

# Softwaredokumentation

## enaio® client – Programmierreferenz

Version 11.10

Sämtliche Softwareprodukte sowie alle Zusatzprogramme und Funktionen sind eingetragene und/oder in Gebrauch befindliche Marken der OPTIMAL SYSTEMS GmbH, Berlin oder einer ihrer Gesellschaften. Sie dürfen nur mit gültigem Lizenzvertrag benutzt werden. Die Software sowie die jeweils zugehörige Dokumentation sind nach deutschem und internationalem Recht urheberrechtlich geschützt. Das illegale Kopieren und Vertreiben der Software stellt Diebstahl geistigen Eigentums dar und wird strafrechtlich verfolgt. Alle Rechte vorbehalten, einschließlich der Wiedergabe, Übermittlung, Übersetzung sowie Speicherung mit/auf Medien aller Art. Für vorkonfigurierte Testszenarien oder Demo-Präsentationen gilt: Alle Firmennamen und Personen, die in Beispielen (Screenshots) erscheinen, sind frei erfunden. Eventuelle Ähnlichkeiten mit tatsächlich existierenden Firmen und Personen sind zufällig und unbeabsichtigt.

Copyright 1992 – 2024 by      OPTIMAL SYSTEMS GmbH  
Cicerostraße 26  
D-10709 Berlin

18.07.2024  
Version 11.10

# Inhalt

Einführung.....	6
Übergabedateien.....	6
Interne Namen .....	6
Klauseln und Ausdrücke.....	7
Klauseln .....	7
Ausdrücke.....	7
Platzhalter in Anfragedateien .....	10
Code-Beispiele.....	11
VB und VBA.....	11
Objekt-Haupttypen.....	12
Alle COM-Befehle .....	13
ActivateApp .....	13
AdjustRetention .....	14
AppBrowserSession.....	15
ApplicationLogin.....	16
ArchiveMailFile .....	17
CalcFileDigest .....	20
CheckInDocument.....	21
CheckLicence.....	22
CheckObjectAccess .....	23
CheckOutDocument.....	24
ClearSignatureProperties.....	25
CloseAllWindows .....	26
CloseBrowser .....	27
CloseObjectID .....	28
ConvertImage.....	29
CopyObject.....	30
CreateDocumentLink .....	31
CreateMimeFile.....	32
DecodeIMAPFile .....	33
CreateNewDocShare .....	34
DeleteFromArchive.....	35
DeleteResourceString.....	36
DoPrefetch.....	37
EMailImport.....	38
ExecuteRequest.....	40
ExecuteRequestEx .....	41
ExistBrowser .....	42
FindObjectType.....	43
FindObjectTypeEx .....	44
FreeDocument.....	45
FreeDocumentEx.....	46
GenerateColdFiles .....	47
GenerateOSFile.....	48
GetActiveDocument.....	49
GetAllArchives.....	50
GetAllOpenFolders .....	51
GetCatalogItemInfo .....	52
GetCurrentResultList.....	53

GetCurrentSelection .....	54
GetDataID.....	55
GetDescription .....	56
GetDocTypesFromArchive.....	57
GetEnvironment.....	58
GetFilesFromID.....	60
GetFilesFromIDEx .....	61
GetImageType .....	62
GetLastError .....	63
GetMainType.....	64
GetObjectFields .....	65
GetObjectFieldsEx.....	67
GetObjectName.....	69
GetObjectNameEx .....	70
GetObjectPath .....	71
GetObjectPathEx.....	72
GetObjectPattern.....	73
GetObjectPatternPath.....	74
GetObjectTypeInfo .....	75
GetRegTypeFromArchive .....	76
GetResourceString.....	77
GetResultFields.....	78
GetSignatureProperty .....	79
GetSelectedObject .....	80
GetSignDocumentResult .....	81
GetWDocPattern.....	82
GetWDocPatternNames .....	83
GoToDocPage .....	84
InfoWindow .....	85
InsertFileList .....	86
InsertFileListS.....	88
InsertIntoArchive .....	89
InsertIntoArchiveS.....	90
InsertIntoDocument .....	91
InsertIntoDocuments .....	93
InsertIntoRegister.....	94
InsertIntoRegisterS.....	95
InsertNewDMSObject.....	95
LicLogin .....	97
LicLogout.....	97
LinkDocuments.....	98
MergeArchives.....	99
MoveObject .....	100
OleDdeRequest.....	101
OpenAboDialog.....	102
OpenBrowser.....	103
OpenDataDlg.....	104
OpenModalBrowserDialog.....	105
OpenObjectID .....	107
OpenResultList.....	108
OpenObjectIDEx.....	109
OpenURL.....	110
OpenWorkItem .....	111
PrintDocumentID .....	112
RefreshFolderWindow .....	113
ScanDocument .....	114

ShowVariantsDialog.....	115
SelectObject .....	116
SendMail.....	117
SendMailMapi.....	118
SetBrowserURL .....	119
SetPlannedRetention.....	120
SetResultListSelection .....	121
SetResourceString.....	122
SetSignatureProperty .....	123
SetWaitingCursor.....	124
SignDocument.....	125
SignDocumentEx .....	126
StartArchiveRequest.....	127
StartDocRequest.....	129
StartRegRequest .....	131
StartClientRequest.....	133
StoreNotice .....	134
TransformXML .....	135
UndoCheckOut .....	136
UnlinkDocuments.....	137
UpdateArchiveData.....	138
UpdateArchiveDataS.....	140
UpdateDocFileList.....	141
UpdateDocShare .....	142
UpdateDocumentData.....	143
UpdateRegisterData .....	145
COM-Schnittstelle der Vorschau-Fenster.....	147
Einleitung.....	147
Alle COM-Befehle.....	147
Skript-Schnittstelle der Vorschau-Fenster .....	157
Einleitung.....	157
Funktionen zum Blättern über Dokumentgrenzen hinweg .....	157
Funktionen zur Fenstersteuerung des Clients.....	159
Funktionen zur Dashletsteuerung.....	164
Anhang: Konfiguration der enaio®-Drucker.....	170
Einleitung.....	170
Die Konfigurationsdatei.....	170
Protokollierung .....	173
Terminalserver .....	173
Anhang: Strukturbaumkataloge .....	175
Datenkonvertierung.....	175
Anhang: Ressourcenstorage.....	178
Allgemeines Schema der JSON-Parameter .....	178
Beispiel für Abfrageparameter.....	179
Beispiel für Einfüge- und Rückgabeparameter.....	179

## 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 **Component Object Model**, es ist ein von Microsoft eingeführtes Modell zur Kommunikation zwischen Windowsanwendungen. Häufig wird dafür auch der Begriff **OLE** verwendet, das für **Object Linking and Embedding** 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 Indexdaten der Objekte, die erzeugt werden sollen. Dateien für Anfragebefehle (**StartArchiveRequest**, **StartRegRequest**, **StartDocRequest**) enthalten Anfrageinformationen, also die gesuchte Indexierung. Beide Dateitypen enthalten außerdem die Bezeichnung der angefragten bzw. einzufügenden Objekte.

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

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

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

```
[ANFRAGE]
SCHRANK = Schränkname
KLAUSEL1=Schränkname@Feldname1=Feldwert1
KLAUSEL2= Schränkname@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=%Eingangsbefugnis@Datum1^6^10.03.2000
```

Beispiel für eine Klausel:

```
KLAUSEL1=%Eingangsbefugnis@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 Indexierung 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=Eingangsbefugnis@Erstellt=10.03.1997
```

Dabei ist **Eingangsbefugnis** der Objektname und **Vorlage** der Name eines Feldes des Objekts **Eingangsbefugnis**. Zurückgeliefert werden alle Eingangsbefugnisse, 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 Boolesche Ausdrücke von Booleschen Operatoren.

In einer Anfrage können mehrere Ausdrücke 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 Booleschen 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 Ordner Ebene zurückgegeben. Durch **1130^1^815** wird der Ordner mit der ID 815 als Parentordner festgelegt. **1133^1^0** gibt an, dass die gesuchten Dokumente 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 Ordner Ebene 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 Booleschen Operatoren (**BoolOP**)

#### Feldname/Feldnr.



Feldname	Feldnr	Beschreibung
<b>Ordner</b>		
STAMM_ID	1000	ID des Ordners
STAMM_TIME	1001	Zeitpunkt der Anlage
STAMM_LINKS	1002	Anzahl der Verknüpfungen mit diesem Ordner
<b>Objekte</b>		
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, 64 und systemid=0 = Verweisdokument, 64 und systemid!=0 = archiviert auf Fremdsystem
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
<b>Register</b>		
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
<b>Ordner-Dokument-Relation</b>		
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
<b>Mappe-Dokument-Relation</b>		
MDEL	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)
MSEND	1150	Absender (unbenutzt)
<b>Mappe</b>		
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)
<b>Benutzer</b>		
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 Objektamen 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

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## 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, dass 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

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

## AdjustRetention

<b>Definition</b>	AdjustRetention	(long long Variant	lObjectID, lObjectType, varRetentionDate)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Passt das Retention-Datum an das geplante Retention-Datum an.		
<b>Parameter</b>	<b>lObjectID</b> <b>lObjectType</b> <b>varRetentionDate</b>	ID des Objekts Typ des Objekts das angepasste Retention-Datum im Format DD.MM.YYYY als Rückgabewert	
<b>Rückgabewert</b>	0 -1 -7 -22 -38 -40 -112	Retention-Datum erfolgreich angepasst Retention-Datum konnte nicht angepasst werden Objekttyp unbekannt Objekt mit gegebener ID existiert nicht Ungültiger Dokumenttyp (es wurde ein Ordner oder Registertyp angegeben) Ungültiger Parameter wurde übergeben Das angegebene Objekt wurde (noch) nicht archiviert.	
<b>Bemerkung</b>	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

<b>Definition</b>	ApplicationLogin (String strUser)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermöglicht das Einloggen eines neuen Benutzers.
<b>Parameter</b>	strUser JSON-String mit Benutzername und Passwort.

```
{
    user: name,
    password: secret
}
```

<b>Rückgabewert</b>	1 kein Fehler 0 Fehler, Einloggen nicht möglich -119 Einloggen zurzeit nicht möglich, da der Client gerade gestartet wird
---------------------	---

<b>Bemerkung</b>	Wenn der String <b>DIALOG</b> übergeben wird, dann wird in jedem Fall der Login-Dialog geöffnet. Dies steht nur zur Verfügung, wenn die Passwortverifizierung nicht über <b>Novell NetWare</b> , sondern über den Benutzerdialog von enaio abgewickelt wird. Beim Fehler <b>-119</b> 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.
------------------	--

Aus Abwärtskompatibilität ist es möglich als strUser einen String bestehend aus Benutzername und Passwort, getrennt durch ein '@' zu übergeben. In diesem Fall darf das Passwort des entsprechenden Nutzers nicht das Zeichen '#' enthalten.

### Optionen

#### Beispiel

```
set ax = CreateObject("optimal_AS.Application")

do while ax.ApplicationLogin ("{"user": "name",
    "password": "secret"}") = -119
    idleLittleTime() 'your own idle function
loop
```



## ArchiveMailFile

<b>Definition</b>	ArchiveMailFile (String config)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Legt E-Mails am gegebenen Standort mit angegebenen Dokumenttyp über den EMS Service an.
<b>Parameter</b>	config Konfiguration als String
<b>Rückgabewert</b>	<p>Die Rückgabe ist ein Objekt, welches im Fehlerfall ein Integer ist oder im Erfolgsfall ein String.</p> <ul style="list-style-type: none"> <li>-101 Konfiguration kein String</li> <li>-102 Keine Konfiguration übergeben (String ist leer)</li> <li>-103 Keine E-Mail-Dateien angegeben</li> <li>-104 Mindestens eine angegebenen E-Mail-Dateien hat eine falsche Datei-Endung</li> <li>-105 Keine oder ungültige Standort-ID angegeben</li> <li>-106 Keinen oder ungültigen Standorttyp angegeben</li> <li>-107 Angegebener Standorttyp kein Register oder Ordner</li> <li>-108 Angegebener Standorttyp unbekannt</li> <li>-109 Angegebener Standort existiert nicht</li> <li>-110 Kein Dokumenttyp angegeben</li> <li>-111 Angegebener Dokumenttyp unbekannt</li> <li>-112 Am angegebenen Standort kann kein Dokument von angegebenen Dokumenttyp erstellt werden</li> <li>-113 Kein Feld mit dem internen Namen 'MAIL_DIGEST' im angegebenen Dokumenttyp vorhanden</li> <li>-114 JSON-Parser meldet Fehler</li> <li>-201 bis -213 Extraction Service meldet Fehler</li> <li>-301 bis -313 EMS Service meldet Fehler</li> </ul> <p>Der genaue Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.</p>
<b>Bemerkung</b>	Die Konfiguration kann als JSON- oder INI-Format übergeben werden.
<b>Beispiel</b>	Beispiel (INI):

```

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

sInsert = "[LOCATION]" & vbCrLf
sInsert = sInsert & "ID=1707" & vbCrLf
sInsert = sInsert & "TYPE=6488090" & vbCrLf
sInsert = sInsert & "[MAIL]" & vbCrLf
sInsert = sInsert & "TYPE=393239" & vbCrLf
sInsert = sInsert & "[FILES]" & vbCrLf
sInsert = sInsert &
"FILE0=C:\Users\ludwig\Documents\0010DE80.eml" & vbCrLf
sInsert = sInsert &
"FILE1=C:\Users\ludwig\Documents\1C772761.eml" & vbCrLf

sResult = Application.ArchiveMailFile(sInsert)

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

```

Ist das Einfügen der E-Mails erfolgreich, dann wird ein String mit folgendem Inhalt zurückgegeben:

```

[LOCATION]
ID=1707
TYPE=6488090
[MAIL]
TYPE=393239
[FILES]
FILE0=C:\Users\user\AppData\Local\Temp\OSTEMP\003703AD.eml@114366
FILE1=C:\Users\user\AppData\Local\Temp\OSTEMP\003703BD.eml@114367

```

Die IDs der neuen Dokumente sind hinter den Dateinamen angehängt.

Beispiel (JSON):

```

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

sInsert = "{\"objects\": [{\"options\":
{ \"ems:parent:enaio:objectTypeId\": { \"value\":
\"6488088\"}, \"ems:parent:enaio:objectId\": { \"value\":
\"114162\"}, \"ems:target:enaio:objectTypeId\": { \"value\":
\"393239\"}, \"client:showIndexdata\": { \"value\":
true}, \"client:refreshTarget\": { \"value\":
true}, \"client:files\": { \"value\":
[\"C:\\Users\\ludwig\\Documents\\0010DE80.eml\", \"C:\\Users\\user
\\Documents\\1C772761.eml\"]} } ] }"

sResult = Application.ArchiveMailFile(sInsert)

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

```

Ist das Einfügen der E-Mails erfolgreich, dann wird ein String mit folgendem Inhalt zurückgegeben:

```

{
"objects": [{

```

```

        "properties": {
            "enaio:objectId": {
                "value": 114369
            }
        },
        "options": {
            "client:stored": {
                "value":
"C:\\Users\\ludwig\\AppData\\Local\\Temp\\OSTEMP\\0077E675.eml"
            }
        },
        {
            "properties": {
                "enaio:objectId": {
                    "value": 114370
                }
            },
            "options": {
                "client:stored": {
                    "value":
"C:\\Users\\user\\AppData\\Local\\Temp\\OSTEMP\\0077E676.eml"
                }
            }
        }
    ]
}

```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## CalcFileDigest

<b>Definition</b>	CalcFileDigest	(String FileName, VARIANT* vRetDigest)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Ermittelt den Hash-Wert (Digest) der angegebenen Datei	
<b>Parameter</b>	<b>FileName</b>	Der Dateiname
	<b>vRetDigest</b>	Der ermittelte Hash-Wert (Digest)
<b>Rückgabewert</b>	<b>0</b>	kein Fehler
	<b>-1</b>	Hash-Wert konnte nicht ermittelt werden
	<b>-32</b>	Die angegebene Datei existiert nicht
<b>Bemerkung</b>		
<b>Optionen</b>		
<b>Beispiel</b>		

# CheckInDocument

<b>Definition</b>	CheckInDocument	(long long String	IDocID, IDocType, strSourcePath)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Checkt ein für die externe Bearbeitung ausgechecktes Dokument wieder ein.		
<b>Parameter</b>	<b>IDocID</b> <b>IDocType</b> <b>strSourcePath</b>	Dokumenten-ID Dokumententyp Pfad(ohne Dateiname) zur einzucheckenden Datei	
<b>Rückgabewert</b>	0 -7 -53 -54 -55 -56 -57	kein Fehler Unbekannter Dokumenttyp Dokument wurde nicht ausgecheckt Dokument wurde nicht für externe Bearbeitung ausgecheckt Datei konnte nicht in Cache-Verzeichnis kopiert werden Name des Dokuments im Cache konnte nicht ermittelt werden Quelldatei existiert nicht	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.

**Bemerkung**

**Optionen**

**Beispiel**

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## CheckLicence

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

# CheckObjectAccess

<b>Definition</b>	CheckObjectAccess	(long long long long short	lObjectID, lObjectType, lDesiredAccess, lFlags, nShowError)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Prüft Zugriffsrechte für das angegebene Objekt		
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID	
	<b>lObjectType</b>	Objekttyp	
	<b>lDesiredAccess</b>	<b>0</b>	Indexdaten lesen
		<b>1</b>	Indexdaten schreiben
		<b>2</b>	Objekt löschen
		<b>3</b>	Objekt ausgeben (z.B. Drucken)
		<b>4</b>	Objekt schreiben
		<b>5</b>	Auscheckstatus
		<b>6</b>	Archivstatus
		<b>7</b>	gibt an, ob das Objekt in der Workflowablage liegt (Rückgabewert =1) oder nicht ( <b>0</b> )
		<b>8</b>	gibt an, ob das Objekt zum Löschen vorgemerkt wurde. Rückgabe: <b>0</b> , Objekt nicht zum Löschen vorgemerkt <b>1</b> , Objekt zum Löschen vorgemerkt <b>-22</b> Unbekanntes Objekt.
	<b>lFlags</b>	ab 4.20 SpII wird dieser Wert auf <b>1</b> gesetzt, so werden die richtigen Benutzerrechte angefragt, auch wenn das Sicherheitssystem für dieses Objekt nicht beachtet werden soll.	
	<b>nShowError</b>	wenn ungleich <b>0</b> , wird ggf. eine Fehlerbox angezeigt	
<b>Rückgabewert</b>	<b>0</b>	Benutzer hat kein Zugriffsrecht	
	<b>1</b>	Benutzer hat Zugriffsrecht	
	<b>-1</b>	Unbekanntes Recht	
	<b>-7</b>	Dokumenttyp unbekannt	
	<b>-22</b>	Unbekanntes Objekt	
	<b>-46</b>	Unbekannter Benutzer	
	<b>-51</b>	Dokument enthält keine Dateien	
	<b>-52</b>	Objekt bereits archiviert	
	<b>-58</b>	Dokument bereits ausgelagert	
	<b>-59</b>	Dokument von anderem Benutzer ausgecheckt	
	<b>-60</b>	Objekt ist kein W-Dokument	

**Bemerkung**

**Optionen**

**Beispiel**

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## CheckOutDocument

<b>Definition</b>	CheckOutDocument	(long IDocID, long IDocType, String strPath, Variant* varRetDocName)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Checkt ein Dokument für die externe Bearbeitung aus.	
<b>Parameter</b>	<b>IDocID</b>	Dokumenten-ID
	<b>IDocType</b>	Dokumententyp
	<b>strPath</b>	Pfad in den das Dokument ausgecheckt werden soll
	<b>varRetDocName</b>	hier wird der vollständige Pfad und Dateiname des ausgecheckten Dokuments geliefert
<b>Rückgabewert</b>	<b>0</b>	kein Fehler
	<b>-1</b>	Dokument konnte nicht ausgecheckt werden
	<b>-7</b>	unbekannter Dokumenttyp
	<b>-51</b>	Dokument enthält keine Dateien
	<b>-52</b>	Objekt bereits archiviert
	<b>-58</b>	Dokument bereits extern ausgecheckt
	<b>-59</b>	Dokument von anderem Benutzer ausgecheckt
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.	

### — Bemerkung

#### Optionen

#### Beispiel

**Hinweise:** Die ausgecheckte Datei darf nicht umbenannt werden.



## ClearSignatureProperties

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

## CloseAllWindows

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

## CloseBrowser

<b>Definition</b>	CloseBrowser (String GUID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Schließt ein Browser-Fenster im Client (vgl. OpenBrowser)
<b>Parameter</b>	<b>GUID</b> Die eindeutige ID des Browser-Fensters
<b>Rückgabewert</b>	<b>0</b> Browser-Fenster geschlossen <b>-1</b> Fehler

## CloseObjectID

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

## ConvertImage

<b>Definition</b>	ConvertImage	(String String short short short short	strSource, strDestination, nFormat, nCompression, nBitsPerPixel, nFlags)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Die Funktion konvertiert eine Bilddatei ins angegebene Format.		
<b>Parameter</b>	<b>strSource</b>	Quelldatei	
	<b>strDestination</b>	Zieldatei	
	<b>nFormat</b>	Zielformat, siehe GetImageType	
	<b>nCompression</b>	Kompressionsart oder Verlustfaktor für JPEG-Format, 1 empfohlen	
	<b>nBitsPerPixel</b>	Farbtiefe 1, 2, 4, 8, 16, 24 oder 0 für Farbtiefe der Quelldatei	
	<b>nFlags</b>	unbenutzt	
<b>Rückgabewert</b>	0	Datei wurde erfolgreich konvertiert	
	<> 0	Fehler	
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

# CopyObject

<b>Definition</b>	CopyObject	(long long long long short Variant	(IObjectID, IObjectType, IFolderID, IRegisterID, nFlags, varRetObjectID)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Erstellt eine Kopie eines Objektes incl. Mehrfachfelder und Dokument.		
<b>Parameter</b>	<b>IObjectID</b> <b>IObjectType</b> <b>IFolderID</b> <b>IRegisterID</b> <b>nFlags</b> <b>varRetObjectID</b>	Objekt-ID des zu kopierenden Objekts Objekttyp des zu kopierenden Objekts ID des Zielordners, zum Kopieren eines Ordners sollte dieser Wert 0 sein. ID des Zielregisters, 0, falls nicht in ein Register kopiert werden soll 0 Indexierung und Dokument kopieren 1 nur Indexierung kopieren hier wird die ID des neuen Objekts zurückgegeben	
<b>Rückgabewert</b>	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 <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	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.		
<b>Optionen</b>			
<b>Beispiel</b>			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## CreateDocumentLink

<b>Definition</b>	CreateDocumentLink (long long long long	lObjectID, lObjectType, lTargetID, lTargetType)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Erzeugt einen neuen Verweis auf ein Dokument	
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID
	<b>lObjectType</b>	Objekttyp
	<b>lTargetID</b>	Index des Zielordners/-registers.
	<b>lTargetType</b>	Objekttyp des Zielordners/-registers.
<b>Rückgabewert</b>	-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	Angebener 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 <b>GetLastError()</b> 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**

<b>strFrom</b>	Senderadresse
<b>strTo</b>	Empfänger
<b>strCC</b>	CC
<b>strBCC</b>	BCC
<b>strSubject</b>	Betreff
<b>strBody</b>	der Text
<b>strAttachments</b>	angehangene Dateien getrennt durch Pipe-Zeichen
<b>strCreationTime</b>	Erstellungszeit in der Form: Tag.Monat.Jahr Stunde:Minute
<b>varRetFilename</b>	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

<b>Definition</b>	DecodeIMAPFile	(String Variant Variant Variant Variant Variant Variant Variant Variant)	strFilename, strMailDate, strFrom, strTo, strCC, strBCC, strSubject, strBody, strAttachments)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Dekodiert eine IMAP-Datei und gibt deren Inhalt zurück.		
<b>Parameter</b>	<b>strFilename</b>	Vollständiger Pfad und Dateiname der IMAP Datei	
	<b>strMailDate</b>	Das Sendedatum der Mail wird hier zurückgegeben.	
	<b>strFrom</b>	Der Name des Absenders wird hier zurückgegeben.	
	<b>strTo</b>	Die Empfängerliste (durch Semikolon getrennt) wird hier zurückgegeben.	
	<b>strCC</b>	Die CC-Liste (durch Semikolon getrennt) wird hier zurückgegeben.	
	<b>strBCC</b>	Die BCC-Liste (durch Semikolon getrennt) wird hier zurückgegeben.	
	<b>strSubject</b>	Der Betreff wird hier zurückgegeben.	
	<b>strBody</b>	Der Mailtext wird hier zurückgegeben.	
	<b>strAttachments</b>	Die Attachments werden hier durch Semikolon getrennt zurückgegeben	
<b>Rückgabewert</b>	0	wenn kein Fehler	
	-1	wenn die Datei nicht dekodiert werden konnte	
<b>Bemerkung</b>			

## CreateNewDocShare

<b>Definition</b>	CreateNewDocShare (long long string string string IIdent, IType, Rights Users Info)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Erzeugt neue Freigaben des angegebenen Dokuments für die angegebenen Benutzer
<b>Parameter</b>	<b>IIdent</b> Dokument-ID <b>IType</b> Dokumenttyp <b>Rights</b> Rechte, die vorbelegt werden (RWXU) <b>Users</b> Benutzer-ID's der Benutzer, welchen das angegebene Dokument freigegeben werden sollen, getrennt durch Semikolon <b>Info</b> Infotext für die Freigabe
<b>Rückgabewert</b>	<b>0</b> "Neue Freigabe(n) angelegt." <b>-125</b> "Ein modaler Dialog ist geöffnet." <b>-130</b> "Angegebener DMS-Objektyp unbekannt." <b>-131</b> "Angegebener DMS-Objektyp kein Dokument." <b>-132</b> "Nur Rechte 'RXWU' sind zulässig." <b>-133</b> "Angegebenes Dokument existiert nicht oder wurde gelöscht." <b>-134</b> "Dokumentfreigaben nicht möglich." <b>-135</b> "Die erforderliche Systemrolle um Dokumente freizugeben ist nicht vorhanden." <b>-136</b> "Mindestens ein angegebener Benutzer existiert nicht." <b>-137</b> "Abbruch durch Benutzer." <b>-138</b> "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

<b>Definition</b>	DeleteFromArchive (String strParam)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Löscht den angegebenen Ordner oder das angegebene Dokument.
<b>Parameter</b>	<b>strParam</b> 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**.

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

HIER LOCHEN ODER DIGITAL-ARCHIVIEREN

## DeleteResourceString

<b>Definition</b>	Application.GetASSystem.DeleteResourceString (String Key, String Language, String JSON_Request )
<b>Typ</b>	Methode
<b>Beschreibung</b>	Löscht selbst angelegte Sprachressourcen-Schlüssel aus der Ressourcenstorage-Tabelle der Datenbank.
<b>Parameter</b>	<p><b>Key</b> Zu löschender Schlüssel Beispiel: <code>Project.key1.Workflow_3</code></p> <p><b>Language</b> Sprachkürzel. Beispiel: <code>de_DE</code></p> <p><b>JSON_Request</b> String im JSON-Format, um mehrere Schlüssel gleichzeitig zu löschen.</p>
<b>Rückgabewert</b>	0 kein Fehler
<b>Bemerkung</b>	<ul style="list-style-type: none"> <li>▪ Parameter können die Platzhalter '*' und '?' enthalten.</li> <li>▪ Der Parameter 'JSON_Request' und die Parameter 'Key/Language' können unabhängig voneinander verwendet werden.</li> <li>▪ Die Parameter 'Key' und 'Language' werden immer getrimmt: Leerzeichen am Anfang und Ende werden entfernt.</li> </ul> <p>Die Beschreibung des JSON-Formats befindet sich im 'Anhang: Ressourcenstorage'.</p>
<b>Optionen</b>	-
<b>Beispiel</b>	<pre>Set assystem = Application.GetASSystem value = assystem.DeleteResourceString("Project.key1.*", "en_US", "")</pre>

## DoPrefetch

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

## EMailImport

<b>Definition</b>	EMailImport (Variant & config)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ablage von mehreren E-Mails an einen Standort mit Dubletten-Prüfung
<b>Parameter</b>	config JSON zur Konfiguration (siehe Beispiel)
<b>Rückgabewert</b>	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: 'objectId' 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", "parentObjectType": "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", "parentObjectType": "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

```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## ExecuteRequest

<b>Definition</b>	ExecuteRequest (String strRequestName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Führt eine gespeicherte Anfrage aus.
<b>Parameter</b>	<b>strRequestName</b> Name der gespeicherten Anfrage
<b>Rückgabewert</b>	<b>0</b> kein Fehler <b>-1</b> Es existieren keine gespeicherten Anfragen <b>-2</b> Es existiert keine Anfrage mit dem angegebenen Namen Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	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‘

<b>Optionen</b>	<b>OPT1=1</b> Erzwingt das Öffnen der Anfragemaske
-----------------	---

<b>Beispiele</b>	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

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

## ExistBrowser

<b>Definition</b>	ExistBrower (String GUID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt, ob ein bestimmtes Browser-Fenster im Client geöffnet ist (vgl. OpenBrowser).
<b>Parameter</b>	<b>GUID</b> Die eindeutige ID des Browser-Fensters
<b>Rückgabewert</b>	<b>0</b> Browser-Fenster existiert <b>-1</b> Browser-Fenster existiert nicht

## FindObjectType

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

## FindObjectTypeEx

<b>Definition</b>	FindObjectTypeEx	(long long VARIANT* IObjectID IType, vRetObjTypes)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Ermittelt zur angegebenen Objekt-ID den zugehörigen Objekttyp	
<b>Parameter</b>	<b>IObjectID</b>	Objekt-ID zu dem der Objekttyp ermittelt werden soll
	<b>IType</b>	Art des zu ermittelnden Objekttyps: <b>0</b> Dokument <b>1</b> Ordner <b>2</b> Register
	<b>vRetObjTypes</b>	hier wird der ermittelte Objekttyp zurückgegeben
<b>Rückgabewert</b>	<b>0</b>	kein Fehler
	<b>-26</b>	Objekt-ID ungültig
	<b>-40</b>	angegebener Typ ungültig
<b>Bemerkung</b>	Diese Funktion beschleunigt das Finden des Objekttyps durch die Angabe der Art des Typs.	
<b>Optionen</b>		
<b>Beispiel</b>		

## FreeDocument

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

## FreeDocumentEx

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

### Optionen

### Beispiel

## GenerateColdFiles

<b>Definition</b>	GenerateColdFiles	(String String String VARIANT*	strFile1, strFile2, strToken, vRetFileList)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Erzeugt aus ASCII-COLD-Import-Dateien entsprechende Bildobjekte.		
<b>Parameter</b>	<b>strFile1</b>	entweder Datendatei oder Steuerdatei	
	<b>strFile2</b>	entweder Datendatei oder Steuerdatei	
	<b>strToken</b>	Trennzeichen für Dateiliste	
	<b>vRetFileList</b>	Liste mit den Dateinamen der erzeugten Bilder getrennt durch die mit strToken angegebenen Zeichen.	
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-83</b>	Unterverzeichnis für die Cold-Dateien konnte nicht erstellt werden.	
	<b>-84</b>	Bild konnte nicht erzeugt werden	
	<b>-85</b>	Datei konnte nicht ins Zielverzeichnis verschoben werden.	
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

## GenerateOSFile

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



## GetActiveDocument

<b>Definition</b>	GetActiveDocument (String strDocName, BOOL bOpen, VARIANT vRetVal)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Liefert das COM-Objekt zu einem optimal_AS: Active Document.
<b>Parameter</b>	<p><b>strDocName</b> Name des Active Documents, wie in der <i>as.cfg</i> angegeben.</p> <p><b>bOpen</b> <b>TRUE</b> wenn das Dokument geöffnet werden soll, falls es nicht geöffnet ist.</p> <p><b>vRetVal</b> hier wird eine Fehlernummer zurückgegeben</p>
<b>Rückgabewert</b>	<p><b>0</b> kein Fehler</p> <p><b>-70</b> kein Active Document mit diesem Namen gefunden</p> <p><b>-72</b> kein optimal_AS: Active Document</p> <p>Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.</p>
<b>Bemerkung</b>	
<b>Optionen</b>	
<b>Beispiel</b>	

## GetAllArchives

**Definition** GetAllArchives()

**Typ** Methode

**Beschreibung** Ermittelt alle Schränke

**Parameter**

**Rückgabewert** String mit allen verfügbaren Schränken

**Bemerkung** Der Rückgabestring hat folgendes Format:

```
SCHRANK1, OBJEKTYP1; SCHRANK2, OBJEKTYP2; ... ;  
SCHRANKn, OBJEKTYPn
```

**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**

## GetCatalogItemInfo

<b>Definition</b>	GetCatalogItemInfo (string long string short	sRawValue, IObjectType, sFieldName, iMode)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Ermittelt einen Katalogwert eines mehrsprachigen Katalogs in der im Client eingestellten Objektdefinitions-Sprache	
<b>Parameter</b>	<b>sRawValue</b> <b>IObjectType</b> <b>sFieldName</b> <b>iMode</b>	Wert in der Datenbank (Raw-Value) Dokumenttyp Feldname, dem der Katalog zugeordnet ist Modus
<b>Rückgabewert</b>	String: Der übersetzte Katalogwert. Ist die Rückgabe ein Leerstring, kann mit <b>GetLastError()</b> eine Fehlermeldung ermittelt werden.	
<b>Bemerkung</b>	iMode = 0 ermittelt den übersetzten Katalogwert einschließlich der durch ein  -Zeichen (Pipe) getrennten Beschreibung. iMode = 1 ermittelt den übersetzten Katalogwert. iMode = 2 ermittelt, falls vorhanden, die durch ein  -Zeichen (Pipe) getrennten Beschreibung.	

### — Beispiel

```

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

' Test-System: 10.1.7.113#4000
' Benutzer: user
' Passwd: secret

sRet = Application.GetCatalogItemInfo("schrägheck", 196618,
"%VEHICLETYPE%", 0)

Application.ActivateApp 1

if sRet = "" then
    sErr = Application.GetLastError
end if

if sRet = "" && sErr <> ""
    sRet = "Error: " + sErr
end if

MsgBox sRet

set Application = nothing

```

## 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 übergebenen 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

<b>Definition</b>	GetCurrentSelection (VARIANT* varResult)	
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Liefert die in der aktuellen Trefferliste selektierten IDs und Objekttypen	
<b>Parameter</b>	<b>varResult</b> hier werden die Indizes und Objekttypen in folgender Form zurückgegeben: <table border="1"><tr><td>ObjektID1,Objekttyp1;ObjektID2,Objekttyp2;.....</td></tr></table>	ObjektID1,Objekttyp1;ObjektID2,Objekttyp2;.....
ObjektID1,Objekttyp1;ObjektID2,Objekttyp2;.....		
<b>Rückgabewert</b>	Der Rückgabewert entspricht der Anzahl der selektierten Objekte.	
<b>Bemerkung</b>		
<b>Optionen</b>		
<b>Beispiel</b>		

## GetDataID

<b>Definition</b>	GetDataID	(long long short short	lObjectID, lObjectType, nMode, bWriteToFile)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Gibt Objektdaten zurück, d.h. Feldnamen und -werte		
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID des zu öffnenden Objekts	
	<b>lObjectType</b>	Objekttyp	
	<b>nMode</b>	Steuert den Rückgabewert.	
	<b>0</b>	Gibt Feldnamen und -inhalte zurück	
	<b>1</b>	Gibt Basisparameter zurück, wenn es sich nicht um Ordner oder Register handelt	
	<b>2</b>	Gibt Namen und Werte von Mehrfachfeldern zurück	
	<b>10</b>	Entspricht dem Wert <b>0</b> , gibt aber die internen Feldnamen zurück	
	<b>12</b>	Entspricht dem Wert <b>2</b> , gibt aber die internen Feldnamen zurück	
	<b>bWriteToFile</b>	Ist der Wert <b>0</b> , dann werden die Objektdaten in einem String zurückgegeben. Bei Wert <b>1</b> 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

<b>Definition</b>	GetDescription	(String long String	strShortName, IObjectType, strFieldName)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Liefert den Beschreibungstext aus einem Listenfeld oder Strukturbaum zu einem gegebenen Kürzel.		
<b>Parameter</b>	<b>strShortName</b>	Kürzel, zu dem der Beschreibungstext geliefert werden soll.	
	<b>IObjectType</b>	Objektyp	
	<b>strFieldName</b>	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

<b>Definition</b>	GetDocTypesFromArchive (String strSchrankName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt alle zu einem Schrank gehörenden Dokumenttypen.
<b>Parameter</b>	<b>strSchrankName</b> String mit dem Schrankbezeichner
<b>Rückgabewert</b>	String mit den zum Schrank gehörenden Dokumenttypen.
<b>Bemerkung</b>	Der Rückgabestring hat folgendes Format:

```
DOKUMENT1, OBJEKTYP1; DOKUMENT2, OBJEKTYP2 ...  
DOKUMENTn, OBJEKTYPn
```

### Optionen

#### Beispiel

Codebeispiel:

```
Helpstr=MyAX.GetDocTypesFromArchive
```

Helpstr enthält z.B. folgende String:

```
Eingangsbefugnis, 131085; Ausgangsbefugnis, 262159; Attribut, 131086
```

## GetEnvironment

<b>Definition</b>	GetEnvironment (short nEnvType)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt Einstellungen aus dem Archivsystem.
<b>Parameter</b>	<b>nEnvType</b> Nummer der zu ermittelnden Einstellung
<b>Rückgabewert</b>	String mit gewünschter Einstellung
<b>Bemerkung</b>	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 Dokumente 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) ausgegeben (vgl. 29).
	25 ermittelt die ID der im Client eingestellte Sprache der Objektdefinition: z.B. 7-Deutsch, 9-Englisch.

- 26 ermittelt das im Client eingestellte Farbschema: Weiß 0 4, Hellgrau = 5, Dunkelgrau = 6, Schwarz = 1.
- 27 ermittelt die aktuelle SessionGUID des Clients.
- 28 ermittelt die im Client eingestellte Vergrößerung des Schriftgrads: 0 = keine, 1 = leicht, 2 = stark, 3 = sehr stark.
- 29 ermittelt die im Client eingestellte GUI-Sprache.  
Format:  

```
{
  "Languages": [
    {
      "ID": 35212,
      "Short": "De",
      "LangID": 1031,
      "Locale": "German.1252",
      "Localename": "de_DE",
      "FlaggIconID": 1073742795
    }
  ]
}
```
- 30 ermittelt die im Client einstellbaren GUI-Sprachen. Format analog zu 29.
- 31 ermittelt die im Client eingestellte Vergrößerung des Schriftgrads in Trefferlisten: 0 = keine bis 11 = maximal.
- 32 ermittelt die im Client einstellbaren Sprachen der Objektdefinition.  
Format:  

```
{
  "Languages": [
    {
      "Name": "Deutsch",
      "Short": "De",
      "LangID": 7,
      "Locale": "German.1252",
      "Localename": "de_DE",
      "Active": true,
      "FlaggIconID": 1073742795
    },
    {
      "Name": "Englisch",
      "Short": "En",
      "LangID": 9,
      "Locale": "English.1252",
      "Localename": "en_US",
      "Active": false,
      "FlaggIconID": 1073742935
    }
  ]
}
```
- 33 ermittelt die im Client eingestellte Sprache (Sprachkürzel) der Objektdefinition. Beispiel: `fr_FR`

## Optionen

### Beispiel

## GetFilesFromID

<b>Definition</b>	GetFilesFromID	(long long String	lObjectID, lObjectType, strToken)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ermittelt die zu dem Objekt gehörenden Imagefiles. Sind die Images archiviert, so werden sie zunächst in den Cache geladen.		
<b>Parameter</b>	<b>lObjectID</b>	ID des Objekts	
	<b>lObjectType</b>	Objekttyp	
	<b>strToken</b>	Trennzeichen	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Rückgabewert</b>	<b>String</b>	mit Dateinamen inkl. Pfad	
	<b>Leerer String</b>	wenn Fehler	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	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.		
<b>Optionen</b>			
<b>Beispiel</b>			

## GetFilesFromIDEx

<b>Definition</b>	GetFilesFromIDEx	(long long String BOOL VARIANT	(long long String bWriteProtected, vFileNames)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ermittelt die zu einem Objekt gehörigen Dateien.		
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID	
	<b>lObjectType</b>	Objekttyp	
	<b>strToken</b>	Trennzeichen, das zur Trennung der Namen im Rückgabestring verwendet werden soll	
	<b>bWriteProtected</b>	TRUE	Dokument schreibgeschützt anfordern
	<b>vFileNames</b>	hier werden die Dateinamen zurückgegeben	
<b>Rückgabewert</b>	0	kein Fehler	
	-7	Dokumenttyp unbekannt	
	-15	Dokument-ID unbekannt	
	-51	Dokument hat keine Dateien	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	Im Unterschied zu <b>GetFilesFromID()</b> kann angegeben werden, ob die Dateien ausgecheckt, oder ob sie schreibgeschützt angefordert werden sollen.		
<b>Optionen</b>			
<b>Beispiel</b>			

## GetImageType

<b>Definition</b>	GetImageType (String strSource)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Bestimmt den Bildtyp der angegebenen Datei.
<b>Parameter</b>	<b>strSource</b> vollständiger Pfad und Dateiname der Bilddatei
<b>Rückgabewert</b>	<p>&gt; 0 Dateityp siehe Bemerkung</p> <p>&lt; 0 Fehler</p> <p>-20009 Dateiformat fehlerhaft</p>
<b>Bemerkung</b>	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

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

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

## GetMainType

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



## GetObjectFields

<b>Definition</b>	GetObjectFields	(String long short)	(strObjectName, lObjectType, nMode)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ermittelt alle Felder eines Objektes. Typen und Längen der Felder werden ebenfalls ermittelt. Objekttyp kann hier ein Ordner, Register oder Dokument sein.		
<b>Parameter</b>	<b>strObjectName</b>	wird nicht verwendet.	
	<b>lObjectType</b>	Objekttyp	
	<b>nMode</b>	Modus (siehe Bemerkung)	
<b>Rückgabewert</b>	<b>String</b> mit Feldbezeichnern <b>Leerer String</b> wenn Fehler. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		

**Bemerkung**

Der Rückgabestring hat folgendes Format:

Für nMode = 0:

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

Für nMode = 1

```
Feldbezeichner1\Feldtyp1\Feldlänge1\Tabellename.Feldname1
TAB
Feldbezeichner2\Feldtyp2\Feldlänge2\Tabellename.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

<b>Definition</b>	GetObjectFieldsEx (String long short strObjectName, IObjectType, nMode)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt alle Felder eines Objektes. Diese Funktion entspricht der Funktion <b>GetObjectFields()</b> . 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.

<b>Parameter</b>	<b>strObjectName</b> wird nicht verwendet <b>IObjectType</b> Objekttyp <b>nMode</b> Modus (siehe Bemerkung)
------------------	---

<b>Rückgabewert</b>	<b>String</b> mit Feldbezeichnern <b>Leerer String</b> wenn Fehler. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
---------------------	---

<b>Bemerkung</b>	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\x30\0 Bemerkung\x50\1
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## GetObjectName

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

## GetObjectNameEx

<b>Definition</b>	GetObjectNameEx	(long long short VARIANT*	lObjectID, lType, nMode, strReturnObjectName)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Liefert den Objektnamen zur gegebenen Objekt-ID		
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID	
	<b>lType</b>	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	
	<b>nMode</b>	Art des zu liefernden Ergebnisses 0 Objekttypname 1 Interner Objekttypname 2 Tabellename des Objekttyps 3 UID des Objekttyps	
	<b>strReturnObjectName</b>	hier wird der Name des Objekts zurückgegeben	
<b>Rückgabewert</b>	0	kein Fehler	
	-1	Fehler	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

# GetObjectPath

<b>Definition</b>	GetObjectPath (long lObjectID, long lObjectType)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt den Standortpfad zu einem Objekt. Der Ordner sowie alle Register werden ermittelt. Objekttyp kann hier ein Ordner, Register oder Dokument sein.
<b>Parameter</b>	<b>lObjectID</b> Objekt-ID <b>lObjectType</b> Objekttyp
<b>Rückgabewert</b>	<b>String</b> mit dem Pfad zum Objekt. <b>Leerer String</b> wenn Fehler. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.

**Bemerkung** Der Rückgabestring hat folgendes Format:

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

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## GetObjectPathEx

<b>Definition</b>	GetObjectPathEx (long lObjectID, long lObjectType, VARIANT* vRetObjectPath)
<b>Typ</b>	Methode
<b>Beschreibung</b>	<p>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.</p>
<b>Parameter</b>	<p><b>lObjectID</b> Objekt-ID  <b>lObjectType</b> Objekttyp  <b>vRetObjectPath</b> hier wird der Pfad zurückgegeben.</p>
<b>Rückgabewert</b>	<p>-3 Schrank unbekannt  -5 Register unbekannt  -22 Dokumenttyp unbekannt  Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.</p>
<b>Bemerkung</b>	<p>Der Rückgabestring hat folgendes Format:</p> <div style="border: 1px solid black; padding: 2px; margin: 5px 0;"> <p>SCHRANKINDEX, SCHRANKTYP; REGISTERINDEX1, REGISTERTYP; REGISTERINDEX2, REGISTERTYP; ...; REGISTERINDEXn, REGISTERTYP</p> </div> <p>Registerangaben werden dann zurückgegeben, wenn sich das Objekt in einem Register befindet. Die einzelnen Pfade bei Verweisdokumenten werden durch   (Pipe) getrennt.</p> <p>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.</p> <p>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:</p> <div style="border: 1px solid black; padding: 2px; margin: 5px 0;"> <p>Ordner-ID, OrdnerTyp; REGISTER3-ID, REGISTER3-Typ; REGISTER2-ID, REGISTER2-Typ; REGISTER1-ID, REGISTER1-Typ</p> </div>
<b>Optionen</b>	
<b>Beispiel</b>	



## GetObjectPattern

<b>Definition</b>	GetObjectPattern	(long long long string	(IIdent, IType, IMode, Titel)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ermittelt den Titel des angegebenen Objekts, wie er als Fenstertitel verwendet wird oder konfiguriert ist.		
<b>Parameter</b>	<b>IIdent</b>	ID des Objekts	
	<b>IType</b>	Typ des Objekts	
	<b>IMode</b>	zurzeit nur '1'	
	<b>Titel</b>	Titel des angegebenen Objekts wird zurückgegeben	
<b>Rückgabewert</b>	<b>0</b>	Kein Fehler	
	<b>-22</b>	Das Objekt ist unbekannt	
	<b>-117</b>	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

<b>Definition</b>	GetObjectPatternPath (long long string	IIdent, IType, Titel)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Gibt sämtliche Titel des angegebenen Objekts in einem String zurück. Zum Ermitteln des Pfads zum Objekt wird die Methode 'GetObjectPath' benutzt.	
<b>Parameter</b>	<b>IIdent</b>	ID des Objekts
	<b>IType</b>	Typ des Objekts
	<b>Titel</b>	Sämtliche Überschriften der einzelnen Objekte des Pfads
<b>Rückgabewert</b>	<b>0</b>	Kein Fehler
	<b>-1</b>	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

```

## GetObjectTypImage

<b>Definition</b>	GetObjectTypImage (long long String Variant	lObjectID, lObjectType, strImageFormat, vRetImagePath)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Gibt das Icon eines Objekttyps in Form einer Bilddatei zurück. Die Bilddatei liegt derzeit nur im GIF-Format vor.	
<b>Parameter</b>	<b>lObjectID</b>	ID des Objekts (kann 0 sein, falls es kein Objekt mit Iconkatalog ist)
	<b>lObjectType</b>	Typ des Objekts
	<b>strImageFormat</b>	zurzeit unbenutzt
	<b>vRetImagePath</b>	der vollständige Pfad zur Bilddatei wird hier zurückgegeben
<b>Rückgabewert</b>	<b>0</b>	Bilddatei erfolgreich ermittelt
	<b>-1</b>	Bilddatei konnte nicht ermittelt werden
<b>Bemerkung</b>	Die Fehlertexte können nicht mit GetLastError() ermittelt werden.	
<b>Optionen</b>		
<b>Beispiel</b>		

```
Dim a As Object
Dim vRetImagePath as Variant

Set a = CreateObject("optimal_as.application")

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

## GetRegTypeFromArchive

<b>Definition</b>	GetRegTypeFromArchive (String strSchrankName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt alle zu einem Schrank gehörenden Register mit Namen und Objekttyp
<b>Parameter</b>	<b>strSchrankName</b> String mit dem Schrankbezeichner
<b>Rückgabewert</b>	Semikolon-getrennter String mit Registername, Registertyp.
<b>Bemerkung</b>	Der Rückgabestring hat folgendes Format:

```
REGISTERNAME1, OBJEKTYP1; REGISTERNAME2, OBJEKTYP2
```

### Optionen

### Beispiel

Code-Beispiel:

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

sRet kann z.B. folgenden String enthalten:

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

## GetResourceString

<b>Definition</b>	<pre>Application.GetASSystem.GetResourceString ( String      Key,   String      Language,   String      JSON_Request,   VARIANT     Result_JSON,   VARIANT*    Result_Key,   VARIANT*    Result_Language)</pre>												
<b>Typ</b>	Methode												
<b>Beschreibung</b>	Fragt Sprachressourcen-Schlüssel aus der Ressourcenstorage-Tabelle der Datenbank an.												
<b>Parameter</b>	<table border="0"> <tr> <td style="vertical-align: top;"><b>Key</b></td> <td>Angefragter Schlüssel Beispiel: <code>Project.key1.Workflow_3</code></td> </tr> <tr> <td style="vertical-align: top;"><b>Language</b></td> <td>Sprachkürzel. Beispiel: <code>de_DE</code></td> </tr> <tr> <td style="vertical-align: top;"><b>JSON_Request</b></td> <td>String im JSON-Format, um mehrere Schlüssel/Sprachen anzufragen</td> </tr> <tr> <td style="vertical-align: top;"><b>Result_JSON</b></td> <td>Rückgabe der gefundenen Werte</td> </tr> <tr> <td style="vertical-align: top;"><b>Result_Key</b></td> <td>Rückgabe des gefundenen Schlüssels (wenn nur ein Schlüssel angefragt und gefunden wurde)</td> </tr> <tr> <td style="vertical-align: top;"><b>Result_Language</b></td> <td>Rückgabe des gefundenen Kürzels (wenn nur ein Schlüssel angefragt und gefunden wurde)</td> </tr> </table>	<b>Key</b>	Angefragter Schlüssel Beispiel: <code>Project.key1.Workflow_3</code>	<b>Language</b>	Sprachkürzel. Beispiel: <code>de_DE</code>	<b>JSON_Request</b>	String im JSON-Format, um mehrere Schlüssel/Sprachen anzufragen	<b>Result_JSON</b>	Rückgabe der gefundenen Werte	<b>Result_Key</b>	Rückgabe des gefundenen Schlüssels (wenn nur ein Schlüssel angefragt und gefunden wurde)	<b>Result_Language</b>	Rückgabe des gefundenen Kürzels (wenn nur ein Schlüssel angefragt und gefunden wurde)
<b>Key</b>	Angefragter Schlüssel Beispiel: <code>Project.key1.Workflow_3</code>												
<b>Language</b>	Sprachkürzel. Beispiel: <code>de_DE</code>												
<b>JSON_Request</b>	String im JSON-Format, um mehrere Schlüssel/Sprachen anzufragen												
<b>Result_JSON</b>	Rückgabe der gefundenen Werte												
<b>Result_Key</b>	Rückgabe des gefundenen Schlüssels (wenn nur ein Schlüssel angefragt und gefunden wurde)												
<b>Result_Language</b>	Rückgabe des gefundenen Kürzels (wenn nur ein Schlüssel angefragt und gefunden wurde)												
<b>Rückgabewert</b>	<p>Der Wert des angefragten Schlüssel-Sprachkürzel-Paars, nur wenn die Parameter 'Key' und 'Language' benutzt und nur ein Ergebniswert gefunden wurde.</p> <p>Sonst ist der Rückgabewert ein leerer String und die Ergebnisse sind im Rückgabeparameter 'Result_JSON' gespeichert.</p>												
<b>Bemerkung</b>	<ul style="list-style-type: none"> <li>▪ Parameter können die Platzhalter '*' und '?' enthalten.</li> <li>▪ Der Parameter 'JSON_Request' und die Parameter 'Key/Language' können unabhängig voneinander verwendet werden.</li> <li>▪ Die Parameter 'Key' und 'Language' werden immer getrimmt: Leerzeichen am Anfang und Ende werden entfernt.</li> </ul> <p>Die Beschreibung des JSON-Formats befindet sich im 'Anhang: Ressourcenstorage'.</p>												
<b>Optionen</b>	-												
<b>Beispiel</b>	<pre>Set assystem = Application.GetASSystem value = assystem.GetResourceString("project.test.mykey", "en_US", "", json, key, lang) msgbox "value = " &amp; value &amp; "; key = " &amp; key &amp; "; lang = " &amp; lang &amp; "; json = " &amp; json</pre>												

## GetResultFields

<b>Definition</b>	GetResultFields	(String long long String VARIANT*	(strObjectType, IObjectType, IMode, strDelimiter, pstrFields)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Gibt die vom aktuellen Benutzer für Trefferlisten des angegebenen Typs konfigurierten Felder zurück.		
<b>Parameter</b>	<b>strObjectType</b>	interner Name des Objekttyps (wird nur ausgewertet, wenn ID=-1 übergeben wird)	
	<b>IObjectType</b>	Objekttyp	
	<b>IMode</b>	Art des zu liefernden Ergebnisses	
	<b>strDelimiter</b>	Trennzeichen, das zur Trennung der Feldnamen im Rückgabestring verwendet wird. Wird kein Trennzeichen übergeben, wird per Default ein ‚;‘ verwendet.	
	<b>pstrFields</b>	hier werden die benutzerkonfigurierten Felder in dieser Form zurückgegeben (siehe Bemerkungen):	
	<code>Feld1; Feld2; ...; Feldn</code>		
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-7</b>	Objekttyp unbekannt	
	<b>-40</b>	fehlerhafter Eingabeparameter	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	IMode = 0 gibt die Displaynamen der Felder zurück IMode = 1 gibt die internen Namen der Felder zurück IMode = 2 gibt die OSGUIDs der Felder zurück		
<b>Optionen</b>			
<b>Beispiel</b>			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## GetSignatureProperty

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

## GetSelectedObject

<b>Definition</b>	GetSelectedObject (Variant strSelectedObjects)														
<b>Typ</b>	Methode														
<b>Beschreibung</b>	Gibt die aus der Trefferliste durch <b>SelectObject</b> gewählten Objekte zurück.														
<b>Parameter</b>	<b>strSelectedObjects</b> hier werden die Objekte in dieser Form zurückgegeben: <div style="border: 1px solid black; padding: 2px; width: fit-content; margin-top: 5px;">ObjektID1,Objekttyp1;...;...;ObjektIDn,Objekttypn</div>														
<b>Rückgabewert</b>	<table> <tr> <td><b>3</b></td> <td>Es wurden Objekte ausgewählt, Auswahl ist beendet</td> </tr> <tr> <td><b>2</b></td> <td>Auswahl abgebrochen</td> </tr> <tr> <td><b>1</b></td> <td>noch kein Objekt ausgewählt</td> </tr> <tr> <td><b>0</b></td> <td>SelectObject wurde nicht aufgerufen</td> </tr> <tr> <td><b>-3</b></td> <td>Schrank unbekannt</td> </tr> <tr> <td><b>-5</b></td> <td>Register unbekannt</td> </tr> <tr> <td><b>-26</b></td> <td>Objekt-ID unbekannt</td> </tr> </table>	<b>3</b>	Es wurden Objekte ausgewählt, Auswahl ist beendet	<b>2</b>	Auswahl abgebrochen	<b>1</b>	noch kein Objekt ausgewählt	<b>0</b>	SelectObject wurde nicht aufgerufen	<b>-3</b>	Schrank unbekannt	<b>-5</b>	Register unbekannt	<b>-26</b>	Objekt-ID unbekannt
<b>3</b>	Es wurden Objekte ausgewählt, Auswahl ist beendet														
<b>2</b>	Auswahl abgebrochen														
<b>1</b>	noch kein Objekt ausgewählt														
<b>0</b>	SelectObject wurde nicht aufgerufen														
<b>-3</b>	Schrank unbekannt														
<b>-5</b>	Register unbekannt														
<b>-26</b>	Objekt-ID unbekannt														
<b>Bemerkung</b>	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.														
<b>Optionen</b>															
<b>Beispiel</b>															



## GetSignDocumentResult

<b>Definition</b>	GetSignDocumentResult (Variant vRetSignText)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Gibt das Ergebnis der letzten Digitalen Signatur eines Dokuments zurück.
<b>Parameter</b>	<b>vRetSignText</b> der gewählte Signatortext wird hier zurückgegeben

<b>Rückgabewert</b>	<b>0</b>	Signatur erfolgreich
	<b>1</b>	Signatur abgebrochen
	<b>100</b>	Digitale Signatur wurde noch nicht gestartet
	<b>101</b>	Digitale Signatur läuft noch
	<b>-1</b>	Fehler bei der Initialisierung der Digitalen Signatur
	<b>-2</b>	Es sind keine Signatortexte konfiguriert.
	<b>-77</b>	Signaturmodul ist nicht initialisiert
	<b>-79</b>	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

<b>Definition</b>	GetWDocPattern	(long String String VARIANT* VARIANT*	(ObjectType, strPatternName, strPath, strEditorName, strFileName)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Kopiert eine Vorlagendatei für den gegebenen Objekttyp in das angegebene Verzeichnis		
<b>Parameter</b>	<b>ObjectType</b>	gewünschter Objekttyp, muss W-Dokument sein	
	<b>strPatternName</b>	Alias-Name der Vorlage	
	<b>strPath</b>	Zielpfad	
	<b>strEditorName</b>	hier wird der Name des Bearbeitungsprogramms für diese Vorlage zurückgegeben	
	<b>strFileName</b>	vollständiger Pfad und Name der kopierten Vorlage wird hier zurückgegeben	
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-7</b>	Dokumenttyp unbekannt	
	<b>-35</b>	keine Vorlagen für diesen Typ definiert	
	<b>-36</b>	keine Vorlage mit diesem Namen gefunden	
	<b>-37</b>	Vorlage konnte nicht geholt werden	
	<b>-60</b>	Objekt ist kein W-Dokument	
	<b>-104</b>	Der angegebene Aliasname ist mehrfach vorhanden.	
	<b>-105</b>	Der angegebene Namespace wurde nicht gefunden. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.	
<b>Bemerkung</b>	Ist in der <i>as.cfg</i> der Eintrag STOREATSERVER=1 enthalten, so wird die Vorlagendatei vom Archivserver übertragen, andernfalls wird sie aus Vorlagenverzeichnis kopiert.		
<b>Optionen</b>	Der Name der Vorlage kann auch mit dem gewünschten Namespace getrennt durch ':' angegeben werden.		
<b>Beispiel</b>			

## GetWDocPatternNames

<b>Definition</b>	GetWDocPatternNames	(long String VARIANT*	IObjectType, strToken, strPatternNames)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Liefert alle Alias-Namen der W-Vorlagen, die mit diesem Dokumenttyp verknüpft sind.		
<b>Parameter</b>	<b>IObjectType</b>	gewünschter Objekttyp, muss W-Dokument sein	
	<b>strToken</b>	Trennzeichen, das zur Trennung der Namen im Rückgabestring verwendet werden soll	
	<b>strPatternName</b>	hier werden die Namen zurückgegeben	
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-7</b>	Dokumenttyp unbekannt	
	<b>-35</b>	keine Vorlagen für diesen Typ definiert	
	<b>-60</b>	Objekt ist kein W-Dokument	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

## GoToDocPage

<b>Definition</b>	GoToDocPage	(long lObjectID, long lObjectType, long lPageNum)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Navigiert in einem geöffneten Dokument zur angegebenen Seite.	
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID
	<b>lObjectType</b>	Objekttyp
	<b>lPageNum</b>	Seitennummer
<b>Rückgabewert</b>	<b>0</b>	kein Fehler
	<b>-7</b>	Dokumenttyp nicht bekannt
	<b>-26</b>	Dokument-ID nicht bekannt
	<b>-91</b>	Dokumenttyp wird nicht unterstützt
	<b>-92</b>	Dokument nicht geöffnet
	<b>-93</b>	Dokument hat keine Seiten
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.	
<b>Bemerkung</b>	Die Funktion unterstützt nur Dokumenttypen mit den Haupttypen	
	<b>1</b>	X = Grauwert
	<b>2</b>	D = Schwarz/Weiss
	<b>3</b>	P = Farbe

### — Optionen

#### Beispiel

## InfoWindow

<b>Definition</b>	InfoWindow
<b>Typ</b>	Eigenschaft
<b>Beschreibung</b>	Liefert ein Objekt zur Steuerung des Infofensters.
<b>Parameter</b>	
<b>Rückgabewert</b>	ein Objekt zur Steuerung des Infofensters
<b>Bemerkung</b>	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. <b>Hinweis:</b> '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

<b>Definition</b>	InsertFileList	(short String long* long*	nMainType, strFileName, lReturnObjectID, lReturnObjectType)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Fügt eine oder mehrere Dateien (je nach Haupttyp) als neues Dokument in das Archiv ein.		
<b>Parameter</b>	<b>nMainType</b> <b>strFileName</b>	Haupttyp Datei mit den Namen der einzufügenden Dateien	
	<b>lReturnObjectID</b> <b>lReturnObjectType</b>	hier wird die Objekt-ID zurückgegeben hier wird der Objekttyp zurückgegeben	
<b>Rückgabewert</b>	<b>0</b> kein Fehler <b>-1</b> Einfügen fehlgeschlagen <b>-62</b> Es ist gerade ein Datenblatt geöffnet. <b>-69</b> Übergabedatei ist leer. Der genaue Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	Mögliche Haupttypen sind im Kapitel <b>Einführung</b> beschrieben.  Die Dateinamen in der Übergabedatei müssen durch <b>CR/LF</b> getrennt werden. Die Dateien müssen in dem, mit „ <b>GetEnvironment(0)</b> “ 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 <b>FILEDATA.TXT</b> angegeben, dann können dort Indexdaten für die initiale Indexierung des neu angelegten Dokumentes angegeben werden. (Hinweis: Diese Datei muss im Arbeitsverzeichnis „ <b>GetEnvironment(9)</b> “ liegen und als letzte Datei im Anschluss an die Dokumentdateien in der Übergabedatei eingetragen sein.) In der Datei <b>FILEDATA.TXT</b> können die Indexfelder durch ihren Namen oder ihren internen Namen mit führendem und folgendem '%' -Zeichen angegeben werden. Der Benutzer bekommt im Normalfall die Indexdatenmaske angezeigt und kann die Maske bearbeiten. Ist in dieser Datei jedoch zusätzlich der Parameter <b>ShowNoDialog=1</b> angegeben, dann wird dem Benutzer am enaio® client keine Indexdatenmaske 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 (Schrack 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“ am angegebenen 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.

<b>Definition</b>	InsertFileList	(short String VARIANT* VARIANT*	nMainType, strFileName, varReturnObjectID, varReturnObjectType)
-------------------	----------------	--	--



## InsertIntoArchive

<b>Definition</b>	InsertIntoArchive (String strFilename, long *lpReturnObjectID, long *lpReturnObjectType)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermöglicht das Einfügen eines neuen Ordners in einen Schrank.
<b>Parameter</b>	<b>strFilename</b> Übergabedatei (Aufbau siehe Bemerkung) <b>*lpReturnObjectID</b> hier wird die Objekt-ID zurückgegeben <b>*lpReturnObjectType</b> hier wird der Objekttyp zurückgegeben.
<b>Rückgabewert</b>	<b>0</b> wenn kein Fehler <b>-1</b> Ordner einfügen fehlgeschlagen <b>-30</b> ein oder mehrere Pflichtfelder nicht ausgefüllt <b>-24</b> Feldnamen konnten nicht aufgelöst werden <b>-28</b> Feldwert für ein gegebenes Feld nicht zulässig <b>-31</b> Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein <b>-32</b> Übergabedatei existiert nicht <b>-47</b> Kein Schreibrecht auf dieses Objekt <b>-64</b> Serverfehler ist aufgetreten Der Fehlertext kann mit <b>GetLastError()</b> 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

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

```
[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.

<b>Definition</b>	InsertIntoArchiveS	(String	strFilename,
		VARIANT*	varReturnObjectID,
		VARIANT*	varReturnObjectType)

# InsertIntoDocument

<b>Definition</b>	InsertIntoDocument (String long long	strFilename, *lpReturnObjectID, *lpReturnObjectType)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Ermöglicht das Einfügen eines neuen Dokuments in einen Schrank.	
<b>Parameter</b>	<b>strFilename</b> <b>*lpReturnObjectID</b> <b>*lpReturnObjectType</b>	Übergabedatei (Aufbau s. Bemerkung) Objekt-ID wird zurückgegeben Objekttyp wird zurückgegeben

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

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.

<b>Definition</b>	InsertIntoDocuments (String VARIANT* VARIANT*	strFilename, varReturnObjectID, varReturnObjectType)
-------------------	---	--

# InsertIntoRegister

<b>Definition</b>	InsertIntoRegister	(String long long	strFilename, *lpReturnObjectID, *lpReturnObjectType)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Einfügen eines neuen Registers		
<b>Parameter</b>	<b>strFilename</b>	Übergabedatei (Aufbau siehe Bemerkung)	
	<b>*lpReturnObjectID</b>	hier wird die Objekt-ID zurückgegeben	
	<b>*lpReturnObjectType</b>	hier wird der Objekttyp zurückgegeben.	
<b>Rückgabewert</b>	<b>0</b>	wenn kein Fehler	
	<b>-1</b>	Register einfügen fehlgeschlagen	
	<b>-2</b>	kein Ordner angegeben	
	<b>-3</b>	Schrank unbekannt	
	<b>-4</b>	kein Register angegeben	
	<b>-5</b>	Register unbekannt	
	<b>-8</b>	Register gehört nicht zum Ordner	
	<b>-24</b>	Feldnamen konnten nicht aufgelöst werden	
	<b>-28</b>	Feldwert für ein gegebenes Feld nicht zulässig	
	<b>-30</b>	ein oder mehrere Pflichtfelder nicht ausgefüllt	
	<b>-31</b>	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein	
	<b>-32</b>	Übergabedatei existiert nicht	
	<b>-47</b>	Kein Schreibrecht auf dieses Objekt	
	<b>-64</b>	Serverfehler ist aufgetreten	
	<b>-89</b>	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.

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

**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 eine 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.

<b>Definition</b>	InsertIntoRegisterS	(String VARIANT* VARIANT*	strFilename, varReturnObjectID, varReturnObjectType)
-------------------	---------------------	---------------------------------	--

## InsertNewDMSObject

<b>Definition</b>	InsertNewDMSObject	(long long String VARIANT*	lType, lPosIdent, strFilename, newObjIdent)
-------------------	--------------------	-------------------------------------	--

<b>Typ</b>	Methode
------------	---------

<b>Beschreibung</b>	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  
**Hinweis:** E-Mails werden über die Methode 'ArchiveMailFile' angelegt.

**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 vom angegebenen Typ angelegt werden.
- 123 Eingabe abgebrochen.
- 124 Erforderliches Index-Schreibrecht zur Neuanlage eines Objekts des angegebenen Typs 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 angegebenen 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

<b>Definition</b>	LicLogin(strModulName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Lizenziert das angegebene Modul.
<b>Parameter</b>	<b>strModulName</b> Name des zu lizensierenden Moduls
<b>Rückgabewert</b>	<b>0</b> wenn das Modul lizenziert werden konnte <b>-1</b> allgemeiner Fehler in der Funktion <b>&gt;0</b> Lizenzfehlernummer vom Applikationsserver
<b>Bemerkung</b>	Ist das angegebene Modul bereits für die aktuelle Station lizenziert, wird diese Lizenzierung benutzt.
<b>Optionen</b>	
<b>Beispiel</b>	

## LicLogout

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

## LinkDocuments

<b>Definition</b>	LinkDocuments	(long long long long	(long long long long	lObjectID1, lObjectType1, lObjectID2, lObjectType2)
<b>Typ</b>	Methode			
<b>Beschreibung</b>	Stellt eine Verknüpfung zwischen zwei Dokumenten her.			
<b>Parameter</b>	<b>lObjectID1</b>	ID des ersten Dokuments		
	<b>lObjectType1</b>	Typ des ersten Dokuments		
	<b>lObjectID2</b>	ID des zweiten Dokuments		
	<b>lObjectType2</b>	Typ des zweiten Dokuments		
<b>Rückgabewert</b>	<b>0</b>	kein Fehler		
	<b>-20</b>	einer der Objekttypen ist kein Dokumenttyp, der fehlerhafte Typ ist im Fehlertext angegeben		
	<b>-29</b>	eine oder beide Objekt-IDs sind unbekannt		
	<b>-34</b>	diese Verknüpfung existiert bereits		
	<b>-86</b>	Objektindizes sind identisch		
	<b>-87</b>	Objekte vom Typ Mappe sind nicht zulässig		
	<b>-88</b>	Objekte aus der Ablage sind nicht zulässig		
		Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	Die Verknüpfung zwischen beiden Dokumenten wird in deren Notizen gespeichert.			
<b>Optionen</b>				
<b>Beispiel</b>				

## MergeArchives

<b>Definition</b>	MergeArchives	(long long long	ISourceID, ITargetID, IObjectType)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Vereinigt die Inhalte zweier Ordner.		
<b>Parameter</b>	<b>ISourceID</b>	Quellordner	
	<b>ITargetID</b>	Zielordner	
	<b>IObjectType</b>	Objekttyp	
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-22</b>	Objekttyp unbekannt	
	<b>-47</b>	Kein Recht zum Anlegen von Objekten.	
	<b>-80</b>	Quellordner unbekannt	
	<b>-81</b>	Zielordner unbekannt	
	<b>-82</b>	Aktualisierung der Datenbank fehlgeschlagen	
		Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.	
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

## MoveObject

<b>Definition</b>	MoveObject	(long long long long	(long long long long	lObjectID, lObjectType, lFolderID, lRegisterID)
<b>Typ</b>	Methode			
<b>Beschreibung</b>	Verschiebt ein Objekt (Register oder Dokument).			
<b>Parameter</b>	<b>lObjectID</b>	ID des zu verschiebenden Objektes		
	<b>lObjectType</b>	zugehöriger Objekttyp		
	<b>lFolderID</b>	Ordner-ID des Zielordners.		
	<b>lRegisterID</b>	Register-ID des Zielregisters		
<b>Rückgabewert</b>	<b>0</b>	kein Fehler		
	<b>-5</b>	Registertyp unbekannt (Register-ID gehört zu falschem Objekttyp)		
	<b>-7</b>	Dokumenttyp unbekannt		
	<b>-13</b>	Register-ID unbekannt		
	<b>-14</b>	Ordner-ID unbekannt		
	<b>-29</b>	Objekt-ID ist ungültig		
	<b>-68</b>	Ordner können nicht verschoben werden.		
	<b>-82</b>	Aktualisierung der Datenbank fehlgeschlagen		
	<b>-90</b>	Das angegebene Objekt ist eine Verweiskopie		
	<b>539</b>	Objekttyprelationsverletzung		
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.			
<b>Bemerkung</b>	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.			
<b>Optionen</b>				
<b>Beispiel</b>				

## OleDdeRequest

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

## OpenAboDialog

<b>Definition</b>	OpenAboDialog	(long long long long String	IHwnd, IObjectID, IObjectType, IFlags, strInfoText)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Zeigt den Abonnement-Dialog an.		
<b>Parameter</b>	<b>IHwnd</b>	Fensterhandle	
	<b>IObjectID</b>	ID des Dokuments	
	<b>IObjectType</b>	Dokumenttyp	
	<b>IFlags</b>	siehe Bemerkung	
	<b>strInfoText</b>	Infotext, wird in das Info-Feld des Dialogs eingetragen	
<b>Rückgabewert</b>	<b>1</b>	Abbruch des Dialogs durch den Benutzer	
	<b>0</b>	kein Fehler	
	<b>-7</b>	Objekttyp unbekannt	
	<b>-26</b>	Objekt-ID ist unbekannt	
	<b>-62</b>	Es ist bereits ein Dialog geöffnet	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	Durch Angabe von Flags können bestimmte Checkboxes des Dialogs gesetzt werden. Die Flags können kombiniert werden.		
	<b>1</b>	Checkbox Dokument geändert	
	<b>2</b>	Checkbox Indexdaten geändert	
	<b>4</b>	Checkbox Objekt angelegt	
	<b>8</b>	Checkbox Objekt gelöscht	
<b>Optionen</b>			
<b>Beispiel</b>			

## OpenBrowser

<b>Definition</b>	OpenBrowser	(String String String	URL, CAPTION, GUID)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Öffnet ein Browser-Fenster im Client.		
<b>Parameter</b>	<b>URL</b>	Die URL, die das Browser-Fenster anzeigen soll	
	<b>CAPTION</b>	Der Fenstertitel des Browser-Fensters	
	<b>GUID</b>	Die eindeutige ID des Browser-Fensters	
<b>Rückgabewert</b>	<b>0</b>	Browser-Fenster geöffnet	
	<b>-1</b>	Fehler	
<b>Bemerkung</b>	<p>Es werden nur URLs angezeigt, die mit 'http://', 'https://' oder 'file://' beginnen.</p> <p>Klickt der Anwender innerhalb der angezeigten HTML-Seite auf einen Link, dann wird dieser in einem externen Browser angezeigt.</p> <p>Aus dem HTML kann auf ein JavaScript-Objekt zugegriffen werden.</p> <p>Implementierte JavaScript -Objekt-Methoden:  <code>osjxGetBrowserEnvironment</code>: Ermittelt Eigenschaften des Clients.</p> <p>Implementierte JavaScript -Objekt-Eigenschaften:  <code>osjxSetBrowserCaption</code>: Setzt den Fenstertitel des Browser Fensters.</p> <p>Weitree <code>osjx</code>-Funktionen stehen nicht zur Verfügung.</p>		

### Beispiel

JavaScript-Beispiele:

```
function button1_script()
{
    if (window.osClient)
    {
        var sessionguid =
window.osClient.osjxGetBrowserEnvironment(27);

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

function button2_script()
{
    if (window.osClient)
    {
        window.osClient.osjxSetBrowserCaption =
"Yes we do";
    }
    else
    {
        alert("window.osClient not exist");
    }
}
```

## OpenDataDlg

<b>Definition</b>	OpenDataDlg	(long long short	lObjectID, lObjectType, nMode)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Öffnet das Datenblatt des angegebenen Objekts.		
<b>Parameter</b>	<b>lObjectID</b> <b>lObjectType</b> <b>nMode</b>	Objekt-ID Objekttyp Bearbeitungsmodus (siehe Bemerkung)	
<b>Rückgabewert</b>	0 -7 -11	kein Fehler Objekt unbekannt Dokumentkennung nicht zulässig (= 0)	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	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



## OpenModalBrowserDialog

<b>Definition</b>	OpenModalBrowserDialog (String OSFILE)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Öffnet das ModalDialog-AddOn via COM-Schnittstelle in einem clientseitigen Event.
<b>Parameter</b>	<b>OSFILE</b> Ini-Datei mit den Aufrufparametern des AddOns
<b>Bemerkung</b>	<p>Das ModalDialog-AddOn stellt einen Dialog mit einem Browsercontrol zur Verfügung, in dem kundenindividuelle und frei gestaltbare modale Dialoge realisiert werden können.</p> <p>Die projektseitige Implementierung erfolgt als Web-Applikation auf Basis von HTML und JavaScript, die im Gateway oder im Service-Manager gehostet werden kann.</p> <p>Über die gemeinsame <a href="#">ModalDialog API</a> ist gewährleistet, dass der HTML/JS-Code sowohl in enaio® client, als auch in enaio® webclient lauffähig ist.</p> <p>Das ModalDialog-AddOn kann sowohl auf DMS-Masken als auch auf Workflow-Masken verwendet werden.</p> <p>Es bietet die Möglichkeit, alle Daten des clientseitigen Datenblattes in das AddOn zu laden, zu visualisieren, um ggfs. Daten aus anderen Quellen dazu zu spiegeln.</p> <p>Beim Schließen des Dialoges aus dem AddOn können beliebige Datenfelder (oder Workflow-Variablen) per SK mit neuen Daten in das Datenblatt zurückübertragen werden.</p>

### Beispiel

Skriptbeispiel DMS:

```
Dim ret : ret = 0
' Eingabe der URL
OXHelp.WriteProfString "DATA", "EXTRA00" ,
"https://optimalsystems.github.io/enaio-webclient-demo-modal-
dialog-addon/index.html", osFile
' Übergabe des Fenstertitels
OXHelp.WriteProfString "DATA", "EXTRA01" , "ModalDialog API
Test Addon", osFile
ret = Application.OpenModalBrowserDialog(osFile)
resultcode = ret
writetofile
```

Skriptbeispiel Workflow:

```
Dim ret : ret = 0
Dim osFile : osFile = ""

osFile = workitem.ExportToFile

OXHelp.WriteProfString "DATA", "EXTRA00" ,
"https://optimalsystems.github.io/enaio-webclient-demo-modal-
dialog-addon/index.html", osFile
' Übergabe des Fenstertitels
OXHelp.WriteProfString "DATA", "EXTRA01" , "ModalDialog API
Test Addon", osFile

ret = Application.OpenModalBrowserDialog(osFile)

scriptresult.ResultCode = 1

if ret = 1 then
scriptresult.ResultCode = 0
workitem.ImportFromFile(osFile)
End if

workitem.CleanUpFile(osFile)
```

## OpenObjectID

<b>Definition</b>	OpenObjectID (long long short	lObjectID, lObjectType, nMode)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	<p>Öffnet das angegebene Objekt.</p> <p>Ist das Objekt ein <b>Dokument</b>, dann wird das Image angezeigt bzw. das entsprechende Dokument geladen(W-Dokument). Hat das Dokument keine Images, wird das Datenblatt geöffnet.</p> <p>Ist das Objekt ein <b>Ordner</b> bzw. <b>Register</b>, dann wird der entsprechende Ordner geöffnet.</p>	
<b>Parameter</b>	<b>lObjectID</b> <b>lObjectType</b> <b>nMode</b>	ID des zu öffnenden Objekts Objekttyp Bewirkt bei zu öffnenden W-Dokumenten(und nur da), ob das Dokument schreibgeschützt geöffnet wird oder zur Bearbeitung. Ist dieser Parameter <b>0</b> , dann wird das Dokument wie bisher mit Schreibe Schutz geöffnet. Ist der Modus = <b>1</b> , dann wird das Dokument zur Bearbeitung geöffnet. <b>nMode = 2</b> , Dokument wird schreibgeschützt und es wird nicht die aktive Variante geöffnet, falls das Dokument der Variantenverwaltung unterliegt, sondern genau das Dokument mit angegebenen ID <b>nMode = 3</b> , 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
<b>Rückgabewert</b>	<b>0</b> <b>-11</b>	kein Fehler wenn lObjectID = 0 Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	Die Anzeige erfolgt durch den enaio® client.	
<b>Optionen</b>		
<b>Beispiel</b>	Ein W-Dokument soll mit Schreibe Schutz geöffnet werden: <pre style="border: 1px solid black; padding: 2px;">intRet = MyAX.OpenObjectID(lngID, lngType, 0)</pre>	

## OpenResultList

<b>Definition</b>	OpenResultList	(String String	strIDList, strTitle)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Öffnet eine Trefferliste mit den übergebenen Objekten.		
<b>Parameter</b>	<b>strIDList</b>	Liste mit den Objekten in der Form: ID,Objekttyp;ID1,Objekttyp1...	
	<b>strTitle</b>	Fenstertitel	
<b>Rückgabewert</b>	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

<b>Definition</b>	OpenObjectIDEx	(long lObjectID, long lObjectType, BOOL bWriteProtected, BOOL bActivateIfOpen)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	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.	
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID
	<b>lObjectType</b>	Objekttyp
	<b>bWriteProtected</b>	TRUE, wenn das Dokument schreibgeschützt geöffnet werden soll
	<b>bActivateIfOpen</b>	TRUE, wenn ein bereits geöffnetes Fenster dieses Typs aktiviert werden soll.
<b>Rückgabewert</b>	0	kein Fehler
	-11	Objekt-ID ungültig
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.	
<b>Bemerkung</b>		
— <b>Optionen</b>		
<b>Beispiel</b>		

## OpenURL

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

## OpenWorkItem

<b>Definition</b>	OpenWorkItem (String strWorkItemID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Öffnet einen Vorgangsschritt im Eingangskorb des angemeldeten Benutzers.
<b>Parameter</b>	<b>strWorkItemID</b> Id des zu öffnenden Vorgangsschrittes.
<b>Rückgabewert</b>	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 <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	Diese Funktion öffnet ausschließlich Vorgangsschritte, die im Eingangskorb des aktuell angemeldeten Benutzers liegen.
<b>Optionen</b>	
<b>Beispiel</b>	

## PrintDocumentID

**Definition** PrintDocumentID (String strParam)

**Typ** Methode

**Beschreibung** Druckt die angegebenen Objekte, wenn die Objekte vom Typ 'Dokument' sind.

**Parameter** **strParam** String mit folgendem Inhalt:

```
INDEX, OBJEKTYP
```

**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, OBJEKTYP ... INDEXn, OBJEKTYP
```

Die einzelnen Objektangaben trennt ein Leerzeichen.

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

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

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

**Optionen**

**Beispiel**



## RefreshFolderWindow

<b>Definition</b>	RefreshFolderWindow	(long long short	lObjectID, lObjectType, lObjectID2Select)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	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.		
<b>Parameter</b>	<b>lObjectID</b> <b>lObjectType</b> <b>lObjectID2Select</b>	Objekt-ID Objekttyp Neu zu selektierendes Objekt (0 wenn alte Selektion erhalten bleiben soll)	
<b>Rückgabewert</b>			
<b>Bemerkung</b>	<p>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 Clients ist.</p> <p>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.</p>		
<b>Optionen</b>			
<b>Beispiel</b>			

## ScanDocument

<b>Definition</b>	ScanDocument	(long long short	lObjectID, lObjectType, nMode)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ermöglicht das Anfügen von Images an ein bestehendes Dokument.		
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID	
	<b>lObjectType</b>	Objekttyp	
	<b>nMode</b>	Modus (siehe Bemerkung)	
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-11</b>	Dokumentkennung nicht zulässig	
	<b>-19</b>	Objekt ist keinem Ordner zugeordnet	
	<b>-20</b>	Angegebener Objekttyp ist kein Dokumenttyp!	
	<b>-21</b>	Dokument bereits archiviert	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden		
<b>Bemerkung</b>	Hat nMode den Wert 1, dann wird vor dem Öffnen des Scan-Fensters die Indexdatenmaske 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.		
<b>Optionen</b>			
<b>Beispiel</b>			

# ShowVariantsDialog

<b>Definition</b>	ShowVariantsDialog	(long long BOOL BOOL BOOL BOOL long VARIANT	(lObjectID, lObjectType, bCreateSingleVariant, bAllowDragDrop, bOpenAfterCreation, bSetNewVersionActive, lHwnd, vRetNewObjectID)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Zeigt den Dialog der Variantenverwaltung an.		
<b>Parameter</b>	<b>lObjectID</b> <b>lObjectType</b> <b>bCreateSingleVariant</b>  <b>bAllowDragDrop</b>  <b>bOpenAfterCreation</b>  <b>bSetNewVersionActive</b>  <b>lHwnd</b> <b>vRetNewObjectID</b>		ID des Dokuments Dokumenttyp gibt an, ob der Benutzer nur eine Variante anlegen darf gibt an, ob Drag&Drop in diesem Dialog erlaubt ist gibt an, ob die neu angelegte Variante gleich geöffnet werden soll gibt an, ob die neu angelegte Variante als aktiv markiert wird Fensterhandle hier werden die ID's der neu angelegten Varianten durch Semikolon getrennt zurückgegeben
<b>Rückgabewert</b>	1 0 -7 -26 -60 -62 -93	Abbruch des Dialogs durch den Benutzer kein Fehler Objekttyp unbekannt Objekt-ID ist unbekannt Objekt ist nicht vom Typ W-Dokument Es ist bereits ein Dialog geöffnet Objekt hat keine Seiten	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## SelectObject

<b>Definition</b>	SelectObject	(long long BOOL String	lObjectID, lObjectType, bMultiSelect strTitle)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Öffnet eine Trefferliste zur Auswahl eines Objekts.		
<b>Parameter</b>	<b>lObjectID</b>	ID des Ordners oder Registers	
	<b>lObjectType</b>	Objekttyp des Ordners oder Registers	
	<b>bMultiSelect</b>	<b>TRUE</b> , für Mehrfachselektion	
	<b>strTitle</b>	Fenstertitel	
<b>Rückgabewert</b>	<b>1</b>	Auswahl wurde gestartet	
	<b>-25</b>	Objekt ist kein Ordner oder Register	
	<b>-26</b>	Objekt-ID nicht zulässig	
	<b>-27</b>	Es ist bereits eine Trefferliste zur Objektauswahl geöffnet	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	Die geöffnete Trefferliste besitzt zwei neue Buttons auf der Toolbar zur Auswahl der selektierten Objekte oder zum Abbruch der Auswahl.		
<b>Optionen</b>			
<b>Beispiel</b>			

## SendMail

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

# 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.

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

**Optionen**

**Beispiel**

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## SetBrowserURL

<b>Definition</b>	SetBrowserURL (String String GUID URL)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Lädt eine neue URL in ein geöffnetes Browser-Fenster. Wird ein per GUID spezifiziertes Browserfenster mit der gleichen URL belegt, dann wird das Fenster neu initialisiert (vgl. OpenBrowser).
<b>Parameter</b>	<b>GUID</b> Die eindeutige ID des Browser-Fensters
<b>Rückgabewert</b>	<b>0</b> Browser-Fenster zeigt die URL an <b>-1</b> Fehler
<b>Bemerkung</b>	Es gelten dieselben Einschränkungen wie bei OpenBrowser.
<b>Beispiel</b>	<pre>Dim ax, guid, res set ax = CreateObject("optimal_AS.Application")  ax.ActivateApp(1)  MsgBox("click to open 'https://html5test.com'")  res = ax.OpenBrowser("https://html.com", "Test", guid)  if res = 0 then     MsgBox("click to open URL 'https://www.os.de'")     res = ax.SetBrowserURL(guid, "https://www.os.de")     MsgBox("click to close browser")     res = ax.CloseBrowser(guid) else     sRes = ax.GetLastError()     MsgBox(sRes) end if  set ax = nothing</pre>

## SetPlannedRetention

<b>Definition</b>	SetPlannedRetention (long long String	lObjectID, lObjectType, strRetentionDate)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Setzt das geplante Retention-Datum für ein zu archivierendes Dokument.	
<b>Parameter</b>	<b>lObjectID</b> <b>lObjectType</b> <b>strRetentionDate</b>	ID des Objekts Typ des Objekts das Retention-Datum im Format DD.MM.YYYY
<b>Rückgabewert</b>	0 -1 -7 -22 -38 -40	Retention-Datum erfolgreich gesetzt Retention-Datum konnte nicht gesetzt werden Objekttyp unbekannt Objekt mit angegebener ID existiert nicht Ungültiger Dokumenttyp (es wurde ein Ordner oder Registertyp angegeben) Ungültiger Parameter wurde übergeben
<b>Bemerkung</b>	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

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

## SetResourceString

<b>Definition</b>	Application.GetASSystem.SetResourceString ( String Key, String Language, String Value, String JSON_Request )
<b>Typ</b>	Methode
<b>Beschreibung</b>	Schreibt Sprachressourcen-Schlüssel mit den zugeordneten Werten für Sprachen in die Ressourcenstorage-Tabelle der Datenbank.
<b>Parameter</b>	<p><b>Key</b> Schlüssel, der eingefügt wird. Beispiel: <code>Project.key1.Workflow_3</code></p> <p><b>Language</b> Sprachkürzel für den Eintrag. Beispiel: <code>de_DE</code></p> <p><b>Value</b> Sprachspezifischer Wert für den Schlüssel.</p> <p><b>JSON_Request</b> String im JSON-Format, um mehrere Schlüssel, Sprachen und Werte gleichzeitig zu schreiben.</p>
<b>Rückgabewert</b>	0 kein Fehler
<b>Bemerkung</b>	<ul style="list-style-type: none"> <li>▪ Parameter sind immer ohne Platzhalter.</li> <li>▪ Der Parameter 'JSON' und die Parameter 'Key/Language' können unabhängig voneinander verwendet werden.</li> <li>▪ Die Parameter 'Key' und 'Language' werden immer getrimmt: Leerzeichen am Anfang und Ende werden entfernt.</li> <li>▪ Bei dem Parameter 'Value' werden Leerzeichen nur am Ende entfernt.</li> </ul> <p>Die Beschreibung des JSON-Formats befindet sich im 'Anhang: Ressourcenstorage'.</p>
<b>Optionen</b>	-
<b>Beispiel</b>	<pre>Set assystem = Application.GetASSystem value = assystem.SetResourceString("Project. key1.Workflow_3", "en_US", "Bill", "")</pre>

# SetSignatureProperty

<b>Definition</b>	SetSignatureProperty (String strPropertyName, String strValue)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Setzt eine Eigenschaft für die digitale Signatur eines Dokuments
<b>Parameter</b>	<b>strPropertyName</b> Name der Eigenschaft <b>strValue</b> 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
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## SetWaitingCursor

<b>Definition</b>	SetWaitingCursor (short sShow)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Zeigt den Mauszeiger als Sanduhr.
<b>Parameter</b>	<b>sShow</b> Sanduhr einschalten (1) und wieder ausschalten (0).
<b>Rückgabewert</b>	Kein Rückgabewert
<b>Bemerkung</b>	Muss am Skriptende wieder ausgeschaltet werden.
<b>Optionen</b>	-

**Beispiel**

```
SetWaitingCursor(1)

Dim dteWait

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

Do Until (Now() > dteWait)
Loop

SetWaitingCursor(0)

ResultCode=1
WriteToFile
```

## SignDocument

<b>Definition</b>	SignDocument (long long      IObjectID, IObjectType)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ruft den Dialog zur Digitalen Signatur eines Dokuments auf.
<b>Parameter</b>	<b>IObjectID</b> ID des zu signierenden Dokuments <b>IObjectType</b> Typ des zu signierenden Dokuments
<b>Rückgabewert</b>	Kein Rückgabewert
<b>Bemerkung</b>	Die Funktion ruft die Signatur in einem eigenen Thread auf. Um den Rückgabewert zu erhalten kann die Funktion GetSignDocumentResult benutzt werden.
<b>Optionen</b>	-

### 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

<b>Definition</b>	SignDocumentEx	(long long BOOL Variant	(ObjectID, ObjectType, bWaitForCompletion, vRetSignText)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ruft den Dialog zur Digitalen Signatur eines Dokuments auf.		
<b>Parameter</b>	<b>ObjectID</b> <b>ObjectType</b> <b>bWaitForCompletion</b> <b>vRetSignText</b>	ID des zu signierenden Dokuments Typ des zu signierenden Dokuments gibt an, ob die Funktion auf die Beendigung der Signatur warten soll. 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
    
```

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

## StartArchiveRequest

<b>Definition</b>	StartArchiveRequest (String strFileName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Startet eine Schrankanfrage
<b>Parameter</b>	<b>strFileName</b> Name der Anfragedatei
<b>Rückgabewert</b>	<i>Dateiname</i> oder <b>READY</b> , wenn ANFRAGEFENSTER <> 0. <b>Leerer String</b> , wenn ein Fehler aufgetreten ist. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	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 zurückzusetzen, um eine korrekte Ausgabe zu gewährleisten. Die Rückgabedatei hat folgenden Aufbau:

```
INDEX1, OBJEKTYP
INDEX2, OBJEKTYP
...
...
INDEXn, OBJEKTYP
```

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><LF>
```

### 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

<b>Definition</b>	StartDocRequest (String strFileName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Startet eine Dokumentanfrage.
<b>Parameter</b>	<b>strFileName</b> Name der Anfragedatei
<b>Rückgabewert</b>	<i>Dateiname</i> oder <b>READY</b> , wenn ANFRAGEFENSTER <> 0 Leerer String, wenn ein Fehler aufgetreten ist. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	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 | Ordnetrefferliste    |
| 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

<b>Definition</b>	StartRegRequest (String strFileName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Startet eine Registeranfrage.
<b>Parameter</b>	<b>strFileName</b> Name der Anfragedatei
<b>Rückgabewert</b>	<b>Dateiname</b> oder <b>READY</b> , wenn ANFRAGEFENSTER <> 0 <b>Leerer String</b> , wenn ein Fehler aufgetreten ist. Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.
<b>Bemerkung</b>	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><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:**

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

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

<b>Definition</b>	StartClientRequest (string request)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Startet die formulierte Client-Anfrage. Volltexttrefferlisten werden mit Facetten angezeigt.
<b>Parameter</b>	<b>request</b> Anfrage im JSON-Format
<b>Bemerkung</b>	Beschreibung JSON <b>request</b> Startobjekt <b>objecttype</b> UINT, Objekttyp der Anfrage <b>fulltext</b> String, Volltextanfragestring <b>fields</b> Array, JSON Elemente vom Typ 'field' <b>field</b> besteht aus: <b>iname</b> String, interner Name des anzufragenden Feldes <b>value</b> String, Wert des anzufragenden Feldes
<b>Rückgabewert</b>	<b>0</b> Anfrage ausgeführt <b>-125</b> Ein modaler Dialog ist geöffnet. <b>-140</b> JSON hat keinen Member 'request' <b>-141</b> Member 'request' hat keinen Member 'objecttype' <b>-143</b> Member 'objecttype' hat unerwarteten Datentyp <b>-144</b> Member 'objecttype' ist ungültig <b>-145</b> Angegebener Objekttyp ist unbekannt <b>-146</b> Angegebener Objekttyp ist nicht im Volltextindex <b>-147</b> Member 'fulltext' hat unerwarteten Datentyp <b>-148</b> Member 'fulltext' hat keinen Wert <b>-149</b> JSON enthält Parserfehler <b>-150</b> Member 'fields' ist kein Array <b>-151</b> Array-Member 'iname' oder 'value' fehlt <b>-152</b> Array-Member 'iname' oder 'value' hat keinen Wert <b>-153</b> 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 = nothing
get Application = nothing
```

## StoreNotice

<b>Definition</b>	StoreNotice	(long long String	lObjectID, lObjectType, strFileName)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Speichert eine Notiz zu einem existierenden Objekt (Ordner, Register, Dokument).		
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID	
	<b>lObjectType</b>	Objekttyp	
	<b>strFileName</b>	Dateiname der Notiz mit Endung 'txt'	
<b>Rückgabewert</b>	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 <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>	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 <b>strFileName</b> 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.		
<b>Optionen</b>			
<b>Beispiel</b>			

## TransformXML

<b>Definition</b>	TransformXML	(String String VARIANT	strXMLFile, strXSLFile, varRetString)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Wandelt eine XML-Datei mit Hilfe der XSL-Datei in ein gewünschtes Zielformat um.		
<b>Parameter</b>	<b>strXMLFile</b>	vollständiger Pfad und Dateiname der XML-Datei	
	<b>strXSLFile</b>	vollständiger Pfad und Dateiname der XSL-Datei	
	<b>varRetString</b>	das transformierte Objekt wird hier in Form eines Strings zurückgegeben.	
<b>Rückgabewert</b>	<b>0</b>	kein Fehler	
	<b>-32</b>	eine der beiden Quelldateien existiert nicht	
	<b>-100</b>	Dateien konnten nicht umgewandelt werden	
	<b>-101</b>	XML-Datei konnte nicht geladen werden	
	<b>-102</b>	XSL-Datei konnte nicht geladen werden	
	<b>-103</b>	XML-Objekt konnte nicht erstellt werden	
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.		
<b>Bemerkung</b>			
<b>Optionen</b>			
<b>Beispiel</b>			

## UndoCheckOut

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



## UnlinkDocuments

<b>Definition</b>	UnlinkDocuments	(long lObjectID1, long lObjectType1, long lObjectID2, long lObjectType2 )
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Löscht eine Verknüpfung zwischen zwei Dokumenten.	
<b>Parameter</b>	<b>lObjectID1</b>	ID des ersten Dokuments
	<b>lObjectType1</b>	Typ des ersten Dokuments
	<b>lObjectID2</b>	ID des zweiten Dokuments
	<b>lObjectType2</b>	Type des zweiten Dokuments
<b>Rückgabewert</b>	<b>0</b>	kein Fehler
	<b>-29</b>	eine oder beide Objekt-IDs sind unbekannt
	<b>-86</b>	Objektindizes sind identisch
	<b>-88</b>	Objekte aus der Ablage sind nicht zulässig
	Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.	
<b>Bemerkung</b>	Funktion steht ab 4.00 SPII zur Verfügung. Die Verknüpfung zwischen beiden Dokumenten wird in deren Notizen gespeichert.	
<b>Optionen</b>		
<b>Beispiel</b>		

# UpdateArchiveData

<b>Definition</b>	UpdateArchiveData (String long long	strFilename, *lpReturnObjectID, *lpReturnObjectType)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Führt eine Aktualisierung von Ordnerdaten aus.	
<b>Parameter</b>	<b>strFilename</b>	Übergabedatei (Aufbau siehe Bemerkung)
	<b>*lpReturnObjectID</b>	hier wird bei erfolgreichem Einfügen die Objekt-ID zurückgegeben
	<b>*lpReturnObjectType</b>	hier wird bei erfolgreichem Einfügen der Objekttyp zurückgegeben.
<b>Rückgabewert</b>	<b>0</b>	wenn kein Fehler
	<b>-3</b>	Schrank unbekannt
	<b>-10</b>	Ordnerkennung fehlt
	<b>-14</b>	Ordnerkennung unbekannt
	<b>-18</b>	Update fehlgeschlagen
	<b>-24</b>	Feldnamen konnten nicht aufgelöst werden
	<b>-28</b>	Feldwert für ein gegebenes Feld nicht zulässig
	<b>-30</b>	ein oder mehrere Pflichtfelder nicht ausgefüllt
	<b>-31</b>	Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein
	<b>-32</b>	Übergabedatei existiert nicht
	<b>-47</b>	Kein Schreibrecht auf dieses Objekt
	<b>-64</b>	Serverfehler ist aufgetreten
	Der Fehlertext kann mit <b>GetLastError()</b> 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]

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

Für den Wert 1 wird eine Überprüfung der Pflichtfelder vorgenommen.

### Beispiel

#### Aktualisierung eines Kundenorders

```
[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

<b>Definition</b>	UpdateDocFileList (String strFileName)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermöglicht eine Änderung, der Dateiliste eines Dokuments.
<b>Parameter</b>	<b>strFileName</b> Dateiname, Aufbau siehe Bemerkung
<b>Rückgabewert</b>	<p>0 kein Fehler</p> <p>-2 Keinen Schrank angegeben</p> <p>-3 Schrank unbekannt</p> <p>-6 Kein Dokumenttyp angegeben</p> <p>-7 Dokumenttyp unbekannt</p> <p>-9 Dokumenttyp gehört nicht zum angegebenen Schrank</p> <p>-11 Dokumentkennung fehlt</p> <p>-21 Dokument bereits archiviert</p> <p>-32 Übergabedatei existiert nicht</p> <p>-41 Kein Dateiname angegeben</p> <p>-64 Serverfehler ist aufgetreten</p> <p>Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.</p>

**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.

<b>Optionen</b>	<p><b>ARCHIVIERBAR=[0;1]</b> Hiermit kann der Archivstatus des Dokumentes geändert werden.</p> <p>0 nicht archivierbar</p> <p>1 archivierbar</p> <p>1 ist Standard</p> <p><b>DATEILÖSCHEN=[0;1]</b> Löscht die angegebenen Quell-Dateien.</p> <p>0 ist Standard</p>
-----------------	---

**Beispiel** Codebeispiel:

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

HIER LOCHEN ODER DIGITAL ARCHIVIEREN

**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

<b>Definition</b>	UpdateDocShare	(long long long	(IIdent, IType, IUser)
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Bearbeitet die Freigabe des angegebenen Dokuments für den angegebenen Benutzer		
<b>Parameter</b>	<b>IIdent</b>	Dokument-ID	
	<b>IType</b>	Dokumenttyp	
	<b>Rights</b>	Rechte, die vorbelegt werden (RWXU)	
	<b>IUser</b>	Benutzer-ID des Benutzers der Freigabe	
<b>Rückgabewert</b>	<b>0</b>	"Freigabe bearbeitet oder Bearbeitung abgebrochen."	
	<b>-125</b>	"Ein modaler Dialog ist geöffnet."	
	<b>-130</b>	"Angegebener DMS-Objekttyp unbekannt."	
	<b>-131</b>	"Angegebener DMS-Objekttyp kein Dokument."	
	<b>-132</b>	"Nur Rechte 'RXWU' sind zulässig."	
	<b>-133</b>	"Angegebenes Dokument existiert nicht oder wurde gelöscht."	
	<b>-134</b>	"Dokumentfreigaben nicht möglich."	
	<b>-135</b>	"Die erforderliche Systemrolle, um Dokumente freizugeben, ist nicht vorhanden."	
	<b>-136</b>	"Angegebener Benutzer existiert nicht."	
	<b>-139</b>	"Angegebenes Objekt wurde vom 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 abgebrochen."
end if

set Application = nothing

```

## UpdateDocumentData

<b>Definition</b>	UpdateDocumentData (String strFilename)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Führt eine Aktualisierung von Dokumentdaten aus.
<b>Parameter</b>	<b>strFilename</b> Übergabedatei (Aufbau siehe Bemerkung)
<b>Rückgabewert</b>	<ul style="list-style-type: none"> <li>0 wenn kein Fehler</li> <li>-2 kein Ordner angegeben</li> <li>-3 Ordner unbekannt</li> <li>-4 kein Register angegeben</li> <li>-5 Register unbekannt</li> <li>-7 Unbekannter Dokumenttyp</li> <li>-8 Register gehört nicht zum angegebenen Ordner</li> <li>-11 Dokumentkennung fehlt</li> <li>-13 Registerkennung unbekannt</li> <li>-16 Update fehlgeschlagen</li> <li>-24 Feldnamen konnten nicht aufgelöst werden</li> <li>-28 Feldwert für ein gegebenes Feld nicht zulässig</li> <li>-30 ein oder mehrere Pflichtfelder nicht ausgefüllt</li> <li>-31 Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein</li> <li>-32 Übergabedatei existiert nicht</li> <li>-47 Kein Schreibrecht auf dieses Objekt</li> <li>-64 Serverfehler ist aufgetreten</li> </ul> Der Fehlertext kann mit <b>GetLastError()</b> ermittelt werden.

**Bemerkung** Die Übergabedatei hat folgenden Aufbau:

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

Ist Mode = 1, werden alle nicht angegebenen 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
DATA2=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

<b>Definition</b>	UpdateRegisterData (String strFilename)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Führt eine Aktualisierung von Registerdaten aus.
<b>Parameter</b>	<b>strFilename</b> Übergabedatei (Aufbau siehe Bemerkung)
<b>Rückgabewert</b>	<p>0 wenn kein Fehler</p> <p>-2 kein Ordner angegeben</p> <p>-3 Ordner unbekannt</p> <p>-4 kein Register angegeben</p> <p>-5 Register unbekannt</p> <p>-8 Register gehört nicht zum angegebenen Ordner</p> <p>-12 Registerkennung fehlt</p> <p>-13 Registerkennung unbekannt</p> <p>-16 Update fehlgeschlagen</p> <p>-24 Feldnamen konnten nicht aufgelöst werden</p> <p>-28 Feldwert für ein gegebenes Feld nicht zulässig</p> <p>-30 ein oder mehrere Pflichtfelder nicht ausgefüllt</p> <p>-31 Datentyp des einzufügenden Wertes stimmt nicht mit dem Datentyp des dazugehörigen Feldes überein</p> <p>-32 Übergabedatei existiert nicht</p> <p>-47 Kein Schreibrecht auf dieses Objekt</p> <p>-64 Serverfehler ist aufgetreten</p>

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 angegebenen 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

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

**Beispiel**

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

## Visible

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

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

## URL

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

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

## Closeable

<b>Definition</b>	Boolean Closeable
<b>Typ</b>	Eigenschaft (lesen/schreiben)
<b>Beschreibung</b>	Legt fest, gibt an, ob das Fenster vom Benutzer geschlossen werden kann.
<b>Parameter</b>	-
<b>Rückgabewert</b>	-
<b>Beispiel</b>	<pre>Dim a as object Set a = createobject("optimal_as.application") a.InfoWindow.Closeable = false</pre>

## HtmlDocument

<b>Definition</b>	Object HtmlDocument
<b>Typ</b>	Eigenschaft (lesen)
<b>Beschreibung</b>	Liefert das HTML-DOM-Document, welches gerade angezeigt wird.
<b>Parameter</b>	-
<b>Rückgabewert</b>	-
<b>Bemerkung</b>	Die HTML-Document Schnittstelle ist in der MSDN beschrieben.
<b>Beispiel</b>	<pre>Dim a as object Set a = createobject("optimal_as.application") Set doc = a.InfoWindow.HtmlDocument</pre>

## EnableContextMenu

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

**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

<b>Definition</b>	GoForward()
<b>Typ</b>	Methode
<b>Beschreibung</b>	Navigiert in der Historie zur folgenden URL.
<b>Parameter</b>	-
<b>Rückgabewert</b>	-

**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

<b>Definition</b>	ShowHtml(String strHtml)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Zeigt den übergebenen HTML-String an.
<b>Parameter</b>	-
<b>Rückgabewert</b>	-

**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

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

**osjxCanPrevDoc**

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

## osjxNextDoc

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

## osjxPrevDoc

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

## Funktionen zur Fenstersteuerung des Clients

### osjxOpenDataSheet

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

### osjxOpenObject

<b>Definition</b>	osjxOpenObject (long lObjectID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Öffnet das Dokument oder den Ordner eines Objekts.
<b>Parameter</b>	<b>lObjectID</b> 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

<b>Definition</b>	osjxPrintObject()
<b>Typ</b>	Methode
<b>Beschreibung</b>	Druckt das Dokument aus.
<b>Parameter</b>	-
<b>Rückgabewert</b>	-

## osjxOpenObjectRemarks

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

## osjxAddFollowUp

<b>Definition</b>	osjxAddFollowUp (long lObjectID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Fügt eine Wiedervorlage hinzu.
<b>Parameter</b>	<b>lObjectID</b> Objekt-ID
<b>Rückgabewert</b>	-

## osjxAddSubscribe

<b>Definition</b>	osjxAddSubscribe (long lObjectID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Fügt ein Abonnement hinzu.
<b>Parameter</b>	<b>lObjectID</b> Objekt-ID
<b>Rückgabewert</b>	-

## osjxStartWorkflow

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

## osjxGetSelectedObjects

<b>Definition</b>	osjxGetSelectedObjects()		
<b>Typ</b>	Methode		
<b>Beschreibung</b>	Ermittelt ausgewählte Objekte aus der Trefferliste.		
<b>Parameter</b>	-		
<b>Rückgabewert</b>	Liste der ausgewählten Objekt in folgender Form: Obj 0-ID, Obj 0-Type; ...; Obj (n) -ID, Obj (n) -Type		

## osjxRefreshObjectInLists

<b>Definition</b>	osjxRefreshObjectInLists	(long lObjectID)
<b>Typ</b>	Methode	
<b>Beschreibung</b>	Aktualisiert das angegebene DMS-Objekt in allen offenen Listen.	
<b>Parameter</b>	<b>lObjectID</b>	Objekt-ID
<b>Rückgabewert</b>	-	
<b>Bemerkung</b>	-	
<b>Beispiel</b>	<pre>if (window.osClient) { window.osClient.osjxRefreshObjectInLists(312); } else { alert("window.osClient not exist"); }</pre>	

## osjxByteArrayToFile

<b>Definition</b>	osjxByteArrayToFile (JavaScriptArray, sting sDateiname)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Nimmt ein JavaScript-Array mit Integer-Werten entgegen und schreibt diese in eine Datei.
<b>Parameter</b>	<b>JavaScript-Array</b> <b>sDateiname</b>
<b>Rückgabewert</b>	-
<b>Bemerkung</b>	Ist der Dateiname ohne eindeutigen Pfad angegeben, wird ein 'Speichern unter'-Dialog geöffnet.
<b>Beispiel</b>	<pre>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"); }</pre>

## osjxURLDownloadToFile

<b>Definition</b>	osjxURLDownloadToFile (string sDateiUrl, sting sDateiname)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Nimmt eine Datei-URL entgegen und schreibt den Inhalt der Datei-URL in eine Datei.
<b>Parameter</b>	<b>sDateiUrl</b> <b>sDateiname</b>
<b>Rückgabewert</b>	-
<b>Bemerkung</b>	Ist der Dateiname ohne eindeutigen Pfad angegeben, wird ein 'Speichern unter'-Dialog geöffnet.
<b>Beispiel</b>	<pre>if (window.osClient) { window.osClient.osjxURLDownloadToFile ("http://www.url.de/bild.jpg", "urlbild.jpg"); } else { alert("window.osClient not exist"); }</pre>

## osjxOpenResultList

<b>Definition</b>	osjxOpenResultList (string sTitel, JSON-String)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Öffnet eine Trefferliste mit dem angegebenen Titel und den angegebenen DMS-Objekten.
<b>Parameter</b>	<b>sTitel</b> <b>JSON-String</b>
<b>Rückgabewert</b>	-

### 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

<b>Definition</b>	osjxOpenChromeDEVTools()
<b>Typ</b>	Methode

<b>Beschreibung</b>	Öffnet die DEV-Tools des Chrome-Browsers.
<b>Parameter</b>	-
<b>Rückgabewert</b>	-

## osjxCloseChromeDEVTools

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

## osjxIsDashletVisible

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

## osjxSetDashletVisible

<b>Definition</b>	osjxSetDashletVisible (dashletID, bool bVisible)
<b>Typ</b>	Methode
<b>Beschreibung</b>	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.
<b>Parameter</b>	<b>dashletID</b> (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen. <b>bVisible</b> 0 = nicht sichtbar / 1 = sichtbar
<b>Rückgabewert</b>	-

## osjxIsDashletEnabled

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

## osjxSetDashletEnabled

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

## osjxGetDashletURL

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

## osjxSetDashletURL

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

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

## osjxDashletPaneState

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

## osjxSetDashletPaneState

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

## osjxSetDashletCaption

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

<b>Parameter</b>	<b>dashletID</b> (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen. <b>sTitel</b>
<b>Rückgabewert</b>	-
<b>Beispiel</b>	<pre>if (window.osClient) { window.osClient.osjxSetDashletCaption ("OSDETAILVIEWER", "Neuer Titel"); } else { alert("window.osClient not exist"); }</pre>

## osjxGetDashletCaption

<b>Definition</b>	osjxGetDashletCaption (dashletID)
<b>Typ</b>	Methode
<b>Beschreibung</b>	Ermittelt den Titel eines Dashlets.
<b>Parameter</b>	<b>dashletID</b> (siehe oben) Dashlet-Titel als String oder Nummer des Dashlets. Ohne Angabe wird das aktuelle Dashlet angesprochen.
<b>Rückgabewert</b>	Titel des Dashlets.
<b>Beispiel</b>	<pre>if (window.osClient) { var State = window.osClient.osjxGetDashletCaption("OSDETAILVIEWER" );  alert(State); } else { alert("window.osClient not exist"); }</pre>

## osjxChangeContextToID

<b>Definition</b>	osjxChangeContextToID
<b>Typ</b>	Methode
<b>Beschreibung</b>	Documentviewer, Detailsviewer, sowie alle im Client eingerichteten Dashlets werden mit der übergebenen DMS-Objekt-ID aktualisiert und zeigen im Anschluss das übergebene Objekt an.
<b>Parameter</b>	<b>ID</b> Die DMS-Objekt-ID
<b>Rückgabewert</b>	-



**Beispiel (JavaScript)**

```
function ChangeContextID()
{
  if (window.osClient)
  {
    window.osClient.osjxChangeContextToID
    ('11460');
  }
  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:

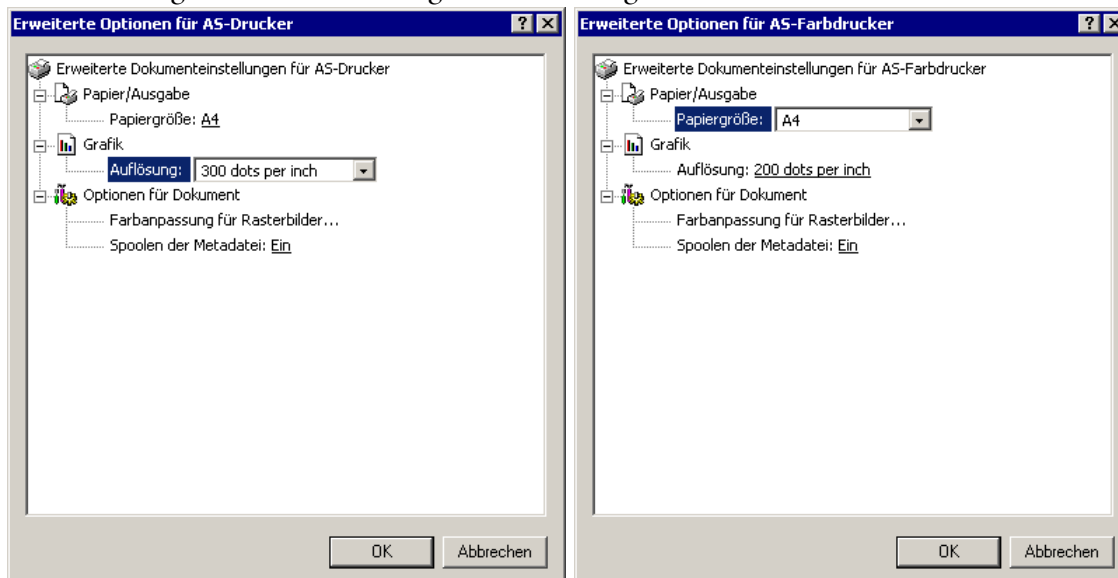
- der OS-Drucker `axprint.dll` für den Schwarz-/Weißdruck
- der OS-Farbdrucker `axcprint.dll` für den Farbdruck

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 indiziert. 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 fortsetzen 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. Darauffolgend 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: <ul style="list-style-type: none"> <li>▪ Pfad und Bezeichnung der Konfigurationsdatei <code>asprint.ini</code></li> <li>▪ Protokolleinträge (vgl. Protokollierung)</li> </ul> Zusätzlich wird in die Datei <code>asprint.ini</code> der Benutzername eingetragen: <code>USERNAME='Windows-Benutzername'</code>
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_MACHINE\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.

Über die Bibliothek `oxlist.dll` steht eine API-Schnittstelle zur Verfügung, über die Textdateien (AsciiFile) in Strukturbaumdateien (TreeFile) und auch umgekehrt Strukturbaumdateien in 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` bzw. `clients\client64` und in die Verzeichnisse der Administrationskomponenten installiert.

### 32-Bit `oxlist.dll`

Die Schnittstelle enthält die folgenden beiden Funktionen:

- 32-bit: `int WINAPI ConvertAsciiFileToTreeFile(LPCSTR lpAsciifile, LPCSTR lpDefinition, LPCTSTR lpTreefile)`
- 32-bit: `int WINAPI ConvertTreeFileToAsciiFile(LPCSTR lpTreefile, LPCSTR lpAsciifile)`

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

/*!
  Die Funktion generiert aus einer ASCII-Datei und einer zugehörigen Ebenendefinition 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 (32-Bit):**

```
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
```

**64-Bit oxlist.dll**

Die Schnittstelle enthält die folgenden beiden Funktionen:

- 64-bit: int WINAPI ConvertAsciiFileToTreeFile(LPCWSTR lpAsciifile, LPCWSTR lpDefinition, LPCTSTR lpTreefile)
- 64-bit: int WINAPI ConvertTreeFileToAsciiFile(LPCWSTR lpTreefile, LPCWSTR lpAsciifile)

```
int WINAPI ConvertAsciiFileToTreeFile(LPCWSTR lpAsciifile, LPCWSTR lpDefinition, LPCWSTR
lpTreefile);
/*!
Die Funktion generiert aus einer ASCII-Datei und einer zugehörigen Ebenendefinition eine
Strukturbaumdatei.

@param LPCWSTR lpAsciifile - ASCII-Datei die sortiert die Kürzel und zugehörigen
Bezeichnungen enthält. (Kürzel und Bezeichnungen müssen durch Leerzeichen getrennt sein,
nicht durch Tabulatoren)
@param LPCWSTR lpDefinition - Ebenendefinition z.Bsp. AA@A.9.AA
(A -> Buchstaben, Zahlen 9 -> nur Zahlen, @ -> kein Trennzeichen, . -> Trennzeichen)
@param LPCWSTR lpTreefile - Dateiname für eine zu erzeugende Baumstrukturdatei.
@return int - 0 -> bei Erfolg, sonst -> Fehlercode
*/
int WINAPI ConvertTreeFileToAsciiFile(LPCWSTR lpTreefile, LPCWSTR lpAsciifile);
/*!
Die Funktion generiert aus einer Baumstrukturdatei eine ASCII-Datei.

@param LPCWSTR lpTreefile - Dateiname einer existierenden Baumstrukturdatei.
@param LPCWSTR lpAsciifile - Zu erzeugende ASCII-Datei die sortiert die Kürzel und
zugehörigen Bezeichnungen enthält.
@return int - 0 -> bei Erfolg, sonst -> Fehlercode
```

**Struktur der Textdatei**

Beim Erzeugen einer Strukturbaumdatei im enaio® editor geben Sie ein Ebenendefinition an. Beim Konvertieren einer Textdatei in eine Strukturbaumdatei geben Sie ebenfalls diese Ebenendefinition mit an.

Innerhalb der 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 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, setzen 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 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.

# Anhang: Ressourcenstorage

## Allgemeines Schema der JSON-Parameter

Die Methoden 'DeleteResourceString', 'GetResourceString' und 'SetResourceString' ermöglichen Ein- und Ausgabeparameter im JSON-Format. Für JSON-Tags muss die Groß- und Kleinschreibung beachtet werden.

Für die Inhalte der Sprachkürzel und Schlüssel wird die Groß- und Kleinschreibung entsprechend der Datenbankeinstellungen beachtet oder nicht beachtet.

Die JSON-Parameter müssen diesem formalen Schema entsprechen:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "Keys": {
      "type": "array",
      "items": [
        {
          "type": "object",
          "properties": {
            "Key": {
              "type": "string"
            },
            "Lang": {
              "type": "string"
            },
            "Values": {
              "type": "array",
              "items": [
                {
                  "type": "object",
                  "properties": {
                    "Lang": {
                      "type": "string"
                    },
                    "Value": {
                      "type": "string"
                    }
                  },
                  "required": [
                    "Value"
                  ]
                }
              ]
            }
          }
        }
      ]
    },
    "required": [
      "Key"
    ]
  }
},
"required": [
  "Keys"
]
}
```

## Beispiel für Abfrageparameter

Für die Methoden 'GetResourceString' und 'DeleteResourceString' kann der Parameter 'JSON\_Request' beispielsweise so gesetzt sein:

```
{
  "Keys": [
    { "Key": "test.key1", "Lang": "en_US" },
    { "Key": "test.test3.*", "Lang": "de_DE" },
    { "Key": "test.test4.*", "Lang": "*" }
  ]
}
```

## Beispiel für Einfüge- und Rückgabeparameter

Für die Methode 'SetResourceString' kann der Parameter 'JSON\_Request' und für die Methode 'GetResourceString' kann der Rückgabeparameter 'Result\_JSON' beispielsweise folgendermaßen gesetzt sein:

```
{
  "Keys": [
    {
      "Key": "test.key1",
      "Values" : [{ "Lang": "en_US", "Value": "Bill" } ]
    },
    {
      "Key": "test.test3.a",
      "Values": [{ "Lang": "de_DE", "Value": "Rechnung" } ]
    },
    {
      "Key": "test.test3.b",
      "Lang": "de_DE",
      "Values": [{ "Value": "Quittung" } ]
    },
    {
      "Key": "test.test4.a",
      "Values": [ { "Lang": "en_US", "Value": "Bills"},
                  { "Lang": "de_DE", "Value": "Rechnungen" } ]
    }
  ]
}
```