Ordners sollte dieser Wert 0 sein.
lRegisterID	ID des Zielregisters, 0, falls nicht in ein Register kopiert werden soll
nFlags	**0** Verschlagwortung und Dokument kopieren **1** nur Verschlagwortung kopieren
varRetObjectID	hier wird die ID des neuen Objekts zurückgegeben

Rückgabewert

0 kein Fehler
-11 Dokument-ID unbekannt
-22 Objekttyp unbekannt
-23 Zielregister-ID unbekannt
-47 Benutzer hat kein Schreibrecht, bzw. darf keine neuen Objekte anlegen.
-61 Objekt kann nicht eindeutig zugeordnet werden.
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Erstellt eine Kopie eines Objektes incl. Mehrfachfelder und Dokument. Unterliegt das Objekt der Versionsverwaltung, wird die aktuelle Version kopiert. Mit dieser Funktion können auch Ordner und Register kopiert werden. In diesem Falle werden die Inhalte von Ordner oder Register nicht kopiert.

Optionen

Beispiel

CreateDocumentLink

Definition

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

Typ

Methode

Beschreibung

Erzeugt einen neuen Verweis auf ein Dokument

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
lTargetID	Index des Zielordners/-registers.
lTargetType	Objekttyp des Zielordners/-registers.

Rückgabewert

-1	Verweis konnte nicht angelegt werden.
-3	Schrank unbekannt
-5	Register unbekannt
-7	Dokumenttyp unbekannt
-13	Registerkennung unbekannt (falls Zieltyp ein Register ist)
-14	Ordnerkennung unbekannt (falls Zieltyp ein Ordner ist)
-20	Angegebener Objekttyp ist kein Dokumenttyp
-22	unbekannter Objekttyp
-33	Dokumenttyp unzulässig (passt nicht zum Schrank)
-89	Objekt darf im Zielordner-/ Register nicht angelegt werden.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

CreateMimeFile

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

Typ Methode

Beschreibung Erzeugt eine Mime-Datei

Parameter

strFrom	Senderadresse
strTo	Empfänger
strCC	CC
strBCC	BCC
strSubject	Betreff
strBody	der Text
strAttachments	angehangene Dateien getrennt durch Pipe-Zeichen
strCreationTime	Erstellungszeit in der Form: Tag.Monat.Jahr Stunde:Minute
varRetFilename	vollständiger Pfad und Dateiname der erzeugten Mime-Datei

Rückgabewert

0	kein Fehler
-1	Mime-Datei konnte nicht erzeugt werden

Bemerkung Die erzeugte Datei ist bei Bedarf von der aufrufenden Anwendung zu löschen.

Optionen

Beispiel

DecodeIMAPFile

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

Typ Methode

Beschreibung Dekodiert eine IMAP-Datei und gibt deren Inhalt zurück.

Parameter

strFilename	Vollständiger Pfad und Dateiname der IMAP Datei
strMailDate	Das Sendedatum der Mail wird hier zurückgegeben.
strFrom	Der Name des Absenders wird hier zurückgegeben.
strTo	Die Empfängerliste (durch Semikolon getrennt) wird hier zurückgegeben.
strCC	Die CC-Liste (durch Semikolon getrennt) wird hier zurückgegeben.
strBCC	Die BCC-Liste (durch Semikolon getrennt) wird hier zurückgegeben.
strSubject	Der Betreff wird hier zurückgegeben.
strBody	Der Mailtext wird hier zurückgegeben.
strAttachments	Die Attachments werden hier durch Semikolon getrennt zurückgegeben

Rückgabewert

0	wenn kein Fehler
-1	wenn die Datei nicht dekodiert werden konnte

Bemerkung

CreateNewDocShare

Definition

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

Typ

Methode

Beschreibung

Erzeugt neue Freigaben des angegebenen Dokuments für die angegebenen Benutzer

Parameter

lIdent Dokument-ID
lType Dokumenttyp
Rights Rechte, die vorbelegt werden (RWXU)
Users Benutzer-ID's der Benutzer, welchen das angegebene Dokument freigegeben werden sollen, getrennt durch Semikolon
Info Infotext für die Freigabe

Rückgabewert

0 "Neue Freigabe(n) angelegt."
-125 "Ein modaler Dialog ist geöffnet."
-130 "Angegebener DMS-Objekttyp unbekannt."
-131 "Angegebener DMS-Objekttyp kein Dokument."
-132 "Nur Rechte 'RXWU' sind zulässig."
-133 "Angegebenes Dokument existiert nicht oder wurde gelöscht."
-134 "Dokumentfreigaben nicht möglich."
-135 "Die erforderliche Systemrolle um Dokumente freizugeben ist nicht vorhanden."
-136 "Mindestens ein angegebener Benutzer existiert nicht."
-137 "Abbruch durch Benutzer."
-138 "Server meldet Fehler: "

Der genaue Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Die Parameter 'Rights', 'Users' und 'Info' können leer sein.

Beispiel

```
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
```

DeleteFromArchive

Definition DeleteFromArchive (String strParam)

Typ Methode

Beschreibung Löscht den angegebenen Ordner oder das angegebene Dokument.

Parameter **strParam** String mit Objekt-ID und Objekttyp oder Dateiname einer Datei mit folgendem Aufbau:*

```
ObjektID1,Objekttyp1\r\n
ObjektID2,Objekttyp2\r\n
...
ObjektIDn,Objekttypn\r\n
```

* **\r\n** steht für einen Zeilenumbruch, **CR** und **LF**.

Rückgabewert

0	kein Fehler
-1	Übergabestring falsch
-11	Dokumentkennung nicht zulässig
-41	Übergabestring leer
-44	Angemeldeter Benutzer hat kein Löschrecht auf mindestens einen im Ordner befindlichen Dokumenttyp
-45	Angemeldeter Benutzer hat keinen Zugriff auf mindestens einen im Ordner befindlichen Dokumenttyp
-69	Die angegebene Übergabedatei ist leer.
-77	Der Server kann das angegebene Objekt nicht löschen. (weitere Informationen evtl. aus Serverprotokollierung ersichtlich)
-78	Das Dokument wird in oder mehreren Workflowprozessen verwendet und kann nicht gelöscht werden.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Diese Funktion löscht das angegebene Objekt **ohne Nachfrage,** wenn der im Archiv-System angemeldete Benutzer ausreichende Rechte zum Löschen des Objekts besitzt.
Auch bei der Übergabe eines Strings mit Objekt-ID und Typ ist es möglich mehrere Objekte mit einem Aufruf zu löschen. Der String muss dann folgenden Aufbau haben:

```
Objekt-ID1,Objekttyp1 ObjektID2,Objekttyp2 ObjektIDn
Objekttypn
```

Optionen

Beispiel

DoPrefetch

Definition	DoPrefetch (Variant lObjectID, Variant lObjectType)
Typ	Methode
Beschreibung	Führt einen Prefetchvorgang für das gegebene Dokument aus.
Parameter	**lObjectID** Objekt-ID als Variant **lObjectType** Objekttyp als Variant
Rückgabewert	**0** kein Fehler **<> 0** Fehler Der Fehlertext kann mit **GetLastError()** ermittelt werden.
Bemerkung	
Optionen	
Beispiel	

EMailImport

Definition EMailImport (Variant & config)

Typ Methode

Beschreibung Ablage von mehreren E-Mails an einen Standort mit Dubletten-Prüfung

Parameter config JSON zur Konfiguration (siehe Beispiel)

Rückgabewert Die Rückgabe kann eine Fehlernummer oder ein String mit JSON sein

Fehlernummer:
-2: Übergebenes JSON enthält formale Fehler
-3: Es wurden keine Dateien angegeben
-4: Angegebene Datei fehlt oder hat keinen Inhalt
-5: Angegebene Datei ist keine Mail (.eml oder .msg)
-6: Auf angegebene Datei kann nicht zugegriffen werden
-20: 'callMethod' fehlt
-21: 'objectTypeId' fehlt
-22: 'parentObjectId' fehlt
-23: 'parentObjectTypeId' fehlt
-10: Der angegebene Schrank ist nicht bekannt
-11: Der angegebene Dokumenttyp ist nicht bekannt
-12: Der angegebene Dokumenttyp ist dem EMS service nicht bekannt
-13: Angegebener Standort existiert nicht oder der angemeldete Benutzer kann nicht darauf zugreifen
-14: Der angegebene Ziel-Dokument-Typ befindet sich nicht im angegebenen Schrank
-15: Der angemeldete Benutzer darf am angegebenen Standort keine Dokumente anlegen
-16: Der angemeldete Benutzer darf am angegebenen Standort keine weiteren Dokumente anlegen

Kommt keine Fehlernummer, ist die Rückgabe ein String, der JSON enthält.

Folgende StatusCodes sind möglich:
0: Mail erfolgreich abgelegt
4: Speichern durch Benutzer abgebrochen
6: Speichern durch Benutzer übersprungen

JSON-Beispiel:

```
{
  "objects": [{
     "properties": {
          "system:objectId": {
             "value": "123456"
                              },
          "storeId": {
              "value": "<String kommt vom caller>"
                                    },
          "entryId": {
              "value": "<String kommt vom caller>"
```

```
                                        },
                "statusCode": {
                    "value": "-101"
                                        }
                                    }
                    },
                    ...

                    ]
}
```

Bemerkung

Die Dubletten-Prüfung findet über den EMS service statt.
Mit der folgenden Einstellung in der as.cfg werden alle Dokumente auf Dubletten geprüft. Das gilt auch für EMS-Konfigurationen, welche keine Dubletten-Prüfung besitzen. (Mode = 'NONE').
Für diese Dokumenttypen wird der EMS service angewiesen nach Dubletten zu suchen.
Ist das beim EMS service konfigurierte Handling 'LINK', dann erfolgt die Suche nach Mode= 'GLOBAL'
Ist das beim EMS service konfigurierte Handling 'COPY', dann erfolgt die Suche nach Mode = „TYPE“

[SYSTEM]
CHECKFORIDENTDOCS=1

Optionen

Beispiel

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

'sInsert = "{""objectTypeId"": ""393239"",""parentObjectId"":
""2737"",""parentObjectTypeId"": ""6488088"",""objects"": [{""properties"": {""filePath"": {
""value"": ""C:\\Downloads\\0000029C.eml""},""storeId"": {""value"": ""<String kommt vom
caller und wird diesem zurückgesendet>""},""entryId"": {
""value"": ""<String kommt vom caller und wird diesem zurückgesendet>""}}},{""properties"":
{""filePath"": {""value"": ""C:\\Downloads\\0000029B.eml""},""storeId"": {""value"":
""<String kommt vom caller und wird diesem zurückgesendet>""},""entryId"": {""value"":
""<String kommt vom caller und wird diesem zurückgesendet>""}}}]}"

sInsert = "{""callMethod"": ""insertEmails"",""objectTypeId"":
""393239"",""parentObjectId"": ""2737"",""parentObjectTypeId"": ""6488088"",""objects"":
[{""properties"": {""filePath"": {              ""value"":
""C:\\Downloads\\0000029C.eml""},""storeId"": {""value"": ""<String kommt vom caller und
wird diesem zurückgesendet>""},""entryId"": {
""value"": ""<String kommt vom caller und wird diesem zurückgesendet>""}}},{""properties"":
{""filePath"": {""value"": ""C:\\Documents\\0000029D.msg""},""storeId"": {""value"":
""<String kommt vom caller und wird diesem zurückgesendet>""},""entryId"": {""value"":
""<String kommt vom caller und wird diesem zurückgesendet>""}}}]}"

sResult = Application.EMailImport(sInsert)

If TypeName(sResult) = "String" Then
       MsgBox sResult
else
       MsgBox Application.GetLastError & " (" & sResult & ")"
end if
```

ExecuteRequest

Definition ExecuteRequest (String strRequestName)

Typ Methode

Beschreibung Führt eine gespeicherte Anfrage aus.

Parameter **strRequestName** Name der gespeicherten Anfrage

Rückgabewert

0 kein Fehler
-1 Es existieren keine gespeicherten Anfragen
-2 Es existiert keine Anfrage mit dem angegebenen Namen
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Anzeige erfolgt durch enaio®client.

Wenn die gespeicherte Anfrage Variablen besitzt, können diese in folgender Form angegeben werden:

```
VAR1=Inhalt1;VAR2=Inhalt2;...VARn=Inhaltn
```

In einem Aufruf können auch die Werte der statischen Variablen gesetzt werden:

```
STAT1=Inhalt1;STAT2=Inhalt2;...STATn=Inhaltn
```

Eine gespeicherte Anfrage mit Variablen öffnet die Anfrageseite, wenn nicht alle Variablen im Aufruf bestimmt wurden, die definierten Felder sind dann bereits ausgefüllt.

In jedem Fall kann ein gespeicherte Anfrage auch dazu ‚gezwungen' werden, die Anfragemaske anzuzeigen, dies geschieht mit dem Schalter ‚OPT1=1'

Optionen

OPT1=1
Erzwingt das Öffnen der Anfragemaske

Beispiele

Aufruf der gespeicherten Anfrage ‚Test', die statische Variablen hat:

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

Aufruf der gespeicherten Anfrage ‚Test', die statische Variablen hat und in jedem Fall die Anfragemaske öffnet:

```
ExecuteRequest „test OPT1=1;STAT1=Hallo;VAR1=Bert*"
```

Alle Felder mit entsprechenden Variablen werden dann mit den entsprechenden Inhalten belegt.

Aufruf der gespeicherten Anfrage ‚Test' mit Variablenübergabe

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

Hinweis: Es wird keine Überprüfung der Anfragedaten vorgenommen. SQL-Fehler sind möglich, wenn z.B. ein Datum erwartet und ein Text angegeben wird.

ExecuteRequestEx

Definition	ExecuteRequestEx	(String String	strRequestName, strParams)
Typ	Methode		
Beschreibung	Wie **ExecuteRequest**, nur dass die Parameter für die gespeicherte Anfrage extra angegeben werden müssen		
Parameter	**strRequestName** **StrParams**	Name der gespeicherten Anfrage Parameter für diese Anfrage (siehe **ExecuteRequest**)	
Rückgabewert	**0** kein Fehler **-1** Es existieren keine gespeicherten Anfragen **-2** Es existiert keine Anfrage mit diesem Namen Der Fehlertext kann mit **GetLastError()** ermittelt werden.		
Bemerkung	Diese Funktion kann im Gegensatz zu **ExecuteRequest** durch die Aufteilung der Parameter auch gespeicherte Anfragen bearbeiten, deren Name ein Leerzeichen enthält. Siehe auch **ExecuteRequest()**		
Optionen			
Beispiel			

FindObjectType

Definition	FindObjectType	(long lObjectID)
Typ	Methode	
Beschreibung	Ermittelt zur angegebenen Objekt-ID den zugehörigen Objekttyp	
Parameter	**lObjectID**	Objekt-ID zu dem der Objekttyp ermittelt werden soll
Rückgabewert	**String** **Leerer String**	mit Objekttyp wenn kein Objekttyp ermittelt werden konnte.
Bemerkung		
Optionen		
Beispiel		

FindObjectTypeEx

Definition

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

Typ

Methode

Beschreibung

Ermittelt zur angegebenen Objekt-ID den zugehörigen Objekttyp

Parameter

lObjectID	Objekt-ID zu dem der Objekttyp ermittelt werden soll
lType	Art des zu ermittelnden Objekttyps: **0** Dokument **1** Ordner **2** Register
vRetObjTypes	hier wird der ermittelte Objekttyp zurückgegeben

Rückgabewert

0	kein Fehler
-26	Objekt-ID ungültig
-40	angegebener Typ ungültig

Bemerkung

Diese Funktion beschleunigt das Finden des Objekttyps durch die Angabe der Art des Typs.

Optionen

Beispiel

FreeDocument

Definition	FreeDocument (long lObjectID)
Typ	Methode
Beschreibung	Gibt das angegebene W-Dokument frei.
Parameter	**lObjectID** Objekt-ID des freizugebenden Objekts
Rückgabewert	**>0** Anzahl der Dokumente die nicht eingecheckt werden konnten **0** kein Fehler **-1** Dokument konnte vom Archivserver nicht eingecheckt werden **-2** Dokument gesperrt
Bemerkung	Wird eine **0** für die Objekt-ID angegeben, werden alle gesperrten Dokumente freigegeben.
Optionen	
Beispiel	

FreeDocumentEx

Definition	FreeDocumentEx	(String strDocuments)
Typ	Methode	
Beschreibung	Gibt die angegebenen Dokumente mit einem zusätzlichen Thread frei.	
Parameter	**strDocuments**	String mit IDs der freizugebenden Objekte durch Semikolon getrennt.
Rückgabewert	**0**	
Bemerkung	Diese Funktion öffnet einen Thread zum Freigeben der Objekte, d.h. es wird nicht gewartet, bis alle Objekte freigegeben wurden. Wird eine **0** für die ObjectID angegeben, werden alle gesperrten Dokumente freigegeben.	
Optionen		
Beispiel		

GenerateColdFiles

Definition

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

Typ

Methode

Beschreibung

Erzeugt aus ASCII-COLD-Import-Dateien entsprechende Bildobjekte.

Parameter

strFile1	entweder Datendatei oder Steuerdatei
strFile2	entweder Datendatei oder Steuerdatei
strToken	Trennzeichen für Dateiliste
vRetFileList	Liste mit den Dateinamen der erzeugten Bilder getrennt durch die mit strToken angegebenen Zeichen.

Rückgabewert

0	kein Fehler
-83	Unterverzeichnis für die Cold-Dateien konnte nicht erstellt werden.
-84	Bild konnte nicht erzeugt werden
-85	Datei konnte nicht ins Zielverzeichnis verschoben werden.

Bemerkung

Optionen

Beispiel

GenerateOSFile

Definition	GenerateOSFile	(long long	lObjectID, lObjectType)
Typ	Methode		
Beschreibung	Erzeugt eine OS-Datei aus den angegebenen Parametern.		
Parameter	**lObjectID** Objekt-ID **lObjectType** Objekttyp		
Rückgabewert	**leerer String** wenn Fehler **Dateiname** der erzeugten OS-Datei. Der Fehlertext kann mit **GetLastError()** ermittelt werden.		
Bemerkung	Die erzeugte OS-Datei ist bei Bedarf vom aufrufenden Programm zu löschen.		
Optionen			
Beispiel			

GetActiveDocument

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

Typ Methode

Beschreibung Liefert das COM-Objekt zu einem optimal_AS: Active Document.

Parameter

strDocName	Name des Active Documents, wie in der ***as.cfg*** angegeben.	
bOpen	**TRUE**	wenn das Dokument geöffnet werden soll, falls es nicht geöffnet ist.
vRetVal	hier wird eine Fehlernummer zurückgegeben	

Rückgabewert

0	kein Fehler
-70	kein Active Document mit diesem Namen gefunden
-72	kein optimal_AS: Active Document

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

GetAllArchives

Definition GetAllArchives()

Typ Methode

Beschreibung Ermittelt alle Schränke

Parameter

Rückgabewert String mit allen verfügbaren Schränken

Bemerkung Der Rückgabestring hat folgendes Format:

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

Optionen

Beispiel Codebeispiel:

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

Helpstr kann z.B. folgenden String enthalten:

```
Kunde,1;Patient,2
```

GetAllOpenFolders

Definition GetAllOpenFolders()

Typ Methode

Beschreibung liefert eine Liste mit IDs und Typen aller geöffneten Ordner. Falls in Ordnern Register geöffnet sind, werden ebenfalls dessen IDs und Typen angegeben.

Parameter

Rückgabewert String mit Liste der IDs und Typen folgender Form:

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

Bemerkung

Optionen

Beispiel

GetCurrentResultList

Definition GetCurrentResultList (VARIANT* pstrItems)

Typ Methode

Beschreibung Gibt die Items der aktuellen Trefferliste zurück.

Parameter

pstrItems hier werden die IDs und die Objekttypen der in der aktuellen Trefferliste angezeigten Objekte zurückgegeben

```
ObjektID1,ObjekttypI1; ObjektID2,ObjekttypI2; …;
ObjektIDn,ObjekttypIn
```

Rückgabewert Der Rückgabewert entspricht der Anzahl der übergegeben Objekte, wenn er größer **0** ist, bzw. einem Fehler wenn er kleiner **0** ist.
-40 fehlerhafter Eingabeparameter
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

GetCurrentSelection

Definition GetCurrentSelection (VARIANT* varResult)

Typ Methode

Beschreibung Liefert die in der aktuellen Trefferliste selektierten IDs und Objekttypen

Parameter **varResult** hier werden die Indizes und Objekttypen in folgender Form zurückgegeben:

```
ObjektID1,Objekttyp1;ObjektID2,Objekttyp2;......
```

Rückgabewert Der Rückgabewert entspricht der Anzahl der selektierten Objekte.

Bemerkung

Optionen

Beispiel

GetDataID

Definition

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

Typ

Methode

Beschreibung

Gibt Objektdaten zurück, d.h. Feldnamen und -werte

Parameter

lObjectID Objekt-ID des zu öffnenden Objekts
lObjectType Objekttyp
nMode Steuert den Rückgabewert.

0	Gibt Feldnamen und –inhalte zurück
1	Gibt Basisparameter zurück, wenn es sich nicht um Ordner oder Register handelt
2	Gibt Namen und Werte von Mehrfachfeldern zurück
10	Entspricht dem Wert **0**, gibt aber die internen Feldnamen zurück
12	Entspricht dem Wert **2**, gibt aber die internen Feldnamen zurück
bWriteToFile	Ist der Wert **0**, dann werden die Objektdaten in einem String zurückgegeben. Bei Wert **1** wird in eine Datei geschrieben und der Dateiname zurückgegeben.

Rückgabewert

Ergebnisstring oder **Dateiname**.
Leerer String wenn Fehler.
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

Für nMode = 0:

```
Feldbezeichner = Feldinhalt\r\n
Feldbezeichner = Feldinhalt\r\n
```

Für nMode = 1:

```
ZEITSTEMPEL = Feldinhalt\r\n
ANLEGER = Feldinhalt\r\n
ARCHIVAR = Feldinhalt\r\n
ANGELEGT = Feldinhalt\r\n
ARCHIVIERT = Feldinhalt\r\n
_STANDORTE = Anzahl in SDREL_
```

Für nMode = 2 werden die Mehrfachfelder eines Objekts:*

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

* **\r\n** steht für einen Zeilenumbruch, **CR** und **LF**.

GetDescription

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

Typ Methode

Beschreibung Liefert den Beschreibungstext aus einem Listenfeld oder Strukturbaum zu einem gegebenen Kürzel.

Parameter

strShortName	Kürzel, zu dem der Beschreibungstext geliefert werden soll.
lObjectType	Objekttyp
strFieldName	Name des Listen- oder Strukturbaumfeldes

Rückgabewert

String	mit der Beschreibung,
leerer String	wenn keine Beschreibung vorhanden oder Fehler.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Funktion gilt ausschließlich für Listen- und Strukturbaumfelder.

Optionen

Beispiel Das Feld ‚Monat' hat eine Liste mit folgenden Einträgen:

```
01|Januar
02|Februar
03|März
```

Codebeispiel

```
HelpStr=MyAX.GetDescription(„02", 65535, „Monat")
```

HelpStr hat jetzt den Wert „Februar"

GetDocTypesFromArchive

Definition GetDocTypesFromArchive (String strSchrankName)

Typ Methode

Beschreibung Ermittelt alle zu einem Schrank gehörenden Dokumenttypen.

Parameter **strSchrankName** String mit dem Schrankbezeichner

Rückgabewert String mit den zum Schrank gehörenden Dokumenttypen.

Bemerkung Der Rückgabestring hat folgendes Format:

```
DOKUMENT1,OBJEKTTYP1;DOKUMENT2,OBJEKTTYP2 ...
DOKUMENTn,OBJEKTTYPn
```

Optionen

Beispiel Codebeispiel:

```
Helpstr=MyAX.GetDocTypesFromArchive
```

Helpstr enthält z.B. folgende String:

```
Eingangsbeleg,131085;Ausgangsbeleg,262159;Attribut,131086
```

GetEnvironment

Definition GetEnvironment (short nEnvType)

Typ Methode

Beschreibung Ermittelt Einstellungen aus dem Archivsystem.

Parameter **nEnvType** Nummer der zu ermittelnden Einstellung

Rückgabewert String mit gewünschter Einstellung

Bemerkung Mögliche Werte für nEnvType sind:

0	ermittelt osGetTmpDir
1	ermittelt osGetAppCWD
2	ermittelt osGetCfgFileName
3	ermittelt osGetUserName
4	ermittelt osGetHomeCWD
5	ermittelt osGetIniFileName
6	ermittelt den Anwendungsnamen, im Falle einer OEM-Version, den OEM-Namen des Clients
7	ermittelt die ID des aktuellen Benutzers
8	ermittelt osGetStationNumber
9	ermittelt osGetLocalWorkDir
10	ermittelt FileVersion des Clients
11	ermittelt alle Gruppennamen des aktuellen Benutzers. Ist der Benutzer in mehr als einer Gruppe, werden die Gruppennamen durch Semikolon getrennt.
12	unbenutzt
13	ermittelt eine Liste mit den in der aktuellen Sitzung neu angelegten Dokument-Indizes. Wurden bereits mehrere Dok. neu angelegt, so werden sie durch Komma getrennt hintereinander geschrieben. Bsp.: „1234,1235,1236“. Wurden noch keine neuen Dokumente angelegt, enthält der Rückgabewert „EMPTY“
14	ermittelt den vollständigen Benutzernamen.
15	ermittelt die maximal zulässige Größe von Textnotizen.
16	ermittelt die E-Mail-Adresse des angemeldeten Benutzers, sofern diese dem System bekannt ist.
17	ermittelt das Bemerkungsfeld zum angemeldeten Benutzer.
18	ermittelt die GUID des angemeldeten Benutzers.
19	ermittelt die GUID der Abteilung des angemeldeten Benutzers.
20	ermittelt die Anzahl der vergeblichen Logins eines Benutzers, seit dem letzten erfolgreichen Login.
21	ermittelt die Zeit ab wann der Account des aktuellen Benutzers gültig ist.
22	ermittelt die Zeit bis wann der Account des aktuellen Benutzers gültig ist.
23	ermittelt osGetDocLockStation.
24	ermittelt die im Client eingestellte GUI Sprache (hier wird die Endung 'eng' für Englisch, 'fra' für Französisch und 'de' für Deutsch (Default).
25	ermittelt die ID der im Client eingestellten Sprache der Objektdefinition: z.B. **7**-Deutsch, **9**-Englisch.

Optionen

Beispiel

GetFilesFromID

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

Typ

Methode

Beschreibung

Ermittelt die zu dem Objekt gehörenden Imagefiles.
Sind die Images archiviert, so werden sie zunächst in den Cache geladen.

Parameter

lObjectID	ID des Objekts
lObjectType	Objekttyp
strToken	Trennzeichen

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Rückgabewert

String	mit Dateinamen inkl. Pfad
Leerer String	wenn Fehler

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Gehören zu einem Dokument mehrere Seiten, stehen die Dateinamen hintereinander durch ein Leerzeichen getrennt im String. Ist ein Trennzeichen angegeben, sind die Dateinamen durch dieses Zeichen getrennt. Das macht Sinn, weil innerhalb der Dateinamen Leerzeichen auftauchen können.

Optionen

Beispiel

GetFilesFromIDEx

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

Typ Methode

Beschreibung Ermittelt die zu einem Objekt gehörigen Dateien.

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
strToken	Trennzeichen, das zur Trennung der Namen im Rückgabestring verwendet werden soll
bWriteProtected	**TRUE** Dokument schreibgeschützt anfordern
vFileNames	hier werden die Dateinamen zurückgegeben

Rückgabewert

0	kein Fehler
-7	Dokumenttyp unbekannt
-15	Dokument-ID unbekannt
-51	Dokument hat keine Dateien

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Im Unterschied zu **GetFilesFromID()** kann angegeben werden, ob die Dateien ausgecheckt, oder ob sie schreibgeschützt angefordert werden sollen.

Optionen

Beispiel

GetImageType

Definition GetImageType (String strSource)

Typ Methode

Beschreibung Bestimmt den Bildtyp der angegebenen Datei.

Parameter

strSource	vollständiger Pfad und Dateiname der Bilddatei

Rückgabewert

> 0	Dateityp siehe Bemerkung
< 0	Fehler
-20009	Dateiformat fehlerhaft

Bemerkung Folgende Dateitypen sind definiert:

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) (von OPTIMAL SYSTEMS verschlüsselte JPG-Datei), TIFOS(1001) (von OPTIMAL SYSTEMS verschlüsselte TIFF-Datei)

Optionen

Beispiel

GetLastError

Definition	GetLastError()
Typ	Methode
Beschreibung	Liefert den Fehlertext des zuletzt aufgetretenen Fehlers.
Parameter	keine
Rückgabewert	**Fehlertext** als String.
Bemerkung	Diese Funktion ist unmittelbar nach dem Auftreten eines Fehlers aufzurufen, um nicht den Text eines anderen Fehlers zu erhalten.
Optionen	Keine
Beispiel	Bei auftretenden Fehlern soll eine Fehlerbehandlungs-Routine gestartet und dort der Fehlertext ermittelt werden.

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

GetMainType

Definition	GetMainType	(long long	lObjectID, lObjectType)
Typ	Methode		
Beschreibung	Ermittelt den Haupttyp eines Objekts. Im Falle eines multimedialen Dokumenttyps wird der tatsächliche Haupttyp des Objekts geliefert.		
Parameter	**lObjectID** Objekt-ID **lObjectType** Objekttyp		
Rückgabewert	**> = 0** Haupttyp des Objekts **-22** Objekttyp unbekannt **-26** Objekt-ID unbekannt		
Bemerkung			
Optionen			
Beispiel			

GetObjectFields

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

Typ Methode

Beschreibung Ermittelt alle Felder eines Objektes.
Typen und Längen der Felder werden ebenfalls ermittelt.
Objekttyp kann hier ein Ordner, Register oder Dokument sein.

Parameter

strObjectName	wird nicht verwendet.
lObjectType	Objekttyp
nMode	Modus (siehe Bemerkung)

Rückgabewert

String	mit Feldbezeichnern
Leerer String	wenn Fehler.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Der Rückgabestring hat folgendes Format:
Für nMode = 0:

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

Für nMode = 1

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

Mögliche Feldtypen:

Typ	Beschreibung	Datenbank
A	alle Zeichen ohne Ziffern	CHAR
X	alle Zeichen	CHAR
Z	nur Ziffern	CHAR
S	männlich/weiblich/divers	CHAR
Q	ja/nein	CHAR
G	Großbuchstaben	CHAR
P	Patientenart	CHAR
T	links/rechts	CHAR
L	alle Zeichen	CHAR
M	alle Zeichen	CHAR
D	Datum	DATE
9	Zahlen	INTEGER
I	Volltextindex	INTEGER
#	Realzahlen	DECIMAL
1	Optionsschaltfläche	SHORT
0	Kontrollkästchen	SHORT

Feldlängen beziehen sich auf die Anzahl der Zeichen die eingegeben werden können. Bei Realzahlen setzt sich die Kapazität des Dezimalfeldes aus zwei Nachkommastellen sowie (Feldlänge - 2) Vorkommastellen zusammen.

Optionen

Beispiel

Codebeispiel 1:

```
Helpstr=MyAX.GetObjectFields("D-Berichte",131073,0)
```

Helpstr enthält z.B. folgenden String

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

Codebeispiel 2:

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

Helpstr enthält z.B. folgenden String

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

GetObjectFieldsEx

Definition

GetObjectFieldsEx	(String	strObjectName,
	long	lObjectType,
	short	nMode)

Typ

Methode

Beschreibung

Ermittelt alle Felder eines Objektes. Diese Funktion entspricht der Funktion **GetObjectFields()**. Der einzige Unterschied ist, dass zusätzlich noch der Katalogtyp ermittelt wird.

Typen und Längen der Felder werden ebenfalls ermittelt.

Objekttyp kann hier ein Ordner, Register oder Dokument sein.

Parameter

strObjectName	wird nicht verwendet
lObjectType	Objekttyp
nMode	Modus (siehe Bemerkung)

Rückgabewert

String	mit Feldbezeichnern
Leerer String	wenn Fehler.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Der Rückgabestring hat folgendes Format:

Für nMode = 0:

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

Für nMode = 1

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

Für nMode = 2

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

Für 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...
```

Für 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...
```

Mögliche Feldtypen:

Typ	Beschreibung	Datenbank
A	alle Zeichen ohne Ziffern	CHAR
X	alle Zeichen	CHAR
Z	nur Ziffern	CHAR
S	männlich/weiblich/divers	CHAR
Q	ja/nein	CHAR
G	Großbuchstaben	CHAR
P	Patientenart	CHAR
T	links/rechts	CHAR
L	alle Zeichen	CHAR
M	alle Zeichen	CHAR
D	Datum	DATE
9	Zahlen	INTEGER
I	Volltextindex	INTEGER
#	Realzahlen	DECIMAL
1	Optionsschaltfläche	SHORT
0	Kontrollkästchen	SHORT
K	Schaltfläche	AB 3.50.619 !!!

Feldlängen beziehen sich auf die Anzahl der Zeichen die eingegeben werden können. Bei Realzahlen setzt sich die Kapazität des Dezimalfeldes aus zwei Nachkommastellen sowie (Feldlänge - 2) Vorkommastellen zusammen.

Mögliche Werte für den Katalogtyp:

0 ohne Katalog
1 Liste
2 Baum
3 Hierarchie
4 Datenbank
5 Strukturbaum
6 Addon
7 Volltext

Optionen

Beispiel

Codebeispiel 1:

```
Helpstr=MyAX.GetObjectFields("D-Berichte",131073,0)
```

Helpstr enthält z.B. folgenden String

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

GetObjectName

Definition	GetObjectName	(long VARIANT*	lObjectID, strReturnObjectName)
Typ	Methode		
Beschreibung	Liefert den Objektnamen zur gegebenen Objekt-ID		
Parameter	**lObjectID** **strReturnObjectName**		Objekt-ID hier wird der Name des Objekts zurückgegeben
Rückgabewert	**0** kein Fehler **-1** Fehler Der Fehlertext kann mit **GetLastError()** ermittelt werden.		
Bemerkung			
Optionen			
Beispiel			

GetObjectNameEx

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

Typ Methode

Beschreibung Liefert den Objektnamen zur gegebenen Objekt-ID

Parameter

lObjectID	Objekt-ID
lType	Typ des zu ermittelnden Objekts **-1** Typ nicht bekannt **0** die Objekt-ID stammt von einem Dokument **1** die Objekt-ID stammt von einem Ordner **2** die Objekt-ID stammt von einem Register
nMode	Art des zu liefernden Ergebnisses **0** Objekttypname **1** Interner Objekttypname **2** Tabellenname des Objekttyps **3** UID des Objekttyps
strReturnObjectName	hier wird der Name des Objekts zurückgegeben

Rückgabewert

0	kein Fehler
-1	Fehler

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

GetObjectPath

Definition GetObjectPath (long lObjectID,
long lObjectType)

Typ Methode

Beschreibung Ermittelt den Standortpfad zu einem Objekt.
Der Ordner sowie alle Register werden ermittelt.
Objekttyp kann hier ein Ordner, Register oder Dokument sein.

Parameter
lObjectID Objekt-ID
lObjectType Objekttyp

Rückgabewert
String mit dem Pfad zum Objekt.
Leerer String wenn Fehler.
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Der Rückgabestring hat folgendes Format:

```
SCHRANKINDEX,SCHRANKTYP;REGISTERINDEX1,REGISTERTYP;REGISTERI
NDEX2,REGISTERTYP;...;REGISTERINDEXn,REGISTERTYP
```

Registerangaben werden nur dann zurückgegeben, wenn sich das Objekt in einem Register befindet.
Befindet sich das gesuchte Objekt in einem Register, welches wiederum in einem Register liegt, so setzt sich der Pfad wie folgt zusammen:
Ordner-ID, Ordnertyp, Register-ID und Typ in dem das Objekt liegt, Register-ID, Registertyp in dem sich das Register befindet usw.

Beispiel:
Ein Objekt befindet sich in einem Register ‚REGISTER3', dieses liegt in einem Register REGISTER2 und dieses wiederum liegt ebenfalls in einem Register REGISTER1. Der Objektpfad sieht dann folgendermaßen aus:

```
Ordner-ID,OrdnerTyp;REGISTER3-ID,REGISTER3-Typ;REGISTER2-
ID,REGISTER2-Typ;REGISTER1-ID,REGISTER1-Typ
```

Optionen

Beispiel

GetObjectPathEx

Definition

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

Typ

Methode

Beschreibung

Ermittelt wie GetObjectPath den Standortpfad zu einem Objekt.
Die Funktion ermittelt auch die Standorte aller Verweisdokumente, sofern welche vorhanden sind.
Der Ordner sowie alle Register werden ermittelt.
Objekttyp kann hier ein Ordner, Register oder Dokument sein.

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
vRetObjectPath	hier wird der Pfad zurückgegeben.

Rückgabewert

-3	Schrank unbekannt
-5	Register unbekannt
-22	Dokumenttyp unbekannt

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Der Rückgabestring hat folgendes Format:

```
SCHRANKINDEX,SCHRANKTYP;REGISTERINDEX1,REGISTERTYP;REGISTERINDEX2,REGISTERTYP;...;REGISTERINDEXn,REGISTERTYP
```

Registerangaben werden dann zurückgegeben, wenn sich das Objekt in einem Register befindet. Die einzelnen Pfade bei Verweisdokumenten werden durch | (Pipe) getrennt.

Befindet sich das gesuchte Objekt in einem Register, welches wiederum in einem Register liegt, so setzt sich der Pfad wie folgt zusammen:
Ordner-ID, Ordnertyp, Register-ID und Typ in dem das Objekt liegt, Register-ID, Registertyp in dem sich das Register befindet usw.

Beispiel:
Ein Objekt befindet sich in einem Register ‚REGISTER3', dieses liegt in einem Register REGISTER2 und dieses wiederum liegt ebenfalls in einem Register REGISTER1. Der Objektpfad sieht dann folgendermaßen aus:

```
Ordner-ID,OrdnerTyp;REGISTER3-ID,REGISTER3-Typ;REGISTER2-ID,REGISTER2-Typ;REGISTER1-ID,REGISTER1-Typ
```

Optionen

Beispiel

GetObjectPattern

Definition	GetObjectPattern	(long	lIdent,
		long	lType,
		long	lMode,
		string	Titel)

Typ Methode

Beschreibung Ermittelt den Titel des angegebenen Objekts, wie er als Fenstertitel verwendet wird oder konfiguriert ist.

Parameter

lIdent	ID des Objekts
lType	Typ des Objekts
lMode	zur Zeit nur '1'
Titel	Titel des angegebenen Objekts wird zurückgegeben

Rückgabewert

0	Kein Fehler
-22	Das Objekt ist unbekannt
-117	Unbekannter Mode

Bemerkung Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Optionen

Beispiel

```
Dim ax
Dim sRet

set ax = CreateObject("optimal_AS.Application")

ax.GetObjectPattern 846, 41, 1, sRet

set ax = Nothing
MsgBox sRet
```

GetObjectPatternPath

Definition

```
GetObjectPatternPath (long   lIdent,
                      long   lType,
                      string Titel)
```

Typ

Methode

Beschreibung

Gibt sämtliche Titel des angegebenen Objekts in einem String zurück.
Zum ermitteln des Pfads zum Objekt wird die Methode 'GetObjectPath' benutzt.

Parameter

lIdent ID des Objekts
lType Typ des Objekts
Titel Sämtliche Überschriften der einzelnen Objekte des Pfads

Rückgabewert

0 Kein Fehler
-1 Pfad konnte nicht ermittelt werden

Bemerkung

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Optionen

Beispiel

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

' Test-System: 10.1.4.106#4000
' Benutzer: peter
' Passwd: peterson


iRet = ax.GetObjectPatternPath(1103, 6488089, sRet)

if iRet <> 0 Then
    sRet = ax.GetLastError()
End If

MsgBox sRet
```

GetObjectTypeImage

Definition

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

Typ

Methode

Beschreibung

Gibt das Icon eines Objekttyps in Form einer Bilddatei zurück. Die Bilddatei liegt derzeit nur im GIF-Format vor.

Parameter

lObjectID	ID des Objekts (kann 0 sein, falls es kein Objekt mit Iconkatalog ist)
lObjectType	Typ des Objekts
strImageFormat	zur Zeit unbenutzt
vRetImagePath	der vollständig Pfad zur Bilddatei wird hier zurückgegeben

Rückgabewert

0	Bilddatei erfolgreich ermittelt
-1	Bilddatei konnte nicht ermittelt werden

Bemerkung

Die Fehlertexte können nicht mit GetLastError() ermittelt werden.

Optionen

Beispiel

```
Dim a As Object
Dim vRetImagePath as Variant

Set a = CreateObject("optimal_as.application")

a.GetObjectTypeImage lObjectID, lObjectType, "",
vRetImagePath
```

GetRegTypeFromArchive

Definition GetRegTypeFromArchive (String strSchrankName)

Typ Methode

Beschreibung Ermittelt alle zu einem Schrank gehörenden Register mit Namen und Objekttyp

Parameter **strSchrankName** String mit dem Schrankbezeichner

Rückgabewert Semikolon-getrennter String mit Registername,Registertyp.

Bemerkung Der Rückgabestring hat folgendes Format:

```
REGISTERNAME1,OBJEKTTYP1;REGISTERNAME2,OBJEKTTYP2
```

Optionen

Beispiel Code-Beispiel:

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

sRet kann z.B. folgenden String enthalten:

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

GetResultFields

Definition

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

Typ

Methode

Beschreibung

Gibt die vom aktuellen Benutzer für Trefferlisten des angegebenen Typs konfigurierten Felder zurück.

Parameter

strObjectType	interner Name des Objekttyps (wird nur ausgewertet wenn ID=-1 übergeben wird)
lObjectType	Objekttyp
lMode	Art des zu liefernden Ergebnisses
strDelimiter	Trennzeichen, das zur Trennung der Feldnamen im Rückgabestring verwendet wird. Wird kein Trennzeichen übergeben, wird per Default ein „;“ verwendet.
pstrFields	hier werden die benutzerkonfigurierten Felder in dieser Form zurückgegeben (siehe Bemerkungen):

```
Feld1; Feld2; …; Feldn
```

Rückgabewert

0 kein Fehler
-7 Objekttyp unbekannt
-40 fehlerhafter Eingabeparameter
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

lMode = 0 gibt die Displaynamen der Felder zurück
lMode = 1gibt die internen Namen der Felder zurück
lMode = 2 gibt die OSGUIDs der Felder zurück

Optionen

Beispiel

GetSignatureProperty

Definition	String GetSignatureProperty (String strPropertyName)	
Typ	Methode	
Beschreibung	Gibt eine Eigenschaft zurück, die zuvor mit SetSignaturProperty gesetzt wurde	
Parameter	**strPropertyName**	Name der Eigenschaft
Rückgabewert	Der Wert der Eigenschaft.	
Bemerkung		
Optionen		
Beispiel		

GetSelectedObject

Definition GetSelectedObject (Variant strSelectedObjects)

Typ Methode

Beschreibung Gibt die aus der Trefferliste durch **SelectObject** gewählten Objekte zurück.

Parameter **strSelectedObjects** hier werden die Objekte in dieser Form zurückgegeben:

```
ObjektID1,Objekttyp1;...,...;ObjektIDn,Objekttypn
```

Rückgabewert

3	Es wurden Objekte ausgewählt, Auswahl ist beendet
2	Auswahl abgebrochen
1	noch kein Objekt ausgewählt
0	SelectObject wurde nicht aufgerufen
-3	Schrank unbekannt
-5	Register unbekannt
-26	Objekt-ID unbekannt

Bemerkung Wird diese Funktion innerhalb einer Schleife verwendet, ist dafür zu sorgen, dass Fensternachrichten verarbeitet werden können, sonst kann kein Objekt aus der Trefferliste gewählt werden.

Optionen

Beispiel

GetSignDocumentResult

Definition GetSignDocumentResult (Variant vRetSignText)

Typ Methode

Beschreibung Gibt das Ergebnis der letzten Digitalen Signatur eines Dokuments zurück.

Parameter **vRetSignText** der gewählte Signaturtext wird hier zurückgegeben

Rückgabewert

0	Signatur erfolgreich
1	Signatur abgebrochen
100	Digitale Signatur wurde noch nicht gestartet
101	Digitale Signatur läuft noch
-1	Fehler bei der Initialisierung der Digitalen Signatur
-2	Es sind keine Signaturtexte konfiguriert.
-77	Signaturmodul ist nicht initialisiert
-79	zu signierende Datei existiert nicht.

Bemerkung Die Fehlertexte können nicht mit GetLastError() ermittelt werden.

Optionen

Beispiel

```
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
```

GetWDocPattern

Definition

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

Typ

Methode

Beschreibung

Kopiert eine Vorlagendatei für den gegebenen Objekttyp in das angegebene Verzeichnis

Parameter

lObjectType	gewünschter Objekttyp, muss W-Dokument sein
strPatternName	Alias-Name der Vorlage
strPath	Zielpfad
strEditorName	hier wird der Name des Bearbeitungsprogamms für diese Vorlage zurückgegeben
strFileName	vollständiger Pfad und Name der kopierten Vorlage wird hier zurückgegeben

Rückgabewert

0	kein Fehler
-7	Dokumenttyp unbekannt
-35	kein Vorlagen für diesen Typ definiert
-36	keine Vorlage mit diesem Namen gefunden
-37	Vorlage konnte nicht geholt werden
-60	Objekt ist kein W-Dokument
-104	Der angegebene Aliasname ist mehrfach vorhanden.
-105	Der angegebene Namespace wurde nicht gefunden.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Ist in der *as.cfg* der Eintrag STOREATSERVER=1 enthalten, so wird die Vorlagendatei vom Archivserver übertragen, andernfalls wird sie aus Vorlagenverzeichnis kopiert.

Optionen

Der Name der Vorlage kann auch mit dem gewünschten Namespace getrennt durch '::' angegeben werden.

Beispiel

GetWDocPatternNames

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

Typ Methode

Beschreibung Liefert alle Alias-Namen der W-Vorlagen, die mit diesem Dokumenttyp verknüpft sind.

Parameter

lObjectType	gewünschter Objekttyp, muss W-Dokument sein
strToken	Trennzeichen, das zur Trennung der Namen im Rückgabestring verwendet werden soll
strPatternName	hier werden die Namen zurückgegeben

Rückgabewert

0	kein Fehler
-7	Dokumenttyp unbekannt
-35	kein Vorlagen für diesen Typ definiert
-60	Objekt ist kein W-Dokument

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

GoToDocPage

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

Typ Methode

Beschreibung Navigiert in einem geöffneten Dokument zur angegebenen Seite.

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
lPageNum	Seitennummer

Rückgabewert

0	kein Fehler
-7	Dokumenttyp nicht bekannt
-26	Dokument-ID nicht bekannt
-91	Dokumenttyp wird nicht unterstützt
-92	Dokument nicht geöffnet
-93	Dokument hat keine Seiten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Funktion unterstützt nur Dokumenttypen mit den Haupttypen

1	X = Grauwert
2	D = Schwarz/Weiss
3	P = Farbe

Optionen

Beispiel

InfoWindow

Definition InfoWindow

Typ Eigenschaft

Beschreibung Liefert ein Objekt zur Steuerung des Infofensters.

Parameter

Rückgabewert ein Objekt zur Steuerung des Infofensters

Bemerkung Möchte man das Infofenster innerhalb eines Events benutzen, so ist es nicht nötig, sich das Objekt über diese Schnittstelle zu holen, da es in jedem Event unter dem Namen 'InfoWindow' angesteuert werden kann.
Hinweis: 'InfoWindow' verwendet zur Anzeige von HTML Funktionen des Internet Explorer Controls, Bestandteil des Betriebssystems. Aktuelle Standards werden durch dieses Control teilweise fehlerhaft oder gar nicht unterstützt.

Optionen

Beispiel

```
Dim a As Object

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

InsertFileList

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

Typ Methode

Beschreibung Fügt eine oder mehrere Dateien (je nach Haupttyp) als neues Dokument in das Archiv ein.

Parameter

nMainType	Haupttyp
strFileName	Datei mit den Namen der einzufügenden Dateien
lReturnObjectID	hier wird die Objekt-ID zurückgegeben
lReturnObjectType	hier wird der Objekttyp zurückgegeben

Rückgabewert

0	kein Fehler
-1	Einfügen fehlgeschlagen
-62	Es ist gerade ein Datenblatt geöffnet.
-69	Übergabedatei ist leer.

Der genaue Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Mögliche Haupttypen sind im Kapitel **Einführung** beschrieben.

Die Dateinamen in der Übergabedatei müssen durch **CR/LF** getrennt werden. Die Dateien müssen in dem, mit „**GetEnvironment**(0)“ ermittelten Pfad liegen. (Hinweis: Gibt diese Funktion, je nach Rechnerkonfiguration, kurze Dateinamen zurück, so müssen die Dateipfade in der Übergabedatei ebenso mit kurzen Dateinamen eingetragen werden.)

Der Benutzer kann im enaio® client festlegen, in welchen Schrank, bzw. in welchen Ordner das Dokument eingefügt werden soll. Zudem ist es möglich in die Ablage einzufügen und zuvor den Dokumenttyp festzulegen. Für die Haupttypen 1, 2 und 3 sind mehrere Dateien in der Übergabedatei zulässig. Bei alle anderen muss es genau eine sein.

Ist in der Übergabedatei zusätzlich eine Datei namens **FILEDATA.TXT** angegeben, dann können dort Indexdaten für die initiale Verschlagwortung des neu angelegten Dokumentes angegeben werden. (Hinweis: Diese Datei muss im Arbeitsverzeichnis „**GetEnvironment**(9)“ liegen und als letzte Datei im Anschluss an die Dokumentdateien in der Übergabedatei eingetragen sein.) In der Datei **FILEDATA.TXT** können die Indexfelder durch ihren Namen oder ihren internen Namen mit führendem und folgendem '%'-Zeichen angegeben werden. Der Benutzer bekommt im Normalfall die Verschlagwortungsmaske angezeigt und kann die Maske bearbeiten. Ist in dieser Datei jedoch zusätzlich der Parameter **ShowNoDialog**=1 angegeben, dann wird dem Benutzer am enaio® client keine Verschlagwortungsmaske angezeigt. Das Einfügen des neuen Dokumentes wird also ausgeführt, ohne dass der Benutzer Einfluss auf die Indexdaten hätte.

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Sie haben zwei Möglichkeiten zur Auswahl um eine oder mehrere Dateien als Dokument in das Archiv einzufügen:

1. Dateien und eventuelle **FILEDATA.TXT** Informationen in einer Zeichenkette

```
[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]
FELD0=Titel=Expose
FELD1=Bemerkung=Per Skript eingefügt
FELD2=%Author%=Hans Mustermann
```

2. **FILEDATA.TXT** in der Dateiliste:

```
[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
```


Optionen

Beispiel

Beispiel der Übergabedatei

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

Beispiel einer möglichen FILEDATA.TXT

```
[DATA]
FELD0=Titel=Expose
FELD1=Bemerkung= Per Skript eingefügt
FELD2=%Author%=Hans Mustermann
…
ShowNoDialog=1
```

In der FILEDATA.TXT kann nun auch ein Standort angegeben werden.

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

Folgende Regeln sind zu beachten:

- Es muss ein Dokumenttyp angegeben sein.
- Es muss ein Schranktyp angegeben sein.
- Es kann ein Registertyp angegeben sein.
- Wenn kein Registertyp angegeben ist, muss eine Schrank-ID angegeben sein.

- Ist ein Registertyp abgegeben, muss eine Register-ID angegeben sein. Eine Schrank-ID kann in diesem Fall entfallen.
- Ist ein Registertyp angegeben, muss er zum Schrank gehören.
- Der angegebene Dokumenttyp muss zum Schrank gehören.
- Ein weiteres Dokument des angegebenen Dokumenttyps muss am Standort (Schrank oder Register) anzulegen sein.

Beispiel:

```
[PREDEFTARGET]
DOKUMENTTYP=Bericht
SCHRANK=Patient
REGISTER=Aufenthalt
REGISTERIDENT=123456
```

Verletzen diese Angaben keine der Regeln, wird ein Dokument gemäß den Angaben in der Sektion „DATA“ an den angegeben Standort angelegt.

Ein Dokument kann auch in der Ablage des Benutzers angelegt werden.

Beispiel:

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


InsertFileListS

Entspricht 'InsertFileList' und kann aus VB-Script verwendet werden.

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

InsertIntoArchive

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

Typ Methode

Beschreibung Ermöglicht das Einfügen eines neuen Ordners in einen Schrank.

Parameter

strFilename	Übergabedatei (Aufbau siehe Bemerkung)
***lpReturnObjectID**	hier wird die Objekt-ID zurückgegeben
***lpReturnObjectType**	hier wird der Objekttyp zurückgegeben.

Rückgabewert

0	wenn kein Fehler
-1	Ordner einfügen fehlgeschlagen
-30	ein oder mehrere Pflichtfelder nicht ausgefüllt
-24	Feldnamen konnten nicht aufgelöst werden
-28	Feldwert für ein gegebenes Feld nicht zulässig
-31	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
-32	Übergabedatei existiert nicht
-47	Kein Schreibrecht auf dieses Objekt
-64	Serverfehler ist aufgetreten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Übergabedatei hat folgenden Aufbau:

```
[EINFÜGEN]
SCHRANK=
FELD1=
FELD2=
...
FELDn=
```

Und folgende Zeilen sind einzutragen, um Werte für Tabellencontrols festzulegen, wobei Trennzeichen dem Zeichen chr(17) entspricht.

```
[Tabelle@TABELLENNAME]
Zeile0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Zeile1={Spalte1Zeile2(Trennzeichen)Spalte2Zeile2}
```

Der Parameter **strFilename** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen

PFLICHTFELDER=[0;1]
Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

VORBELEGUNG=[0;1]
belegt Felder vor, die nicht extra angegeben wurden. Vorgabe: 0

CHECKKEYFIELDS=[0;1]
Für den Wert 0 wird die Schlüsselfeldprüfung deaktiviert. Vorgabe: 1

Beispiel Einfügen eines Ordners in den Kunden-Schrank

```
[EINFÜGEN]
SCHRANK=Kunde
FELD1=Klasse=Kunde
FELD2=Vorlage=10.03.1997
FELD3=optimal_AS=1
```

Hinweis: Es findet keine Überprüfung von Schlüsselfeldern statt. Die Nummerierung darf keine Lücke aufweisen. Angaben nach einer Lücke werden ignoriert!

InsertIntoArchiveS

Entspricht 'InsertIntoArchive' und kann aus VB-Script verwendet werden.

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

InsertIntoDocument

Definition

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

Typ

Methode

Beschreibung

Ermöglicht das Einfügen eines neuen Dokuments in einen Schrank.

Parameter

strFilename	Übergabedatei (Aufbau s. Bemerkung)
***lpReturnObjectID**	Objekt-ID wird zurückgegeben
***lpReturnObjectType**	Objekttyp wird zurückgegeben

Rückgabewert

0	wenn kein Fehler
-1	Dokument einfügen fehlgeschlagen
-2	kein Ordner angegeben
-3	Ordner unbekannt
-6	Kein Dokumenttyp angegeben
-7	Dokumenttyp unbekannt
-9	Dokumenttyp gehört nicht zum angegebenen Ordner
-10	Ordnerkennung fehlt
-24	Feldnamen konnten nicht aufgelöst werden
-28	Feldwert für ein gegebenes Feld nicht zulässig
-30	ein oder mehrere Pflichtfelder nicht ausgefüllt
-31	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
-32	Übergabedatei existiert nicht
-47	Kein Schreibrecht auf dieses Objekt
-64	Serverfehler ist aufgetreten
-89	Das Objekt darf im Zielordner/-register nicht angelegt werden.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Die Übergabedatei hat folgenden Aufbau:

```
[EINFÜGEN]
SCHRANK=
DOKUMENT=
SCHRANK-ID=
REGISTER-ID=
MAINTYPE=
FELD1=
FELD2=
...
FELDn=
DATEI1=
DATEI2=
...
DATEIn=
```

Und folgende Zeilen sind einzutragen, um Werte für Tabellencontrols festzulegen, wobei Trennzeichen dem Zeichen chr(17) entspricht.

```
[Tabelle@TABELLENNAME]
Zeile0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Zeile1={Spalte1Zeile2(Trennzeichen)Spalte2Zeile2}
```

Der Parameter **strFilename** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein.

Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Hinweis: Der Haupttyp des Zieldokuments muss mit **MAINTYPE=#** angegeben werden, wobei # für die Haupttyp-Nr. steht. Mögliche Haupttypen sind im Kapitel **Einführung** beschrieben.

Mit dieser Funktion sollten keine Verweise auf Dokumente erzeugt werden, die der Variantenverwaltung unterliegen.

Optionen

PFLICHTFELDER=[0;1]
Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

SHOWTEMPLATES=[0;1]
Für den Wert 1 wird für W-Dokumente der Vorlagendialog angezeigt. Die ausgewählte Vorlage wird eingefügt. Der Eintrag DATEI1 muss dann nicht gemacht werden. Wird keine Vorlage ausgewählt, so wird eine Fundstelle angelegt.

ENABLEOPTIONS=[0;1]
Für den Wert 1 werden die Optionen des Vorlagendialogs aktiviert.

ARCHIVIERBAR=[0;1]
Setzt, wenn Dateien angegeben, den Archivstatus.

DATEILÖSCHEN=[0;1]
Löscht die angegebenen Quelldateien. Vorgabe ist 0.

VORBELEGUNG=[0;1]
belegt Felder vor, die nicht extra angegeben wurden. Vorgabe: 0

CHECKKEYFIELDS=[0;1]
Für den Wert 0 wird die Schlüsselfeldprüfung deaktiviert. Vorgabe: 1

FOREIGN-ID=Dokument-ID
SYSTEM-ID=
Diese beiden Parameter können für das Anlegen von schrankübergreifenden Verweisen benutzt werden. Dazu setzt man die FOREIGN-ID auf die Dokument-ID, auf die verwiesen werden soll und die SYSTEM-ID auf Null und gibt die entsprechende SCHRANK-ID an.

Um Mehrfachparameter anzufügen, sind in der Datei neue Sektionen einzufügen. Diese Sektionen beginnen mit **MULTI_** gefolgt vom internen Feldbezeichner des Mehrfachfeldes. (z.B.: MULTI_FELD1). Dann folgen Daten, die in den Mehrfachparametern eingetragen werden sollen, z.B.:

```
[MULTI_FELD1]
Data1=Seitennummer,Text
Data2=Seitennummer,Text
```

Beispiel

Einfügen in das Kunden-Dokument, 'Eingangsbeleg‘

```
[EINFÜGEN]
SCHRANK=Kunde
REGISTER=Register
DOKUMENT=Eingangsbeleg
SCHRANK-ID=4711
REGISTER-ID=0
FELD1=Klasse=Kunde
FELD2=Vorlage=10.03.1997
DATEI1=c:\temp\image.tif
[MULTI_FELD1]
DATA1=1,Peter
DATA2=2,Hans
```

Anlegen eines schrankübergreifenden Verweises:

```
[EINFÜGEN]
SCHRANK=Kunde
DOKUMENT=Anlagendokumentation
SCHRANK-ID=6296
SYSTEM-ID=0
FOREIGN-ID=6305
```

Hinweis: Es findet keine Überprüfung von Schlüsselfeldern statt. Die Nummerierung darf keine Lücke aufweisen. Angaben nach einer Lücke werden ignoriert. Die Angabe einer Schrank-ID kann entfallen, wenn eine Register-ID angegeben wird. Wird keine Register-ID angegeben (REGISTER-ID=0), dann wird das Dokument in der obersten Ebene erzeugt. Wird eine Register-ID angegeben, dann wird das Dokument im angegebenen Register erzeugt.

InsertIntoDocumentS

Entspricht 'InsertIntoDocument' und kann aus VB-Script verwendet werden.

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

InsertIntoRegister

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

Typ Methode

Beschreibung Einfügen eines neuen Registers

Parameter

strFilename	Übergabedatei (Aufbau siehe Bemerkung)
***lpReturnObjectID**	hier wird die Objekt-ID zurückgegeben
***lpReturnObjectType**	hier wird der Objekttyp zurückgegeben.

Rückgabewert

0	wenn kein Fehler
-1	Register einfügen fehlgeschlagen
-2	kein Ordner angegeben
-3	Schrank unbekannt
-4	kein Register angegeben
-5	Register unbekannt
-8	Register gehört nicht zum Ordner
-24	Feldnamen konnten nicht aufgelöst werden
-28	Feldwert für ein gegebenes Feld nicht zulässig
-30	ein oder mehrere Pflichtfelder nicht ausgefüllt
-31	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
-32	Übergabedatei existiert nicht
-47	Kein Schreibrecht auf dieses Objekt
-64	Serverfehler ist aufgetreten
-89	Das Objekt darf im Zielordner/-register nicht angelegt werden.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Übergabedatei hat folgenden Aufbau:

```
[EINFÜGEN]
SCHRANK=
REGISTER=
SCHRANK-ID=
REGISTER-ID=
FELD1=
FELD2=
...
...
FELDn=
```

Und folgende Zeilen sind einzutragen, um Werte für Tabellencontrols festzulegen, wobei Trennzeichen dem Zeichen chr(17) entspricht.

```
[Tabelle@TABELLENNAME]
Zeile0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Zeile1={Spalte1Zeile2(Trennzeichen)Spalte2Zeile2}
```

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen

PFLICHTFELDER=[0;1]
Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

VORBELEGUNG=[0;1]
belegt Felder vor, die nicht extra angegeben wurden. Vorgabe: 0

CHECKKEYFIELDS=[0;1]
Für den Wert 0 wird die Schlüsselfeldprüfung deaktiviert. Vorgabe: 1

Beispiel

Einfügen in das Kunden-Register **Register**.

```
[EINFÜGEN]
SCHRANK=Kunde
REGISTER=Register
SCHRANK-ID=4711
REGISTER-ID=0
FELD1=Typ=Brief
FELD2=Text=Test
```


Hinweis: Es findet keine Überprüfung von Schlüsselfeldern statt. Die Nummerierung darf keine Lücke aufweisen. Angaben nach einer Lücke werden ignoriert!

Der Eintrag *PFLICHTFELDER* =1 bewirkt ein Überprüfung der Pflichtfelder. Der Standardwert ist 0.

Der Eintrag VORBELEGUNG=1 füllt alle Felder mit Vorbelegung, die nicht extra angegeben wurden. Default: 0

Die Angabe einer Schrank-ID kann entfallen, wenn eine Register-ID angegeben wird. Wird keine Register-ID angegeben (REGISTER-ID=0), dann wird das Register in der obersten Ebene erzeugt. Wird eine Register-ID angegeben, dann wird das Register im angegebenen Register erzeugt. Ist sowohl Schrank-ID als auch Register-ID angegeben, wird zusätzlich überprüft, ob sich das angegebene Register im angegebenen Ordner befindet.

InsertIntoRegisterS

Entspricht 'InsertIntoRegister' und kann aus VB-Script verwendet werden.

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

InsertNewDMSObject

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

Typ Methode

Beschreibung Ermöglicht das Einfügen eines neuen Objekts – Ordner, Register oder Dokument – über das Öffnen eine Indexdatenmaske in enaio® client.

Parameter

lType	DMS-Objekttyp des neu anzulegenden Objekts

lPosIdent ID des Standorts des neu anzulegenden Objekts, bei Ordnern nicht ausgewertet.
strFilename Übergabedatei mit Vorbelegung der Indexdatenfelder und im Fall von Dokumentdateien Name und Pfad.
newObjIdent ID des neu angelegten Objekts

Rückgabewert

-120 Angegebener DMS-Objekttyp unbekannt.
-121 Standort für die Neuanlage kann nicht ermittelt werden.
-122 Am angegebenen Standort kann kein neues Objekt von angegebenen Typ angelegt werden.
-123 Eingabe abgebrochen.
-124 Erforderliches Index-Schreibrecht zur Neuanlage eins Objekts vom angegeben Typ nicht gegeben.
-125 Ein modaler Dialog ist geöffnet.
-126 Eine Neuanlage läuft bereits.
-127 Anzulegender DMS-Objekttyp nicht im Schrank des Standorts vorhanden.

Folgende Rückgabewerte erzeugen ein Dokument ohne Seiten:
1 Erforderliches Objekt-Schreibrecht zum Speichern von Dateien beim angegeben Typ nicht gegeben.
2 Mindestens eine der angegebenen Dateien existiert nicht.
3 Mindestens eine angegebene Datei passt nicht zum anzulegenden Dokumenttyp.
4 Zu viele Dateien für den anzulegenden Dokumenttyp angegeben.

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Indexdaten als Vorbelegung sind optional. Sie können so angegeben werden:

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

Interne Namen für Felder werden mit Prozentzeichen eingeklammert.

Dateien können so angegeben werden:

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

Über `DeleteAfterInsert=1` werden die Dateien nach der Übernahme gelöscht.
Ohne Dateien wird in enaio® client nach dem Indexieren das entsprechende Modul zum Erstellen von Dateien geöffnet.
Über `OnlyIndexData=1` wird kein Modul geöffnet sondern ein Dokument ohne Seiten angelegt. Ist der Dokumenttyp ohne Seiten, wird ebenfalls kein Modul geöffnet.
Über `ShowNoDialog=1` wird keine Indexdatenmaske geöffnet sondern das Objekt ohne Benutzeraktion angelegt.

Optionen -

Beispiel -

LicLogin

Definition	LicLogin(strModulName)
Typ	Methode
Beschreibung	Lizensiert das angegebene Modul.
Parameter	**strModulName** Name des zu lizensierenden Moduls
Rückgabewert	**0** wenn das Modul lizensiert werden konnte **-1** allgemeiner Fehler in der Funktion **>0** Lizenzfehlernummer vom Applikationsserver
Bemerkung	Ist das angegebene Modul bereits für die aktuelle Station lizensiert, wird diese Lizensierung benutzt.
Optionen	
Beispiel	

LicLogout

Definition	LicLogout(strModulName)
Typ	Methode
Beschreibung	Gibt die Lizenz für das angegebene Modul frei.
Parameter	**strModulName** Name des freizugebenden Moduls
Rückgabewert	**0** wenn die Lizensierung freigegeben werden konnte **-1** wenn die Lizensierung nicht freigegeben werden konnte
Bemerkung	Nur zuvor mit **LicLogin** lizensierte Module können mit **LicLogout** Wieder freigegeben werden.
Optionen	
Beispiel	

LinkDocuments

Definition

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

Typ

Methode

Beschreibung

Stellt eine Verknüpfung zwischen zwei Dokumenten her.

Parameter

lObjectID1	ID des ersten Dokuments
lObjectType1	Typ des ersten Dokuments
lObjectID2	ID des zweiten Dokuments
lObjectType2	Typ des zweiten Dokuments

Rückgabewert

0	kein Fehler
-20	einer der Objekttypen ist kein Dokumenttyp, der fehlerhafte Typ ist im Fehlertext angegeben
-29	eine oder beide Objekt-IDs sind unbekannt
-34	diese Verknüpfung existiert bereits
-86	Objektindizes sind identisch
-87	Objekte vom Typ Mappe sind nicht zulässig
-88	Objekte aus der Ablage sind nicht zulässig

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Die Verknüpfung zwischen beiden Dokumenten wird in deren Notizen gespeichert.

Optionen

Beispiel

MergeArchives

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

Typ Methode

Beschreibung Vereinigt die Inhalte zweier Ordner.

Parameter

lSourceID	Quellordner
lTargetID	Zielordner
lObjectType	Objekttyp

Rückgabewert

0	kein Fehler
-22	Objekttyp unbekannt
-47	Kein Recht zum Anlegen von Objekten.
-80	Quellordner unbekannt
-81	Zielordner unbekannt
-82	Aktualisierung der Datenbank fehlgeschlagen

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

MoveObject

Definition

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

Typ

Methode

Beschreibung

Verschiebt ein Objekt(Register oder Dokument).

Parameter

lObjectID	ID des zu verschiebenden Objektes
lObjectType	zugehöriger Objekttyp
lFolderID	Ordner-ID des Zielordners.
lRegisterID	Register-ID des Zielregisters

Rückgabewert

0	kein Fehler
-5	Registertyp unbekannt (Register-ID gehört zu falschem Objekttyp)
-7	Dokumenttyp unbekannt
-13	Register-ID unbekannt
-14	Ordner-ID unbekannt
-29	Objekt-ID ist ungültig
-68	Ordner können nicht verschoben werden.
-82	Aktualisierung der Datenbank fehlgeschlagen
-90	Das angegebene Objekt ist eine Verweiskopie
539	Objekttyprelationsverletzung

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Wird eine Register-ID > 0 angegeben, so wird die Ordner-ID nicht beachtet. Sind Register- u. Ordner-ID = 0, so wird das Objekt in die Root verschoben.

Verweisdokumente können nicht verschoben werden.

Optionen

Beispiel

OleDdeRequest

Definition	OleDdeRequest	(String String	strFunctionName, strParameter)
Typ	Methode		
Beschreibung	Diese Funktion ist veraltet und wird nicht mehr unterstützt.		
Parameter	**strFunctionName** **strParameter**	Name der Funktion Parameter als Zeichenkette	
Rückgabewert			
Bemerkung	Statt dieser Funktion, sind die COM-Funktionen direkt aufzurufen.		
Optionen			
Beispiel			

OpenAboDialog

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

Typ

Methode

Beschreibung

Zeigt den Abonnement-Dialog an.

Parameter

lHwnd	Fensterhandle
lObjectID	ID des Dokuments
lObjectType	Dokumenttyp
lFlags	siehe Bemerkung
strInfoText	Infotext, wird in das Info-Feld des Dialogs eingetragen

Rückgabewert

1	Abbruch des Dialogs durch den Benutzer
0	kein Fehler
-7	Objekttyp unbekannt
-26	Objekt-ID ist unbekannt
-62	Es ist bereits ein Dialog geöffnet

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Durch Angabe von Flags können bestimmte Checkboxen des Dialogs gesetzt werden. Die Flags können kombiniert werden.

1	Checkbox Dokument geändert
2	Checkbox Indexdaten geändert
4	Checkbox Objekt angelegt
8	Checkbox Objekt gelöscht

Optionen

Beispiel

OpenDataDlg

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

Typ Methode

Beschreibung Öffnet das Datenblatt des angegebenen Objekts.

Parameter

lObjectID	Objekt-ID
lObectType	Objekttyp
nMode	Bearbeitungsmodus (siehe Bemerkung)

Rückgabewert

0	kein Fehler
-7	Objekt unbekannt
-11	Dokumentkennung nicht zulässig (= 0)

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Anzeige erfolgt durch den enaio® client.
Mögliche Werte für nMode sind:

0	das Datenblatt wird schreibgeschützt und modal geöffnet
1	das Datenblatt wird modal zur Bearbeitung geöffnet
2	das Datenblatt wird schreibgeschützt und nicht modal geöffnet
3	das Datenblatt wird nicht modal zur Bearbeitung geöffnet

Optionen

Beispiel

OpenObjectID

Definition

```
OpenObjectID (long  lObjectID,
              long  lObjectType,
              short nMode)
```

Typ

Methode

Beschreibung

Öffnet das angegebene Objekt.

Ist das Objekt ein **Dokument**, dann wird das Image angezeigt bzw. das entsprechende Dokument geladen(W-Dokument). Hat das Dokument keine Images, wird das Datenblatt geöffnet.

Ist das Objekt ein **Ordner** bzw. **Register**, dann wird der entsprechende Ordner geöffnet.

Parameter

lObjectID	ID des zu öffnenden Objekts
lObjectType	Objekttyp
nMode	Bewirkt bei zu öffnenden W-Dokumenten(und nur da), ob das Dokument schreibgeschützt geöffnet wird oder zur Bearbeitung. Ist dieser Parameter **0**, dann wird das Dokument wie bisher mit Schreibschutz geöffnet. Ist der Modus **= 1**, dann wird das Dokument zur Bearbeitung geöffnet. nMode = **2**, Dokument wird schreibgeschützt und es wird nicht die aktive Variante geöffnet, falls das Dokument der Variantenverwaltung unterliegt, sondern genau das Dokument mit angegebenen ID nMode = **3**, Dokument wird nicht schreibgeschützt und es wird nicht die aktive Variante geöffnet, falls das Dokument der Variantenverwaltung unterliegt, sondern genau das Dokument mit angegebenen ID

Rückgabewert

0	kein Fehler
-11	wenn lObjectID = 0

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Die Anzeige erfolgt durch den enaio® client.

Optionen

Beispiel

Ein W-Dokument soll mit Schreibschutz geöffnet werden:

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

OpenResultList

Definition	OpenResultList	(String String	strIDList, strTitle)
Typ	Methode		
Beschreibung	Öffnet eine Trefferliste mit den übergebenen Objekten.		
Parameter	**strIDList**	Liste mit den Objekten in der Form: ID,Objekttyp;ID1,Objekttyp1…	
	strTitle	Fenstertitel	
Rückgabewert	**0** kein Fehler **-1** Trefferliste konnte nicht erzeugt werden.		
Bemerkung			
Optionen			
Beispiel			

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

OpenObjectIDEx

Definition

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

Typ

Methode

Beschreibung

Öffnet das angegebene Objekt. Ist das Objekt ein Dokument wird das Image angezeigt bzw. das entsprechende Dokument geladen(W-Dokument). Hat das Dokument keine Images, wird das Datenblatt geöffnet. Ist das Objekt ein Schrank bzw. Register wird der entsprechende Ordner geöffnet.

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
bWriteProtected	**TRUE**, wenn das Dokument schreibgeschützt geöffnet werden soll
bActivateIfOpen	**TRUE**, wenn ein bereits geöffnetes Fenster dieses Typs aktiviert werden soll.

Rückgabewert

0	kein Fehler
-11	Objekt-ID ungültig

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

OpenURL

Definition	OpenURL	(String BOOL BOOL	strURL, bShowAddressBar, bOpenNewWindow)
Typ	Methode		
Beschreibung	Öffnet eine URL.		
Parameter	**strURL** **bShowAddressBar** **bOpenNewWindow**	zu öffnende Adresse **FALSE**, wenn die Adressleiste ausgeblendet werden soll **TRUE**, wenn die Adresse in einem neuen Fenster geöffnet werden soll	
Rückgabewert	**0** **-1**	kein Fehler URL-Fenster konnte nicht geöffnet werden	
Bemerkung			
Optionen			
Beispiel			

OpenWorkItem

Definition OpenWorkItem (String strWorkItemID)

Typ Methode

Beschreibung Öffnet einen Vorgangsschritt im Eingangskorb des angemeldeten Benutzers.

Parameter

strWorkItemID	Id des zu öffnenden Vorgangsschrittes.

Rückgabewert

0	kein Fehler
-113	Workflowmodul ist nicht verfügbar
-114	Id des Vorgangsschrittes nicht im Eingangskorb gefunden
-115	Interner Fehler beim Öffnen des Vorgangsschrittes

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Diese Funktion öffnet ausschließlich Vorgangsschritte, die im Eingangskorb des aktuell angemeldeten Benutzers liegen.

Optionen

Beispiel

PrintDocumentID

Definition PrintDocumentID (String strParam)

Typ Methode

Beschreibung Druckt die angegebenen Objekte, wenn die Objekte vom Typ Dokument sind. Ausgenommen sind Dokumente vom Typ W.

Parameter **strParam** String mit folgendem Inhalt:

```
INDEX,OBJEKTTYP
```

Rückgabewert

0 kein Fehler
-1 Objekt nicht vom Typ Dokument
-7 Dokumenttyp unbekannt
-40 Übergabeparameter falsch
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Es wird der Druckdialog von enaio®client geöffnet.

Es können auch mehrere Objekte angegeben werden. Der Parameterstring hätte dann die Form:

```
INDEX1,OBJEKTTYP ... INDEXn,OBJEKTTYP
```

Die einzelnen Objektangaben trennt ein Leerzeichen. Es ist weiterhin darauf zu achten, dass nur genau ein Objekttyp verwendet wird.

Es kann auch ein Dateiname übergeben werden.
Die Datei hätte dann folgenden Aufbau:*

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

* **\r\n** steht für einen Zeilenumbruch, **CR** und **LF**.

Optionen

Beispiel

RefreshFolderWindow

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

Typ Methode

Beschreibung Aktualisiert alle im Client geöffneten Ordner-/Registerfenster, die durch die Angabe von ID und Typ spezifiziert sind. Ist das Fenster das aktive Fenster kann durch die Angabe von ID2Select auch ein neues Objekt selektiert werden.

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
lObjectID2Select	Neu zu selektierendes Objekt (**0** wenn alte Selektion erhalten bleiben soll)

Rückgabewert

Bemerkung Aktualisiert werden durch diese Funktion alle offenen Ordnerfenster des angegebenen Ordners/Registers. Eine neue Selektion auf das der Funktion mitgegebene Objekt wird allerdings nur wirksam, wenn das Fenster das aktive Fenster des Client ist.

Das Aktualisieren der Fenster kann je nach System einige Zeit beanspruchen. Während dieser Zeit wird der Client vorübergehend unbedienbar. Daher ist ein Einsatz dieser Funktion gut abzuwägen.

Optionen

Beispiel

ScanDocument

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

Typ Methode

Beschreibung Ermöglicht das Anfügen von Images an ein bestehendes Dokument.

Parameter

lObjectID	Objekt-ID
lObjectType	Objekttyp
nMode	Modus (siehe Bemerkung)

Rückgabewert

0	kein Fehler
-11	Dokumentkennung nicht zulässig
-19	Objekt ist keinem Ordner zugeordnet
-20	Angegebener Objekttyp ist kein Dokumenttyp!
-21	Dokument bereits archiviert

Der Fehlertext kann mit **GetLastError()** ermittelt werden

Bemerkung

Hat nMode den Wert 1, dann wird vor dem Öffnen des Scan-Fensters die Verschlagwortungsmaske geöffnet. Wird diese Maske mit ‚Abbrechen' verlassen, dann wird das Scan-Fenster nicht geöffnet.

Es wird der Scandialog des Clients benutzt.

Diese Funktion kann nicht dazu benutzt werden, um bereits archivierte Dokumente zu ändern.

Optionen

Beispiel

ShowVariantsDialog

Definition

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

Typ

Methode

Beschreibung

Zeigt den Dialog der Variantenverwaltung an.

Parameter

lObjectID	ID des Dokuments
lObjectType	Dokumenttyp
bCreateSingleVariant	gibt an, ob der Benutzer nur eine Variante anlegen darf
bAllowDragDrop	gibt an, ob Drag&Drop in diesem Dialog erlaubt ist
bOpenAfterCreation	gibt an, ob die neu angelegte Variante gleich geöffnet werden soll
bSetNewVersionActive	gibt an, ob die neu angelegte Variante als aktiv markiert wird
lHwnd	Fensterhandle
vRetNewObjectID	hier werden die ID's der neu angelegten Varianten durch Semikolon getrennt zurückgegeben

Rückgabewert

1	Abbruch des Dialogs durch den Benutzer
0	kein Fehler
-7	Objekttyp unbekannt
-26	Objekt-ID ist unbekannt
-60	Objekt ist nicht vom Typ W-Dokument
-62	Es ist bereits ein Dialog geöffnet
-93	Objekt hat keine Seiten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

SelectObject

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

Typ Methode

Beschreibung Öffnet eine Trefferliste zur Auswahl eines Objekts.

Parameter

lObjectID	ID des Ordners oder Registers
lObjectType	Objekttyp des Ordners oder Registers
bMultiSelect	**TRUE**, für Mehrfachselektion
strTitle	Fenstertitel

Rückgabewert

1	Auswahl wurde gestartet
-25	Objekt ist kein Ordner oder Register
-26	Objekt-ID nicht zulässig
-27	Es ist bereits eine Trefferliste zur Objektauswahl geöffnet

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die geöffnete Trefferliste besitzt zwei neue Buttons auf der Toolbar zur Auswahl der selektierten Objekte oder zum Abbruch der Auswahl.

Optionen

Beispiel

SendMail

Definition	SendMail	(String short	strFileList, nDelete)
Typ	Methode		
Beschreibung	Öffnet ein Formular zum Versenden einer E-Mail		
Parameter	**strFileList**	Dateiliste, die an die Mail angehangen wird	
	nDelete	**0**, wenn die Dateien nicht gelöscht werden sollen, sonst **1**	
Rückgabewert			
Bemerkung			
Optionen			
Beispiel			

SendMailMapi

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

Typ

Methode

Beschreibung

Versendet eine E-Mail ohne Dialog

Parameter

strFrom	Absender
StrTo	Empfänger
StrCc	Kopie an
StrBcc	Blindkopie an
StrSubject	Betreff
StrBody	Text
strFiles	Liste mit anzuhängenden Dateien durch Komma getrennt
lReserved	unbenutzt
nCompress	unbenutzt

Rückgabewert

0	kein Fehler
<> 0	Fehler

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Wenn in der *as.cfg* in der Sektion [CLIENT] der Schlüssel **SendMail=OUTLOOK** eingetragen wird, so wird vor dem Versenden das Outlook E-Mail Fenster geöffnet. Somit können auch die enaio®-Outlook Addins genutzt werden.
In der Kombination Client 32-bit und Outlook 64-bit wird der Befehl **SendMailMapi** nur ausgeführt, wenn **SendMail=OUTLOOK** in der *as.cfg* eingetragen ist. Dann wird immer ein Outlook-Dialog geöffnet.

Der Parameter **strFrom** wird in den aktuellen Mapi-Versionen ignoriert.

Optionen

Beispiel

SetPlannedRetention

Definition

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

Typ

Methode

Beschreibung

Setzt das geplante Retention-Datum für ein zu archivierendes Dokument.

Parameter

lObjectID	ID des Objekts
lObjectType	Typ des Objekts
strRetentionDate	das Retention-Datum im Format **DD.MM.YYYY**

Rückgabewert

0	Retention-Datum erfolgreich gesetzt
-1	Retention-Datum konnte nicht gesetzt werden
-7	Objekttyp unbekannt
-22	Objekt mit angegebener ID existiert nicht
-38	Ungültiger Dokumenttyp (es wurde ein Ordner oder Registertyp angegeben)
-40	Ungültiger Parameter wurde übergeben

Bemerkung

Die Fehlertexte können mit GetLastError() ermittelt werden.

Optionen

Beispiel

```
Dim a As Object

Set a = CreateObject("optimal_as.application")

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

SetResultListSelection

Definition	SetResultListSelection (String strObjectIDs)
Typ	Methode
Beschreibung	Markiert Objekte in der aktiven Trefferliste.
Parameter	**strObjectIDs** IDs der zu markierenden Objekte durch Komma getrennt
Rückgabewert	Anzahl der markierten Objekte
Bemerkung	Die Markierung wird nur im gerade aktiven Fenster vorgenommen.
Optionen	
Beispiel	

SetSignatureProperty

Definition SetSignatureProperty (String strPropertyName, String strValue)

Typ Methode

Beschreibung Setzt eine Eigenschaft für die digitale Signatur eines Dokuments

Parameter

strPropertyName Name der Eigenschaft
strValue Wert der Eigenschaft

Rückgabewert

Bemerkung Diese Funktion sollte unmittelbar vor dem Aufruf von SignDocument, SignDocumentEx oder SignDocumentList aufgerufen werden, um bestimmte Eigenschaft der Signatur, z.B. Signaturtext bestimmen zu können.

Folgende Eigenschaften können gesetzt werden:

SIGTYP[0-n]	Wert=Typ der Signatur
SIGTEXT[0-n]	Wert=Zum Signaturtyp zugehöriger Signaturtext
DEFAULTTYP	Wert =0-n, Nummer der Vorauswahl
CAPTION	Wert=Überschrift für Signaturdialog
LOCATION	Wert=Ort für die Signatur
CERTFLAGS	Wert=Filter für Zertifikatstypen mögliche Werte: 64=Zugelassen (Kugelschreiber Unterschrift) 128=Digitale Signatur (Bleistift Unterschrift) 192=Beides
ISSUERFILTER	Wert=Filter für anzuzeigende Zertifikate. Dies bezieht auf den Aussteller eines Zertifikates.

Optionen

Beispiel

```
Set a = createobject("optimal_as.application")
a.SetSignatureProperty "SIGTYP0", "Kenntnisnahme "
a.SetSignatureProperty "SIGTEXT0", "Der Inhalt des
vorliegenden …"
a.SetSignatureProperty "SIGTYP1", "Inhaltliche Richtigkeit"
a.SetSignatureProperty "SIGTEXT1", "Der Inhalt des
vorliegenden…"
a.SetSignatureProperty "DEFAULTTYP", "1"

a.SetSignatureProperty "CAPTION", "Elektronische Signatur "
a.SetSignatureProperty "LOCATION", "Berlin "

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

a.SignDocument 4711,65554
```

SetWaitingCursor

Definition SetWaitingCursor (short sShow)

Typ Methode

Beschreibung Zeigt den Mauszeiger als Sanduhr.

Parameter **sShow** Sanduhr einschalten (1) und wieder ausschalten (0).

Rückgabewert Kein Rückgabewert

Bemerkung Muss am Skriptende wieder ausgeschaltet werden.

Optionen -

Beispiel

```
SetWaitingCursor(1)

Dim dteWait

dteWait = DateAdd("s", 10, Now())

Do Until (Now() > dteWait)
Loop

SetWaitingCursor(0)

ResultCode=1
WriteToFile
```

SignDocument

Definition	SignDocument	(long long	lObjectID, lObjectType)

Typ Methode

Beschreibung Ruft den Dialog zur Digitalen Signatur eines Dokuments auf.

Parameter
lObjectID ID des zu signierenden Dokuments
lObjectType Typ des zu signierenden Dokuments

Rückgabewert Kein Rückgabewert

Bemerkung Die Funktion ruft die Signatur in einem eigenen Thread auf. Um den Rückgabewert zu erhalten kann die Funktion GetSignDocumentResult benutzt werden.

Optionen -

Beispiel

```
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
```

SignDocumentEx

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

Typ Methode

Beschreibung Ruft den Dialog zur Digitalen Signatur eines Dokuments auf.

Parameter

lObjectID	ID des zu signierenden Dokuments
lObjectType	Typ des zu signierenden Dokuments
bWaitForCompletion	gibt an, ob die Funktion auf die Beendigung der Signatur warten soll.
vRetSignText	der gewählter Signaturtext wird hier zurückgegeben

Rückgabewert 0 Signatur erfolgreich, erfolgreich gestartet
Siehe **GetSignDocumentResult**

Bemerkung Die Funktion ruft die Signatur in einem eigenen Thread auf. Um den Rückgabewert zu erhalten kann die Funktion **GetSignDocumentResult** benutzt werden.

Optionen

Beispiel

```
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
```

StartArchiveRequest

Definition StartArchiveRequest (String strFileName)

Typ Methode

Beschreibung Startet eine Schrankanfrage

Parameter **strFileName** Name der Anfragedatei

Rückgabewert *Dateiname* oder **READY**, wenn ANFRAGEFENSTER <> 0.
Leerer String, wenn ein Fehler aufgetreten ist.
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Anfragedatei muss folgenden Aufbau haben:

```
[ANFRAGE]
SCHRANK =
KLAUSEL1=
KLAUSEL2=
...
...
KLAUSELn=
DATENFELDER=0(1)
DATENHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
VOLLTEXT=
```

Für DATENFELDER=1 kann jetzt zusätzlich eine Sektion namens: [ANFRAGEFELDER] bestimmt werden. Hier können alle Felder angegeben werden, die in der Trefferliste in der Datei erscheinen sollen, dabei werden auch nur diese Felder angefragt.

Beispiel:
Der Dokumenttyp ‚Krankendossier' hat 3 Felder: **Formulartyp**, **Thema** und **Bemerkung**. Bei der Anfrage sollen aber nur die Werte für **Thema** und **Bemerkung** angefragt werden, dann sieht die Sektion [ANFRAGEFELDER] so aus:

```
[ANFRAGEFELDER]
Feld0=Thema
Feld1=Bemerkung
```

Die Nummerierung der Felder muss durchgehend sein! Diese Funktionalität gilt auch für die Funktionen **StartRegRequest** und **StartDocRequest**.

Sind in den zurückgelieferten Feldwerten Zeilenumbrüche vorhanden, so werden diese durch das ASCII – Zeichen Nr. 17 ersetzt, somit bleibt es dem Programmierer überlassen, bei Ausgabe der Werte dieses Zeichen wieder in einen Zeilenumbruch zurück zu ersetzen, um eine korrekte Ausgabe zu gewährleisten. Die Rückgabedatei hat folgenden Aufbau:

```
INDEX1,OBJEKTTYP
INDEX2,OBJEKTTYP
...
...
INDEXn,OBJEKTTYP
```

Eine leere Datei ist ein gültiges Ergebnis.

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen

DATENFELDER:
Wird diese Option mit ‚DATENFELDER=1'eingeschaltet, dann werden zusätzlich zu dem Index u. Objekttyp alle weiteren Daten des Objekts in die Datei geschrieben. Die Trennung der einzelnen Spalten erfolgt durch das Sonderzeichen <TAB>.

Eine Zeile in der Ergebnistabelle sähe dann z.B. so aus:

```
1234,131071<TAB>Heiner<TAB>Lauterbach<TAB>Schauspieler<CR><L
F>
```

DATENHEADER:
Wird diese Option mit ‚DATENHEADER=1' eingeschaltet, dann wird in der ersten Zeile der Ergebnisdatei Spaltenüberschriften geschrieben. Die erste Zeile in der Ergebnistabelle sähe dann z.B. so aus:

```
OSID,OSTYPE<TAB>Vorname<TAB>Name<TAB>Beruf<CR><LF>
```

ANFRAGEFENSTER:
Ist ein Wert für Anfragefenster angegeben, können folgende Reaktionen herbeigeführt werden:
1 Anfragefenster öffnet, wenn ein Anfrageergebnis vorliegt. Liegt kein Anfrageergebnis vor, passiert nichts weiter.
2 Anfragefenster öffnet, wenn ein Anfrageergebnis vorliegt. Liegt kein Anfrageergebnis vor, wird eine Nachricht angezeigt aus der hervorgeht, dass die Anfrage zu keinem Ergebnis führte

AUTOSTERN:
Legt das automatische Anhängen von Sternen an die Anfragewerte fest.
Mögliche Werte:
0 Einstellung des Autosterns wie im Client
1 Autostern aktiv
2 Autostern abschalten.
3 Autostern hinten
33 Autostern vorn
35 Autostern vorn und hinten

Beispiel

Anfragedatei für eine Anfrage an den-Schrank **Kunden**

```
[ANFRAGE]
SCHRANK=Kunde
KLAUSEL1=Kunde@Klasse=Kunde
KLAUSEL2=Kunde@Vorlage=10.03.1997
KLAUSEL3=Kunde@optimal_AS=1
DATENHEADER=1
DATENFELDER=1
```

Hinweis: Die Nummerierung der Klauseln muss fortlaufend sein. Existiert eine Lücke, dann werden die nachfolgenden Klauseln nicht berücksichtigt. Schrank- u. Feldbezeichner müssen exakt den Bezeichnern in der Objektdefinition entsprechen.

Es wird keine Überprüfung der Anfragedaten vorgenommen. Es kann also zu SQL-Fehlern führen, wenn z.B. ein Datum erwartet und ein Text angegeben wird.

StartDocRequest

Definition StartDocRequest (String strFileName)

Typ Methode

Beschreibung Startet eine Dokumentanfrage.

Parameter **strFileName** Name der Anfragedatei

Rückgabewert ***Dateiname*** oder **READY**, wenn ANFRAGEFENSTER <> 0
Leerer String, wenn ein Fehler aufgetreten ist.
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Anfragedatei muss folgenden Aufbau haben:

```
[ANFRAGE]
SCHRANK=
DOKUMENT=
KLAUSEL1=
KLAUSEL2=
...
...
KLAUSELn=
DATENFELDER=0(1)
DATENHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
VOLLTEXT=
```

Soll eine Registertrefferliste erzeugt werden, dann muss der Registername angegeben werden. Siehe auch 'StartRegRequest'

Die Rückgabedatei hat folgenden Aufbau:

```
INDEX1,OBJEKTTYP
INDEX2,OBJEKTTYP
...
...
INDEXn,OBJEKTTYP
```

Eine leere Datei ist ein gültiges Ergebnis.

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen

DATENFELDER:
Wird diese Option eingeschaltet ‚DATENFELDER=1', dann werden zusätzlich zu dem Index u. Objekttyp noch alle weiteren Daten des Objekts in die Datei geschrieben. Die Trennung der einzelnen Spalten erfolgt durch das Sonderzeichen <TAB>. Eine Zeile in der Ergebnistabelle sähe dann z.B. so aus:

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

DATENHEADER*:*
Wird diese Option eingeschaltet , DATENHEADER=1', dann wird in der ersten Zeile der Ergebnisdatei Spaltenüberschriften geschrieben. Die erste Zeile in der Ergebnistabelle sähe dann z.B. so aus:

```
OSID,OSTYPE<TAB>Vorname<TAB>Name<TAB>Beruf<CR><LF>
```

ANFRAGEFENSTER:
Ist ein Wert für Anfragefenster angegeben, können folgende Reaktionen herbeigeführt werden:
1 Anfragefenster öffnet wenn ein Anfrageergebnis vorliegt. Liegt kein Anfrageergebnis vor, passiert nichts weiter.
2 Anfragefenster öffnet wenn ein Anfrageergebnis vorliegt. Liegt kein Anfrageergebnis vor, wird eine Nachricht angezeigt aus der hervorgeht, dass die Anfrage zu keinem Ergebnis führte.

AUTOSTERN:
Legt das automatische Anhängen von Sternen an die Anfragewerte fest.
0 Einstellung des Autosterns wie im Client
1 Autostern aktiv
2 Autostern abschalten
3 Autostern hinten
33 Autostern vorn
35 Autostern vorn und hinten

TYP=2(0,1)
Legt fest, welche Art von Trefferliste erzeugt wird:
0 Ordnertrefferliste
1 Registertrefferliste
2 Dokumenttrefferliste
Ist der Typ nicht angegeben, wird eine Dokumenttrefferliste erzeugt.

LOKALESUCHE=0,1,2
Legt fest, ob Dokumente ohne Registerbezug in der Anfrage mit einbezogen werden sollen. Ist dieser Wert gleich 2, so wird die Einstellung des Benutzers im Client berücksichtigt.
Weiterhin kann in der as.cfg festgelegt werden, ob dieser Wert (die Einstellung im Client) automatisch in die Anfragedatei mit aufgenommen werden soll. Hierzu ist in der Sektion CLIENT folgender Eintrag zu setzen:

```
AUTOLOCALSEARCH=1
```

Beispiel

Anfragedatei für eine Anfrage an den Dokumententyp **Eingangsbeleg** aus dem Schrank **Kunde**

```
[ANFRAGE]
SCHRANK=Kunde
REGISTER=Register
DOKUMENT=Eingangsbeleg
KLAUSEL1=Kunde@Name=Müller
KLAUSEL2= Register@Typ=Auftrag
KLAUSEL3=Eingangsbeleg@Vorlage=10.03.1997
DATENHEADER=1
DATENFELDER=1
```

Hinweis: Die Nummerierung der Klauseln muss vorlaufend sein. Existiert eine Lücke, werden die nachfolgenden Klauseln nicht berücksichtigt.

Schrank- u. Feldbezeichner müssen exakt den Bezeichnern in der Objektdefinition entsprechen.

Es wird keine Überprüfung der Anfragedaten vorgenommen. Es kann also zu SQL-Fehlern führen, wenn z.B. ein Datum erwartet und ein Text angegeben wird.

Siehe auch: StartArchiveRequest

StartRegRequest

Definition StartRegRequest (String strFileName)

Typ Methode

Beschreibung Startet eine Registeranfrage.

Parameter **strFileName** Name der Anfragedatei

Rückgabewert **Dateiname** oder **READY**, wenn ANFRAGEFENSTER <> 0
Leerer String, wenn ein Fehler aufgetreten ist.
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Anfragedatei muss folgenden Aufbau haben:

```
[ANFRAGE]
SCHRANK=
REGISTER=
KLAUSEL1=
KLAUSEL2=
...
...
.KLAUSELn=
DATENFELDER=0(1)
DATENHEADER=0(1)
ANFRAGEFENSTER=0(1, 2)
AUTOSTERN=0(1,2)
VOLLTEXT=
```

Die Rückgabedatei hat folgenden Aufbau:

```
INDEX1,OBJEKTTYP
INDEX2,OBJEKTTYP
.
.
.
INDEXn,OBJEKTTYP
```

Eine leere Datei ist ein gültiges Ergebnis.

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen **DATENFELDER**:
Wird diese Option eingeschaltet ‚DATENFELDER=1', dann werden zusätzlich zu dem Index u. Objekttyp noch alle weiteren Daten des Objekts in die Datei geschrieben. Die Trennung der einzelnen Spalten erfolgt durch das Sonderzeichen <TAB>. Eine Zeile in der Ergebnistabelle sähe dann z.B. so aus:

```
1234,131071<TAB>Heiner<TAB>Lauterbach<TAB>Schauspieler<CR><L
F>
```

DATENHEADER:
Wird diese Option eingeschaltet , DATENHEADER=1', dann wird in der ersten Zeile der Ergebnisdatei Spaltenüberschriften geschrieben. Die erste Zeile in der Ergebnistabelle sähe dann z.B. so aus:

```
OSID,OSTYPE<TAB>Vorname<TAB>Name<TAB>Beruf<CR><LF>
```

ANFRAGEFENSTER:

Ist ein Wert für Anfragefenster angegeben, können folgende Reaktionen herbeigeführt werden:
1 Anfragefenster öffnet wenn ein Anfrageergebnis vorliegt. Liegt kein Anfrageergebnis vor, passiert nichts weiter.
2 Anfragefenster öffnet wenn ein Anfrageergebnis vorliegt. Liegt kein Anfrageergebnis vor, wird eine Nachricht angezeigt aus der hervorgeht, dass die Anfrage zu keinem Ergebnis führte

AUTOSTERN:
Legt das automatische Anhängen von Sternen an die Anfragewerte fest.
0 Einstellung des Autosterns wie im Client
1 Autostern aktiv
2 Autostern abschalten
3 Autostern hinten
33 Autostern vorn
35 Autostern vorn und hinten

Beispiel Anfragedatei für eine Anfrage an Kundenregister **Register**

```
[ANFRAGE]
SCHRANK=Kunde
REGISTER=Register
KLAUSEL1=Kunde@Name=Müller
KLAUSEL2=Register@Typ=Auftrag
KLAUSEL3=Kunde@optimal_AS=1
DATENHEADER=1
DATENFELDER=1
```


Hinweis: Die Nummerierung der Klauseln muss fortlaufend sein. Existiert eine Lücke, dann werden die nachfolgenden Klauseln nicht berücksichtigt. Schrank- u. Feldbezeichner müssen exakt den Bezeichnern in der Objektdefinition entsprechen.

Es wird keine Überprüfung der Anfragedaten vorgenommen. Es kann also zu SQL-Fehlern führen, wenn z.B. ein Datum erwartet und ein Text angegeben wird.

Siehe auch: StartArchiveRequest

StartClientRequest

Definition StartClientRequest (string request)

Typ Methode

Beschreibung Startet die formulierte Client-Anfrage.
Volltexttrefferlisten werden mit Facetten angezeigt.

Parameter

request	Anfrage im JSON-Format

Bemerkung Beschreibung JSON

request	Startobjekt
objecttype	UINT, Objekttyp der Anfrage
fulltext	String, Volltextanfragestring
fields	Array, JSON Elemente vom Typ field
field	besteht aus:
iname	String, interner Name des anzufragenden Feldes
value	String, Wert des anzufragenden Feldes

Rückgabewert

0	Anfrage ausgeführt
-125	Ein modaler Dialog ist geöffnet.
-140	JSON hat keinen Member 'request'
-141	Member 'request' hat keinen Member 'objecttype'
-143	Member 'objecttype' hat unerwarteten Datentyp
-144	Member 'objecttype' ist ungültig
-145	Angegebener Objekttyp ist unbekannt
-146	Angegebener Objekttyp ist nicht im Volltextindex
-147	Member 'fulltext' hat unerwarteten Datentyp
-148	Member 'fulltext' hat keinen Wert
-149	JSON enthält Parserfehler
-150	Member 'fields' ist kein Array
-151	Array-Member 'iname' oder 'value' fehlt
-152	Array-Member 'iname' oder 'value' hat keinen Wert
-153	Angegebenes Feld nicht gefunden

Der genaue Fehlertext kann mit **GetLastError()** ermittelt werden.

Beispiel

```
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)

Typ

Methode

Beschreibung

Speichert eine Notiz zu einem existierenden Objekt. (Ordner, Register, Dokument).

Parameter

lObjectID Objekt-ID
lObjectType Objekttyp
strFileName Dateiname der Notiz mit Endung 'txt'

Rückgabewert

0 kein Fehler
-1 kein Zielobjekt angegeben
-3 Schrank unbekannt
-7 Dokumenttyp unbekannt
-14 Ordnerkennung unbekannt
-15 Dokumentkennung unbekannt
-32 Notizdatei existiert nicht
-41 Dateiname leer
Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Die Notiz muss in Form einer ASCII-Textdatei vorliegen, die Endung 'txt' angegeben werden. Unabhängig vom Erfolg oder Misserfolg der Funktion wird die angegebene Datei vom Archivsystem nicht gelöscht. Wird sie nach Aufruf der Funktion nicht weiter benötigt, sollte sie vom aufrufenden Programm gelöscht werden.

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen. Wenn statt einer Datei eine Zeichenkette übergeben wird und die Notizen als Dateien über den Server abgelegt werden, erzeugt der Client die Datei selbst.

Optionen

Beispiel

TransformXML

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

Typ

Methode

Beschreibung

Wandelt eine XML-Datei mit Hilfe der XSL-Datei in ein gewünschtes Zielformat um.

Parameter

strXMLFile	vollständiger Pfad und Dateiname der XML-Datei
strXSLFile	vollständiger Pfad und Dateiname der XSL-Datei
varRetString	das transformierte Objekt wird hier in Form eines Strings zurückgegeben.

Rückgabewert

0	kein Fehler
-32	eine der beiden Quelldateien existiert nicht
-100	Dateien konnten nicht umgewandelt werden
-101	XML-Datei konnte nicht geladen werden
-102	XSL-Datei konnte nicht geladen werden
-103	XML-Objekt konnte nicht erstellt werden

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Optionen

Beispiel

UndoCheckOut

Definition	UndoCheckOut	(long long	lDocID, lDocType)
Typ	Methode		
Beschreibung	Macht einen Auscheckvorgang wieder rückgängig.		
Parameter	**lDocID** Dokumenten-ID **lDocType** Dokumenttyp		
Rückgabewert	**0** kein Fehler **-1** Undo fehlgeschlagen **-7** unbekannter Dokumenttyp **-53** Dokument wurde nicht ausgecheckt **-56** Name des Dokuments im Cache konnte nicht ermittelt werden Der Fehlertext kann mit **GetLastError()** ermittelt werden.		
Bemerkung	Nur für Dokumente, die der angemeldete Benutzer selbst ausgecheckt hat.		
Optionen			
Beispiel			

UnlinkDocuments

Definition

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

Typ

Methode

Beschreibung

Löscht eine Verknüpfung zwischen zwei Dokumenten.

Parameter

lObjectID1	ID des ersten Dokuments
lObjectType1	Typ des ersten Dokuments
lObjectID2	ID des zweiten Dokuments
lObjectType2	Type des zweiten Dokuments

Rückgabewert

0	kein Fehler
-29	eine oder beide Objekt-ID's sind unbekannt
-86	Objektindizes sind identisch
-88	Objekte aus der Ablage sind nicht zulässig

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Funktion steht ab 4.00 SPII zur Verfügung. Die Verknüpfung zwischen beiden Dokumenten wird in deren Notizen gespeichert.

Optionen

Beispiel

UpdateArchiveData

Definition

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

Typ

Methode

Beschreibung

Führt eine Aktualisierung von Ordnerdaten aus.

Parameter

strFilename	Übergabedatei (Aufbau siehe Bemerkung)
***lpReturnObjectID**	hier wird bei erfolgreichem Einfügen die Objekt-ID zurückgegeben
***lpReturnObjectType**	hier wird bei erfolgreichem Einfügen der Objekttyp zurückgegeben.

Rückgabewert

0	wenn kein Fehler
-3	Schrank unbekannt
-10	Ordnerkennung fehlt
-14	Ordnerkennung unbekannt
-18	Update fehlgeschlagen
-24	Feldnamen konnten nicht aufgelöst werden
-28	Feldwert für ein gegebenes Feld nicht zulässig
-30	ein oder mehrere Pflichtfelder nicht ausgefüllt
-31	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
-32	Übergabedatei existiert nicht
-47	Kein Schreibrecht auf dieses Objekt
-64	Serverfehler ist aufgetreten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

Die Übergabedatei hat folgenden Aufbau:

```
[AKTUALISIEREN]
SCHRANK=
SCHRANK-ID=
FELD1=
FELD2=
...
...
FELDn=
Mode=1
```

Ist der Mode = 1, werden alle nicht angegeben Felder nicht zurückgesetzt. Siehe Anmerkung „ACHTUNG“!

Und folgende Zeilen sind einzutragen, um Aktualisierungen für Tabellencontrols vorzunehmen, wobei Trennzeichen dem Zeichen chr(17) entspricht.

```
[Tabelle@TABELLENNAME]
Mode=0[1] 0 = Tabelle leeren, 1 = Tabelle anhängen
Zeile0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Zeile1={Spalte1Zeile2(Trennzeichen)Spalte2Zeile2}
```

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen

PFLICHTFELDER=[0;1]

Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

Beispiel Aktualisierung eines Kundenordners

```
[AKTUALISIEREN]
SCHRANK=Kunde
SCHRANK-ID=4711
FELD1=Klasse=Kunde
FELD2=Vorlage=10.03.1997
FELD3=optimal_AS=0
```

Hinweis: Bei allen Aktualisierungen sollten immer alle Felder angegeben werden. Die Felder, die nicht angegeben werden, haben nach erfolgreicher Aktualisierung keinen Inhalt mehr. Vorbelegte Felder werden nicht berücksichtigt. Es werden keine Schlüsselfelder überprüft!

UpdateArchiveDataS

Entspricht ' UpdateArchiveData' und kann aus VB-Script verwendet werden.

Definition

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

UpdateDocFileList

Definition UpdateDocFileList (String strFileName)

Typ Methode

Beschreibung Ermöglicht eine Änderung, der Dateiliste eines Dokuments.

Parameter **strFileName** Dateiname, Aufbau siehe Bemerkung

Rückgabewert

0	kein Fehler
-2	Keinen Schrank angegeben
-3	Schrank unbekannt
-6	Kein Dokumenttyp angegeben
-7	Dokumenttyp unbekannt
-9	Dokumenttyp gehört nicht zum angegebenen Schrank
-11	Dokumentkennung fehlt
-21	Dokument bereits archiviert
-32	Übergabedatei existiert nicht
-41	Kein Dateiname angegeben
-64	Serverfehler ist aufgetreten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Übergabedatei hat folgendes Format:

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

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Hinweis: Der Haupttyp des Zieldokuments muss mit **MAINTYPE=#** angegeben werden, wobei # für die Haupttyp-Nr. steht. Mögliche Haupttypen sind im Kapitel **Einführung** beschrieben.

Optionen

ARCHIVIERBAR=[0;1]
Hiermit kann der Archivstatus des Dokumentes geändert werden.
0 nicht archivierbar
1 archivierbar
1 ist Standard

DATEILÖSCHEN=[0;1]
Löscht die angegebenen Quell-Dateien.
0 ist Standard

Beispiel Codebeispiel:

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

Hinweis: Die angegebenen Dateinamen dürfen nicht identisch mit den Dateinamen sein, die dem Dokument bereits zugeordnet sind. Folge dieser Vorgehensweise ist sicherer Datenverlust.

Die Dateiliste kann nur geändert werden, wenn das Dokument noch nicht archiviert wurde!

UpdateDocShare

Definition

UpdateDocShare (long lIdent,
long lType,
long IUser)

Typ

Methode

Beschreibung

Bearbeitet die Freigabe des angegebenen Dokuments für den angegebenen Benutzer

Parameter

lIdent Dokument-ID
lType Dokumenttyp
Rights Rechte, die vorbelegt werden (RWXU)
IUser Benutzer-ID des Benutzers der Freigabe

Rückgabewert

0 "Freigabe bearbeitet oder Bearbeitung abgebrochen."
-125 "Ein modaler Dialog ist geöffnet."
-130 "Angegebener DMS-Objekttyp unbekannt."
-131 "Angegebener DMS-Objekttyp kein Dokument."
-132 "Nur Rechte 'RXWU' sind zulässig."
-133 "Angegebenes Dokument existiert nicht oder wurde gelöscht."
-134 "Dokumentfreigaben nicht möglich."
-135 "Die erforderliche Systemrolle um Dokumente freizugeben ist nicht vorhanden."
-136 "Angegebener Benutzer existiert nicht."
-139 "Angegebenes Objekt wurde von angemeldeten Benutzer nicht freigegeben."

Der genaue Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung

-

Beispiel

```
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 "Freigabe bearbeitet oder Bearbeitung abgebrochen."
end if

set Application = nothing
```

UpdateDocumentData

Definition UpdateDocumentData (String strFilename)

Typ Methode

Beschreibung Führt eine Aktualisierung von Dokumentdaten aus.

Parameter **strFilename** Übergabedatei (Aufbau siehe Bemerkung)

Rückgabewert

0	wenn kein Fehler
-2	kein Ordner angegeben
-3	Ordner unbekannt
-4	kein Register angegeben
-5	Register unbekannt
-7	Unbekannter Dokumenttyp
-8	Register gehört nicht zum angegebenen Ordner
-11	Dokumentkennung fehlt
-13	Registerkennung unbekannt
-16	Update fehlgeschlagen
-24	Feldnamen konnten nicht aufgelöst werden
-28	Feldwert für ein gegebenes Feld nicht zulässig
-30	ein oder mehrere Pflichtfelder nicht ausgefüllt
-31	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
-32	Übergabedatei existiert nicht
-47	Kein Schreibrecht auf dieses Objekt
-64	Serverfehler ist aufgetreten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Übergabedatei hat folgenden Aufbau:

```
[AKTUALISIEREN]
SCHRANK=
DOKUMENT=
DOKUMENT-ID=
FELD1=
FELD2=
     .
FELDn=
Mode=1
```

Ist Mode = 1, werden alle nicht angegeben Felder nicht zurückgesetzt. Siehe auch den **Hinweis** bei UpdateArchiveData.
Und folgende Zeilen sind einzutragen, um Aktualisierungen für Tabellencontrols vorzunehmen, wobei Trennzeichen dem Zeichen chr(17) entspricht.

```
[Tabelle@TABELLENNAME]
Mode=0[1] 0 = Tabelle leeren, 1 = Tabelle anhängen
Zeile0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Zeile1={Spalte1Zeile2(Trennzeichen)Spalte2Zeile2}
```

Beispiel:

```
[Tabelle@Protokoll]
Mode=1
Zeile0={01.02.2013(Trennzeichen)Mustermann}
Zeile1={04.02.2013(Trennzeichen)Müller}
Zeile2={05.02.2013(Trennzeichen)Meyer}
```

Bei Dokumenten, die der Dokumentenhistorie unterliegen, sollte der Haupttyp nicht geändert werden.

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

Optionen

PFLICHTFELDER=[0;1]
Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

REFRESHMULTIFIELDS=[0;1]
Ist der Wert 1, werden alle Einträge für die Mehrfachfelder gelöscht, für die neue Werte in den Sektionen [MULTI_...] definiert sind. Steht in der Übergabedatei der Wert REFRESHMULTIFIELDS=1 und es ist eine Sektion [MULTI_Feld1] definiert, so werden für dieses Feld alle Einträge gelöscht und die in der Sektion angegebenen Werte neu eingetragen.
Für das Aktualisieren von Mehrfachfeldern [MULTI_...], für ... steht der Datenbankname des Mehrfachfelds,

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

ARCHIVIERBAR = [0;1]
hiermit kann der Archivstatus des Dokumentes geändert werden.
0 nicht archivierbar
1 archivierbar

SHOWTEMPLATES=[0;1]
Für den Wert 1 wird für W-Dokumente der Vorlagendialog angezeigt. Die ausgewählte Vorlage wird eingefügt.

ENABLEOPTIONS=[0;1]
Für den Wert 1 werden die Optionen des Vorlagendialogs aktiviert.

Beispiel

Bsp.: Aktualisierung eines Kundendokuments **Eingangsbeleg**

```
[AKTUALISIEREN]
SCHRANK=Kunde
REGISTER=Register
DOKUMENT-ID=4713
FELD1=Typ=Brief
FELD2=Text=Hallo
[MULTI_FELD1]
DATA1=1,Peter
DATA2=2,Hans
```

UpdateRegisterData

Definition UpdateRegisterData (String strFilename)

Typ Methode

Beschreibung Führt eine Aktualisierung von Registerdaten aus.

Parameter **strFilename** Übergabedatei (Aufbau siehe Bemerkung)

Rückgabewert

0	wenn kein Fehler
-2	kein Ordner angegeben
-3	Ordner unbekannt
-4	kein Register angegeben
-5	Register unbekannt
-8	Register gehört nicht zum angegebenen Ordner
-12	Registerkennung fehlt
-13	Registerkennung unbekannt
-16	Update fehlgeschlagen
-24	Feldnamen konnten nicht aufgelöst werden
-28	Feldwert für ein gegebenes Feld nicht zulässig
-30	ein oder mehrere Pflichtfelder nicht ausgefüllt
-31	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
-32	Übergabedatei existiert nicht
-47	Kein Schreibrecht auf dieses Objekt
-64	Serverfehler ist aufgetreten

Der Fehlertext kann mit **GetLastError()** ermittelt werden.

Bemerkung Die Übergabedatei hat folgenden Aufbau:

```
[AKTUALISIEREN]
SCHRANK=
REGISTER=
REGISTER-ID=
FELD1=
FELD2=
     .
     .
FELDn=
Mode=1
```

Ist der Mode = 1, werden alle nicht angegeben Felder nicht zurückgesetzt. Siehe auch den Hinweis bei UpdateArchiveData

Und folgende Zeilen sind einzutragen, um Aktualisierungen für Tabellencontrols vorzunehmen, wobei Trennzeichen dem Zeichen chr(17) entspricht.

```
[Tabelle@TABELLENNAME]
Mode=0[1] 0 = Tabelle leeren, 1 = Tabelle anhängen
Zeile0={Spalte1Zeile1(Trennzeichen)Spalte2Zeile1}
Zeile1={Spalte1Zeile2(Trennzeichen)Spalte2Zeile2}
```

Der Parameter **strFileName** kann ein Dateiname oder eine Zeichenkette mit dem Inhalt der ansonsten übergebenen Datei sein. Wenn der Inhalt der Datei als Zeichenkette übergeben wird, muss am Ende jeder Zeile ein Zeilenumbruch stehen.

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

Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

Beispiel Aktualisierung eines Kundenregisters

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

COM-Schnittstelle der Vorschau-Fenster

Einleitung

Detail- und Inhalts-Vorschau des Clients können über eine COM-Schnittstelle angesprochen und gesteuert werden. Diese Schnittstelle kann entweder über die 'Application'-Schnittstelle erreicht werden oder aber auch direkt benutzt werden, solange dies innerhalb eines Events geschieht. Innerhalb von Events kann dieses Objekt unter dem Namen 'InfoWindow' angesprochen werden.

Alle COM-Befehle

Caption

Definition	String Caption
Typ	Eigenschaft (lesen/schreiben)
Beschreibung	Setzt den Titel des Fensters oder gibt ihn zurück
Parameter	-
Rückgabewert	-
Beispiel	

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

Visible

Definition	Boolean Visible
Typ	Eigenschaft (lesen/schreiben)
Beschreibung	Legt die Sichtbarkeit des Fensters fest
Parameter	-
Rückgabewert	-
Beispiel	

```
Dim a as object
Set a = createobject("optimal_as.application")
a.InfoWindow.Visible = false 'macht das Fenster unsichtbar
```

URL

Definition	String URL
Typ	Eigenschaft (lesen/schreiben)
Beschreibung	Legt den URL fest, der angezeigt werden soll
Parameter	-
Rückgabewert	-
Beispiel	

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

Closeable

Definition	Boolean Closeable
Typ	Eigenschaft (lesen/schreiben)
Beschreibung	Legt fest, gibt an, ob das Fenster vom Benutzer geschlossen werden kann.
Parameter	-
Rückgabewert	-
Beispiel	

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

HtmlDocument

Definition	Object HtmlDocument
Typ	Eigenschaft (lesen)
Beschreibung	Liefert das HTML-DOM-Document, welches gerade angezeigt wird.
Parameter	-
Rückgabewert	-
Bemerkung	Die HTML-Document Schnittstelle ist in der MSDN beschrieben.
Beispiel	

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

EnableContextMenu

Definition	Boolean EnableContextMenu
Typ	Eigenschaft (lesen/schreiben)
Beschreibung	Gibt an, legt fest, ob das Kontextmenü angezeigt werden kann
Parameter	-
Rückgabewert	-
Beispiel	

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

Refresh

Definition	Refresh()
Typ	Methode
Beschreibung	Aktualisiert die Anzeige im Fenster
Parameter	-
Rückgabewert	-
Beispiel	

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

GoBack

Definition	GoBack()
Typ	Methode
Beschreibung	Navigiert zur vorherigen URL.
Parameter	-
Rückgabewert	-
Beispiel	

```
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
```

GoForward

Definition	GoForward()
Typ	Methode
Beschreibung	Navigiert in der Historie zur folgenden URL.
Parameter	-
Rückgabewert	-
Beispiel	

```
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
```

ShowHtml

Definition	ShowHtml(String strHtml)
Typ	Methode
Beschreibung	Zeigt den übergebenen Html-String an.
Parameter	-
Rückgabewert	-
Beispiel	

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

Skript-Schnittstelle der Vorschau-Fenster

Einleitung

Mit dieser Schnittstelle werden Ihnen Funktionen zur Verfügung gestellt, mit denen Sie aus Detail- und Inhalts-Vorschau-Fenstern heraus den Client ansteuern können, beispielsweise um eine Trefferliste zu aktualisieren oder ein Datenblatt zu öffnen. Diese Funktionen können in mehreren Durchläufen genutzt werden.

Funktionen zum Blättern über Dokumentgrenzen hinweg

osjxCanNextDoc

Definition	Boolean osjxCanNextDoc	
Typ	Eigenschaft (lesen/schreiben)	
Beschreibung	Zeigt an, ob es in der aktuellen Trefferliste ein nächstes Dokument gibt.	
Parameter	-	
Rückgabewert	**1**	Es gibt ein nächstes Dokument in der aktuellen Trefferliste.
	0	Es gibt kein nächstes Dokument in der aktuellen Trefferliste.

osjxCanPrevDoc

Definition	Boolean osjxCanPrevDoc
Typ	Eigenschaft (lesen/schreiben)
Beschreibung	Zeigt an, ob es in der aktuellen Trefferliste ein vorangehendes Dokument gibt.
Parameter	-
Rückgabewert	**1** Es gibt ein vorangehendes Dokument in der aktuellen Trefferliste. **0** Es gibt kein vorangehendes Dokument in der aktuellen Trefferliste.

osjxNextDoc

Definition	osjxNextDoc()
Typ	Methode
Beschreibung	Setzt den Fokus auf das nächste Dokument.
Parameter	-
Rückgabewert	-

osjxPrevDoc

Definition	osjxPrevDoc()
Typ	Methode
Beschreibung	Setzt den Fokus auf das vorangehende Dokument.
Parameter	-
Rückgabewert	-

Funktionen zur Fenstersteuerung des Clients

osjxOpenDataSheet

Definition	osjxOpenDataSheet	(long BOOL	lObjectID, bReadOnly)
Typ	Methode		
Beschreibung	Zeigt das Datenblatt eines angegebenen Objekts an.		
Parameter	**lObjectID** Objekt-ID **bReadOnly** **TRUE**, wenn das Datenblatt schreibgeschützt angezeigt werden soll.		
Rückgabewert	-		
Bemerkung	-		

osjxOpenObject

Definition	osjxOpenObject	(long lObjectID)
Typ	Methode	
Beschreibung	Öffnet das Dokument oder den Ordner eines Objekts.	
Parameter	**lObjectID** Objekt-ID	

Rückgabewert	-
Bemerkung	-

osjxOpenLocation

Definition	osjxOpenLocation (long lObjectID)
Typ	Methode
Beschreibung	Öffnet den Standort eines Objekts.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-
Bemerkung	-

osjxOpenLocationsAndLinks

Definition	osjxOpenLocationsAndLinks (long lObjectID)
Typ	Methode
Beschreibung	Öffnet Links und Verknüpfungen eines Objekts.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-

osjxOpenObjectHistory

Definition	osjxOpenObjectHistory (long lObjectID)
Typ	Methode
Beschreibung	Öffnet die Bearbeitungshistorie eines Objekts.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-

osjxAddSignature

Definition	osjxAddSignature (long lObjectID)
Typ	Methode
Beschreibung	Fügt eine elektronische Signatur hinzu.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-

osjxPrintObject

Definition	osjxPrintObject()
Typ	Methode
Beschreibung	Druckt das Dokument aus.
Parameter	-
Rückgabewert	-

osjxOpenObjectRemarks

Definition	osjxOpenObjectRemarks (long lObjectID)
Typ	Methode
Beschreibung	Öffnet die Notizen eines Objekts.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-

osjxAddFollowUp

Definition	osjxAddFollowUp (long lObjectID)
Typ	Methode
Beschreibung	Fügt eine Wiedervorlage hinzu.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-

osjxAddSubscribe

Definition	osjxAddSubscribe (long lObjectID)
Typ	Methode
Beschreibung	Fügt ein Abonnement hinzu.
Parameter	**lObjectID** Objekt-ID
Rückgabewert	-

osjxStartWorkflow

Definition	osjxStartWorkflow	(long String	lObjectID, strWorkflowModell)
Typ	Methode		
Beschreibung	Startet mit dem Objekt einen Workflowprozess anhand des Modellnamens.		
Parameter	**lObjectID** **strWorkflowModell**	Objekt-ID Name des Workflow-Modells	
Rückgabewert	-		

osjxGetSelectedObjects

Definition	osjxGetSelectedObjects()
Typ	Methode
Beschreibung	Ermittelt ausgewählte Objekte aus der Trefferliste.
Parameter	-
Rückgabewert	Liste der ausgewählten Objekt in folgender Form: `Obj0-ID,Obj0-Type;…;Obj(n)-ID,Obj(n)-Type`

osjxRefreshObjectInLists

Definition	osjxRefreshObjectInLists	(long	lObjectID)
Typ	Methode		
Beschreibung	Aktualisiert das angegebene DMS-Objekt in allen offenen Listen.		
Parameter	**lObjectID**	Objekt-ID	
Rückgabewert	-		
Bemerkung	-		

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

osjxByteArrayToFile

Definition	osjxByteArrayToFile (JaveScriptArray, sting sDateiname)
Typ	Methode
Beschreibung	Nimmt ein JavaScript-Array mit Integer-Werten entgegen und schreibt diese in eine Datei.
Parameter	**JavaScript-Array** **sDateiname**
Rückgabewert	-
Bemerkung	Ist der Dateiname ohne eindeutigen Pfad angegeben, wird ein 'Speichern unter'-Dialog' geöffnet.
Beispiel	

```
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");
}
```


osjxURLDownloadToFile

Definition	osjxURLDownloadToFile (string sDateiUrl, sting sDateiname)
Typ	Methode
Beschreibung	Nimmt eine Datei-URL entgegen und schreibt den Inhalt der Datei-URL in eine Datei.
Parameter	**sDateiUrl** **sDateiname**
Rückgabewert	-
Bemerkung	Ist der Dateiname ohne eindeutigen Pfad angegeben, wird ein 'Speichern unter'-Dialog' geöffnet.
Beispiel	

```
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)
Typ	Methode
Beschreibung	Öffnet eine Trefferliste mit dem angegeben Titel und den angegeben DMS-Objekten.
Parameter	**sTitel** **JSON-String**
Rückgabewert	-

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

Funktionen zur Dashletsteuerung

Dashlets können auf drei Arten identifiziert werden:

1. Keinen Parameter: Das aktuelle Dashlet
2. Dashlet-Titel: Beim Einrichten des Dashlets im enaio® enterprise-manager angegebener Titel. Auf die systemseitigen Dashlets 'DocumentViewer' oder 'DetailsViewer' wird über die Titel 'OSDOCUMENTVIEWER' bzw. 'OSDETAILVIEWER' zugegriffen.
3. Nummer des Dashlet (1 – 10)

Dieser Funktionsparameter ist im Folgenden als 'dashletID' gekennzeichnet.

osjxOpenChromeDEVTools

Definition	osjxOpenChromeDEVTools()
Typ	Methode

Beschreibung	Öffnet die DEV-Tools des Chrome-Browsers.
Parameter	-
Rückgabewert	-

osjxCloseChromeDEVTools

Definition	osjxCloseChromeDEVTools()
Typ	Methode
Beschreibung	Sind die DEV-Tools des Chrome-Browsers offen, kann man sie hiermit schließen.
Parameter	-
Rückgabewert	-

osjxIsDashletVisible

Definition	osjxIsDashletVisible (dashletID)
Typ	Methode
Beschreibung	Ermittelt, ob das angegebene Dashlet sichtbar ist.
Parameter	**dashletID** (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.
Rückgabewert	**1** Dashlet ist sichtbar **0** Dashlet ist nicht sichtbar

osjxSetDashletVisible

Definition	osjxSetDashletVisible (dashletID, bool bVisible)
Typ	Methode
Beschreibung	Setzt das angegebene Dashlet sichtbar bzw. nicht sichtbar. Ist es nicht sichtbar, ist es auch aus enaio® client über das Menüband 'Ansicht' nicht mehr erreichbar.
Parameter	**dashletID** (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen. **bVisible** 0 = nicht sichtbar / 1 = sichtbar
Rückgabewert	-

osjxIsDashletEnabled

Definition	osjxIsDashletEnabled (dashletID)
Typ	Methode
Beschreibung	Ermittelt, ob das angegebene Dashlet aktiv ist.
Parameter	**dashletID** (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.
Rückgabewert	**1** Dashlet ist aktiv **0** Dashlet ist nicht aktiv

osjxSetDashletEnabled

Definition	osjxSetDashletEnabled (dashletID, bool bEnabled)
Typ	Methode
Beschreibung	Setzt das angegebene Dashlet aktiv oder inaktiv Ist es inaktiv, werden 'ContextChanges' ignoriert.
Parameter	**dashletID** (siehe oben) Dashlet-Namen als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen. **bEnabled** 0 = inaktiv / 1 =aktiv
Rückgabewert	-

osjxGetDashletURL

Definition	osjxGetDashletURL (dashletID)
Typ	Methode
Beschreibung	Ermittelt die URL des angegebenen Dashlets.
Parameter	**dashletID** (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.
Rückgabewert	URL des Dashlets als String

osjxSetDashletURL

Definition	osjxSetDashletURL (dashletID, string sUrl)
Typ	Methode
Beschreibung	Setzt die URL des angegebenen Dashlets. Wird keine URL angegeben, wird die ursprünglich konfigurierte URL gesetzt.

Parameter	**dashletID** (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen. **sUrl** URL des Dashlets als String
Rückgabewert	-

osjxGetDashletPaneState

Definition	osjxGetDashletPaneState (dashletID)
Typ	Methode
Beschreibung	Ermittelt den Anzeigestatus des Bereichs des angegebenen Dashlets.
Parameter	**dashletID** (siehe oben) Dashlet-Namen als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.
Rückgabewert	0: unknown 1: closed 2: hidden 3: floating

osjxSetDashletPaneState

Definition	osjxSetDashletPaneState (dashletID, long lStatus)
Typ	Methode
Beschreibung	Setzt den Anzeigestatus des Bereichs des angegebenen Dashlets. Ist das angegebene Dashlet mit 'osjxSetDashletVisible' auf nicht sichtbar gesetzt, wird der Aufruf ignoriert.
Parameter	**dashletID** (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen. **lStatus**: 0: unknown 1: closed 2: hidden 3: floating
Rückgabewert	-

osjxSetDashletCaption

Definition	osjxSetDashletCaption (dashletID, string sTitle)
Typ	Methode
Beschreibung	Legt einen Titel für ein Dashlet fest.

Parameter **dashletID** (siehe oben)
Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.
sTitel

Rückgabewert -

Beispiel

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

osjxGetDashletCaption

Definition osjxGetDashletCaption (dashletID)

Typ Methode

Beschreibung Ermittelt den Titel eines Dashlets.

Parameter **dashletID** (siehe oben)
Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.

Rückgabewert Titel des Dashlets.

Beispiel

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

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

osjxGetEnvironment

Ermittelt die aus der COM-Funktion 'GetEnvironment' bekannten Werte.

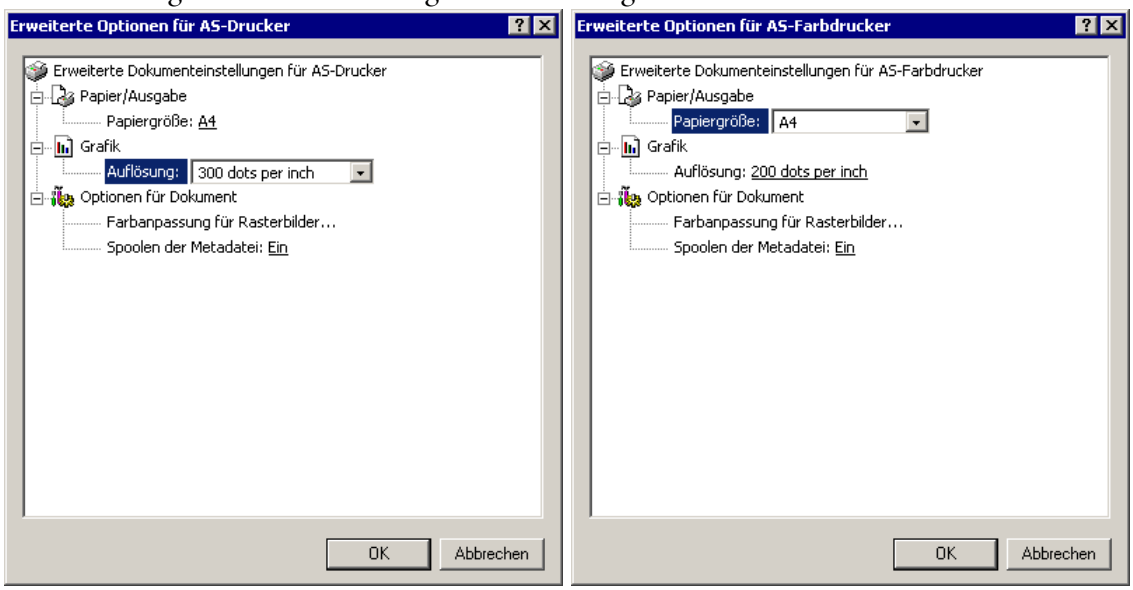
Anhang: Konfiguration der enaio®-Drucker

Einleitung

Bei der Installation von enaio®client können am Arbeitsplatz optional zwei Druckertreiber mit installiert werden:

- der OS-Drucker `axprint.dll` für den Schwarz-/Weißdruck
- der OS-Farbdrucker `axcprint.dll` für den Farbdruck

Haben Sie diese Druckertreiber nicht installiert, erhalten Sie auf Anfrage ein Setup für die Druckertreiber.
Über diese Druckertreiber können aus jeder Anwendung mit Druckfunktion Bilddateien erzeugt und an den enaio® client übergeben werden. Der enaio® client öffnet die entsprechenden Dialoge, über die der Benutzer einen Standort angibt, den Dokumenttyp wählt und das Dokument verschlagwortet. Vorausgesetzt ist, dass der enaio® client am Arbeitsplatz läuft. Läuft er nicht, wird ein entsprechender Hinweis angezeigt. Der Benutzer kann dann den enaio® client starten und den Vorgang fortsetzten oder das Drucken abbrechen.
Die Druckertreiber können ebenfalls verwendet werden, um die Druckdaten als Bild- oder PDF-Dateien im Dateisystem zu speichern. Dazu wird eine Konfigurationsdatei `asprint.ini` benötigt. Diese Konfigurationsdatei muss von der entsprechenden Anwendung erzeugt und im Windows-Verzeichnis gespeichert werden. Nach dem Druck muss in der Regel die Konfigurationsdatei wieder gelöscht werden.
Die Konfiguration der Papiergröße und der Auflösung erfolgt nicht über die Konfigurationsdatei, sondern wie gewohnt über die Eigenschaftsdialoge der Druckertreiber.



Hinweis: Der OS-Farbdrucker darf nicht auf Graustufen-Druck eingestellt werden. Diese Einstellung führt zu Fehlern.

Die Konfigurationsdatei

Die Druckertreiber suchen vor dem Erzeugen der Dateien nach der Konfigurationsdatei `asprint.ini` im Windows-Verzeichnis.
Nur wenn die Datei nicht existiert, wird versucht, die Bilddateien an den enaio® client zu übergeben. Wird die Datei gefunden, werden die Konfigurationsdaten ausgelesen.
Die Datei erstellen Sie mit einem beliebigen Editor. Sie muss die Bezeichnung `asprint.ini` tragen und im Windows-Verzeichnis gespeichert werden.
In der ersten Zeile der Datei geben Sie die Sektionsbezeichnung `[ASPRINT]` an. Darauf folgend die Parameter:

Parameter	Mögliche Werte	Beschreibung
DATEI=	<Pfad\Dateiname>.000	Sie geben Pfad, Dateinamen und Endung an. Die Druckertreiber verwenden immer eine automatisch hexadezimal hochgezählte Endung, wenn mehrere Seiten als einzelne Dateien erzeugt werden. Die erste Seite hat die Endung '000'.
	<Pfad\Dateiname>.tif	Geben Sie als Endung 'tif' an, verwendet der AS-Drucker diese Endung, wenn alle Seiten als eine Multipage-TIF-G4-Datei gespeichert werden. Bei Verwendung von MULTIPAGE=1 wird für jeden Druckauftrag EINE Datei angelegt. Sonst automatisch Endung .000
	<Pfad\Dateiname>.pdf	Geben Sie als Endung 'pdf' an, verwenden OS-Drucker und OS-Farbdrucker diese Endung, wenn alle Seiten als eine PDF-Datei gespeichert werden.
HINTERGRUND1=	<Pfad\Dateiname>.tif	Der OS-Drucker kann Hintergrundbilder einbinden. Die Dateien müssen im TIFF-G4 - Format vorliegen und sollten dieselbe Auflösung besitzen, wie die für den OS-Drucker eingestellte. Dieser Parameter gibt das Hintergrundbild für die erste Seite an.
HINTERGRUND2=	<Dateiname>.tif	Dieser Parameter gibt das Hintergrundbild für alle Folgeseite an.
MULTIFILE=	0	Existieren bereits Dateien mit gleicher Bezeichnung, werden diese nicht überschrieben. Der Druckauftrag wird abgebrochen. In die Protokolldatei wird die Ergebnisnummer '-8 - Datei existiert bereits' eingetragen.
	1	Abhängig vom Parameter 'Multipage' werden Dateien mit gleicher Bezeichnung nicht überschrieben, sondern die Endung wird hochgezählt oder die Datei erweitert.
MULTIPAGE=	0	Für jede Seite wird eine Datei erzeugt. Die Endung wird hexadezimal hochgezählt und beginnt mit '000'. Beim Parameter 'MULTIFILE=1' ermitteln die Druckertreiber, ob bereits Dateien mit gleicher Bezeichnung vorliegen und setzen gegebenenfalls die Nummerierung der Endung fort. Beim Parameter 'MULTIFILE=0' wird der Druckauftrag abgebrochen, wenn bereits Dateien mit gleicher Bezeichnung existieren.

Parameter	Mögliche Werte	Beschreibung
	1	Der OS-Drucker erzeugt eine Multipage-TIF-G4-Datei. Besteht diese Datei bereits, werden die neuen Seiten angehängt. Hat der Parameter 'PDF' den Wert '1', wird die Datei mit einem PDF-Header erzeugt und kann in einem PDF-Viewer geöffnet werden. Der OS-Farbdrucker erzeugt eine PDF-Datei mit allen Seiten, falls der Parameter 'PDF' den Wert '1' hat. Besteht diese Datei bereits, werden die neuen Seiten angehängt. Hat der Parameter 'PDF' den Wert '0', wird für jede Seite eine Datei im Format 'JPEG' erzeugt. Die Endung wird hexadezimal hochgezählt und beginnt mit '000'.
PDF=	0	Der OS-Drucker erzeugt Bilddateien im TIFF-G4 – Format. Der OS-Farbdrucker erzeugt Bilddateien im JPEG-Format. Die Default-Einstellung ist '0'.
	1	Die erzeugten Bilddateien werden mit einem PDF-Header versehen und können in einem PDF-Viewer geöffnet werden.
GRAU=	0	Default-Einstellung für den OS-Farbdrucker, erzeugt werden Farbbilder.
	1	Der OS-Farbdrucker erzeugt Graustufenbilder im JPEG-Format.
KOMPRESSION=	1-100	Für den OS-Farbdrucker kann eine Kompressionsstufe angegeben werden. Der Wert '100' führt zu maximaler Kompression. Die Default-Einstellung ist '1' - minimalste Kompression.
EXE=	<Pfad\Programm>	Pfad und Bezeichnung eines ausführbaren Programms, das nach dem Druckauftrag gestartet werden soll. Dem Programm werden beim Aufruf folgende Parameter in Anführungszeichen übergeben: ▪ Pfad und Bezeichnung der Konfigurationsdatei `asprint.ini` ▪ Protokolleinträge (vgl. Protokollierung) Zusätzlich wird in die Datei `asprint.ini` der Benutzername eingetragen: USERNAME='Windows-Benutzername'
CITRIXMODE=	1	Diesen Eintrag benötigen Sie nur für eine Terminalserver-Installation (vgl. Terminalserver)

Beispiel:

```
[ASPRINT]
DATEI=C:\ASDRUCK\print.tif
HINTERGRUND1=C:\ASDDRUCK\HG1.tif
HINTERGRUND2C:\ASDDRUCK\HGff.tif
MULTIFILE=1
MULTIPAGE=1
PDF=0
```

Der OS-Drucker erzeugt die Datei `print.tif`. Das Format ist 'Multipage TIFF G4'. Bei folgenden Druckaufträgen wird die Datei erweitert.

Protokollierung

Die enaio®-Druckertreiber erzeugen eine LOG-Datei mit einer Ergebnisnummer und einer Beschreibung. Die LOG-Datei trägt die gleiche Bezeichnung wie die Bilddatei und erhält die Endung 'LOG'. Sie kann mit einem beliebigen Editor geöffnet werden.

Nummer	Bedeutung
1	"Seiten wurden erfolgreich erzeugt
-1	"Fehlerhafte bzw. ungültige Parameter."
-2	"Nicht genügend Speicher verfügbar."
-3	"Fehler beim Sperren eines Speicherbereichs."
-4	"Fehler beim Erzeugen einer Datei."
-5	"Datei existiert nicht bzw. konnte nicht geöffnet werden."
-6	"Fehler beim Lesen einer Datei."
-7	"Fehler beim Schreiben einer Datei."
-8	"Datei existiert bereits."
-9	"Fehlerhaftes bzw. nicht unterstütztes Datei-Format."
-10	"Fehlerhaftes bzw. nicht unterstütztes TIFF-Format."
-23	"Fehler beim Erzeugen eines Verzeichnisses."
-24	"Ungültiges Ziel-Verzeichnis."
-25	"Ungültiges Quell-Verzeichnis."
-26	"Ungültiges temporäres Verzeichnis."
-27	"Ungültiges Laufwerk."
-28	"Nicht genügend Plattenspeicher verfügbar."
-29	"Fehler beim Wechseln eines Verzeichnisses."
-30	"Fehler beim Löschen eines Verzeichnisses."
-31	"Fehler beim Bestimmen der Dateigrößen eines Verzeichnisses."
-44	"Fehler beim Laden einer DIB."
-45	"Fehlerhafte bzw. nicht vorhandene DIB."
-46	"Fehler beim Erzeugen einer Bitmap."
-73	"Kein Dateiname für Hintergrund-Bitmap vorhanden"
-74	"Datei enthält keine Bilddaten"
-78	"Unbekannter Fehler."
-79	"Datei existiert nicht."
-80	"Quelldatei konnte nicht geöffnet werden."
-81	"Zieldatei konnte nicht geöffnet werden."
-82	"Quelldatei existiert nicht."
-83	"Zieldatei existiert nicht."
-85	"Fehler beim Erzeugen eines Dias."
-91	"Aktion vom Benutzer abgebrochen."

Beispiel:

```
-8: Datei C:\ASDRUCK\print.000 existiert bereits
```

Starten Sie nach dem Druck eine Anwendung, wird dieser Inhalt als Aufrufparameter der Anwendung zusammen mit Pfad und Bezeichnung der Konfigurationsdatei `asprint.ini` übergeben.

Terminalserver

Bei Terminalserverinstallationen benötigen beide Druckertreiber die Konfigurationsdatei `asprint.ini`.
Den Pfad zur Konfigurationsdatei geben Sie über folgenden Registryeintrag an:

`HKEY_LOCAL_MASCHINE\SOFTWARE\Optimal Systems`

Dort erzeugen Sie den Schlüssel 'asprinter'. Als Zeichenfolge geben Sie 'directory' an, als Wert ein Verzeichnis, welches für alle Benutzer zugänglich ist.
In dieses Verzeichnis legen Sie die Konfigurationsdatei `asprint.ini`. Der Parameter 'CITRIXMODE=1' ist notwendig.
Sie geben über den Parameter 'Datei' eine beliebige Dateibezeichnung an. Pfad und Dateibezeichnung werden nicht ausgewertet, wohl aber gegebenenfalls die Endung.
Die Druckertreiber speichern die Dateien immer nach folgendem Schema:

- Unterhalb des Verzeichnisses mit der Konfigurationsdatei wird für jeden Benutzer, der druckt, ein Unterverzeichnis angelegt, in dem die Dateien gespeichert werden.
- Das Unterverzeichnis trägt als Bezeichnung den Windows-Benutzernamen.
- Die Dateien tragen als Bezeichnung den Windows-Benutzernamen. Die Endung ist von den Parametern abhängig.

Die anderen Parameter haben bei Terminalserverinstallationen die gleichen Funktionen. Erzeugt wird ebenfalls eine LOG-Datei.

Anhang: Strukturbaumkataloge

Datenkonvertierung

Die Daten von Strukturbaumkatalogen werden über Strukturbaumdateien verwaltet. Diese Dateien werden über den enaio® editor erzeugt.
Mit der Version 5.5 steht über die Bibliothek `oxlist.dll` eine API-Schnittstelle zur Verfügung, über die ASCII-Textdateien in Strukturbaumdateien und auch umgekehrt Strukturbaumdateien in ASCII-Textdateien konvertiert werden können.
Damit ist es möglich, Daten für den Strukturbaumkatalog dynamisch zu erzeugen, zu aktualisieren und abzugleichen.
Die Bibliothek `oxlist.dll` wird zusammen mit der ebenfalls notwendigen Bibliothek `oxmisc.dll` in die Installationsverzeichnisse `clients\client32` und `clients\admin` installiert.
Die Schnittstelle enthält die folgenden beiden Funktionen, die beispielsweise aus VisualBasic ansprechbar sind:

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

/*!
  Die Funktion generiert aus einer ASCII-Datei und einer zugehörigen Ebenendefiniton eine
Strukturbaumdatei.

  @param LPCSTR lpAsciifile   - ASCII-Datei die sortiert die Kürzel und zugehörigen
Bezeichnungen enthält.  (Kürzel und Bezeichnungen müssen durch Leerzeichen getrennt sein,
nicht durch Tabulatoren)
  @param LPCSTR lpDefinition  - Ebenendefinition z.Bsp. AA@A.9.AA
(A -> Buchstaben,Zahlen 9 -> nur Zahlen, @ -> kein Trennzeichen, . -> Trennzeichen)

  @param LPCSTR lpTreefile    - Dateiname für eine zu erzeugende Baumstrukturdatei.

  @return int                         - 0 -> bei Erfolg, sonst -> Fehlercode
*/

int WINAPI ConvertTreeFileToAsciiFile(LPCSTR lpTreefile, LPCSTR lpAsciifile);

/*!
  Die Funktion generiert aus einer Baumstrukturdatei eine ASCII-Datei.

  @param LPCSTR lpTreefile    - Dateiname einer existierenden Baumstrukturdatei.
  @param LPCSTR lpAsciifile   - Zu erzeugende ASCII-Datei die sortiert die Kürzel und
zugehörigen Bezeichnungen enthält.

  @return int                         - 0 -> bei Erfolg, sonst -> Fehlercode
*/
```

Beispiel zur Benutzung dieser Funktionen aus VisualBasic:

```
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
```

Struktur der Textdatei

Beim Erzeugen einer Strukturbaumdatei im enaio® editor geben Sie ein Ebenendefinition an. Beim Konvertieren einer ASCII-Textdatei in eine Strukturbaumdatei geben Sie ebenfalls diese Ebenendefinition mit an.
Innerhalb der ASCII-Textdatei ordnen Sie zeilenweise Kürzel und Eintrag einander zu.
Dabei muss das Kürzel der Ebenendefinition entsprechen, sowohl bezüglich der Anzahl der Stellen wie auch bezüglich der Position der Trennzeichen. Welches Trennzeichen verwendet wird, ist für die Syntax in der ASCII-Textdatei gleichgültig, nur die Position ist wichtig. Im Beispiel unten sind statt der Trennzeichen '/' und '-' einfach Leerzeichen eingesetzt.
Verwenden Sie zwischen Ebenen kein Trennzeichen, also das '@' in der Ebenendefinition, setzten Sie auch kein Trennzeichen zwischen die Ebenenkürzel.
Kürzel und Eintrag trennen Sie durch ein Leerzeichen.
Für alle Kürzel bis auf die Kürzel für die letzte Ebene ergänzen Sie immer Leerzeichen, bis die Anzahl der Stellen gemäß der Ebenendefinition erreicht ist.

Beispiel:

```
Ebenendefinition: 99/99-A

Dateiinhalt:
01      2001
01 01   Januar
01 01 F Familienrecht
01 01 U Urheberrecht
01 01 V Vertragsrecht
01 02   Februar
01 02 F Familienrecht
01 02 U Urheberrecht
01 02 V Vertragsrecht
01 03   März
01 03 F Familienrecht
01 03 U Urheberrecht
01 03 V Vertragsrecht
02      2002
02 01   Januar
```


Erläuterung:
Die Anzahl der Stellen laut Ebenendefinition ist sieben, jedes Kürzel ist also siebenstellig. Kürzel der ersten und zweiten Ebene werden entsprechend mit Leerstellen ergänzt. Die achte Stelle ist ein Leerzeichen und trennt Kürzel und Eintrag. Ein Tabulatorzeichen ist nicht erlaubt.
Die dritte Stelle ist ein Trennzeichen, in der ASCII-Textdatei kann ein beliebiges Trennzeichen eingegeben werden. Gleiches gilt für die sechste Stelle.
Die Zuordnungen müssen hierarchisch aufeinander folgen. Eine Zuordnung für die zweite Ebene muss in der Datei also der entsprechenden Zuordnung der ersten Ebene folgen.