OPTIMAL SYSTEMS

enaio®

Softwaredokumentation
enaio® editor-for-events

Version 8.50

06.05.2019
Version 8.50

Inhalt

Zur Einführung

Über das Handbuch

Das Handbuch liegt Ihnen als PDF-Datei im Dokumentationsverzeichnis vor.

Im Handbuch werden die Arbeitsschritte in der Regel so beschrieben, wie sie mit der Maus und den Schaltflächen im Menüband ausgeführt werden. Sie können alle Arbeitsschritte aber auch mit der Tastatur ausführen. Der enaio® editor-for-events hält sich an die Konventionen von MS Windows. Benutzen Sie die Taste **Alt** zusammen mit den unterstrichenen Buchstaben.

Über enaio® editor-for-events

Ein Event ist ein VB-Skript, das einer Aktion in enaio® client mit einem DMS-Objekt oder einer Anwendungssituation zugeordnet ist und aus enaio® client automatisch durch die Benutzer-Aktion gestartet wird. So kann beispielsweise beim Speichern der Verschlagwortung eines DMS-Objekts durch ein VB-Skript eine Plausibilitätsprüfung erfolgen oder es können automatisch Daten ergänzt werden.

Ein Event kann ebenfalls einem Serverjob zugeordnet werden. Wird ein Serverjob ausgeführt, kann vorher oder nachher ein VB-Skript ausgeführt werden.

Events erstellen Sie mit enaio® editor-for-events. enaio® editor-for-events ist als Komponente des Contentmanagement-, Workflow- und Archivsystems enaio® in enaio® client integriert. Verfügt ein Benutzer über die notwendigen Systemrollen und Lizenzen, werden die entsprechenden Funktionen in enaio® client freigeschaltet.

Zu den meisten Events erstellt enaio® client eine Übergabedatei mit kontextbezogenen Daten, auf die Sie mit einem VB-Skript zugreifen und die Sie ändern können. Einige Events benötigen einen Rückgabewert, den Sie per Skript in die Übergabedatei eintragen. Einige aktualisieren Daten anhand der geänderten Einträge in der Übergabedatei. Abhängig vom Rückgabewert führt enaio® client dann die Aktionen weiter aus oder nicht.

Über das eingebundene ActiveX-Control `oxactive.dll` stehen Ihnen Methoden und Objekte zur Verfügung, mit denen Sie in einem Skript Daten aus der Übergabedatei leicht auslesen und ändern können.

Darüber hinaus können Sie über ein VB-Skript unabhängig von enaio® client Aktionen ausführen oder über die COM-Schnittstelle unabhängig von der Übergabedatei Aktionen an enaio® client auslösen. Details zur COM-Schnittstelle finden Sie in der Dokumentation ' enaio® client Programmierreferenz'.

Die Events werden verschlüsselt in der Datenbank gespeichert, können aber auch als Datei gespeichert und so ausgetauscht werden.

Events, die OPTIMAL SYSTEMS für Sie schreibt, erhalten Sie als verschlüsselte Datei, die Sie über enaio® editor-for-events importieren und einem DMS-Objekt zuordnen.

Über enaio® administrator können Sie Events Benutzern zuordnen.

Events für enaio® webclient

Events für enaio® webclient werden ebenfalls mit enaio® editor-for-events erstellt, allerdings nicht in VBScript sondern in JavaScript. Die Dokumentation finden Sie online.

Das Aktivieren und Deaktivieren von Events in enaio® administrator (vgl. 'Administration der Events') gilt nur für enaio® client. Events in enaio® webclient werden immer ausgeführt.

Installation, Lizenzierung, Sicherheitssystem

Die enaio® editor-for-events Komponenten werden bei der Installation des enaio® client automatisch installiert.

Wollen Sie Events erstellen, benötigen Sie am Arbeitsplatz die Lizenz 'EVE', die Clientlizenz 'ASC' und die Systemrollen 'Client: Events erstellen'. Zum Testen von Events benötigen Sie zusätzlich die Systemrolle 'Client: Events debuggen'.

Um Events, die Sie von OPTIMAL SYSTEMS erhalten haben, zu importieren, benötigen Sie am Arbeitsplatz die Clientlizenz 'ASC' und die Systemrolle 'Editor: Datenbank anpassen'.

Zum Löschen von Events benötigen Sie die Systemrollen 'Editor: Datenbank anpassen' und 'Client: Events erstellen'. In beiden Fällen benötigen Sie nicht die Lizenz 'EVE'.

Wird über Events mittels des Serverjobs 'dtr.SynchronizeData' eine Datenübernahme ausgeführt, wird die Lizenz 'SDE' benötigt.

Über die Einstellungen in enaio® client müssen den Zugriff auf die Funktionen einschalten (siehe 'Events erstellen'), damit sie im Bereich 'Objektsuche' zur Verfügung stehen.

Events

Schnelleinstieg

Ein Event wird von enaio® client oder enaio® server als Folge einer Benutzer-Aktion, einer Anwendungs-Situation oder einer Jobausführung gestartet und führt ein Skript aus.

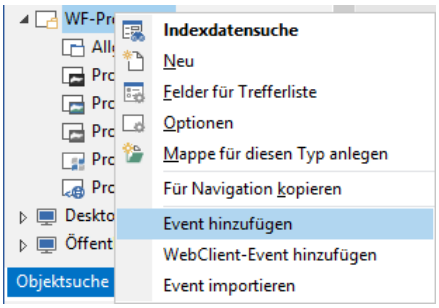
Um Events nutzen zu können, müssen Lizenzen vorhanden sein sowie die nötigen Systemrollen vergeben werden (siehe 'Installation, Lizenzierung, Sicherheitssystem' sowie 'Administration der Events').

Zu unterscheiden ist zwischen Client- und Server-Events:

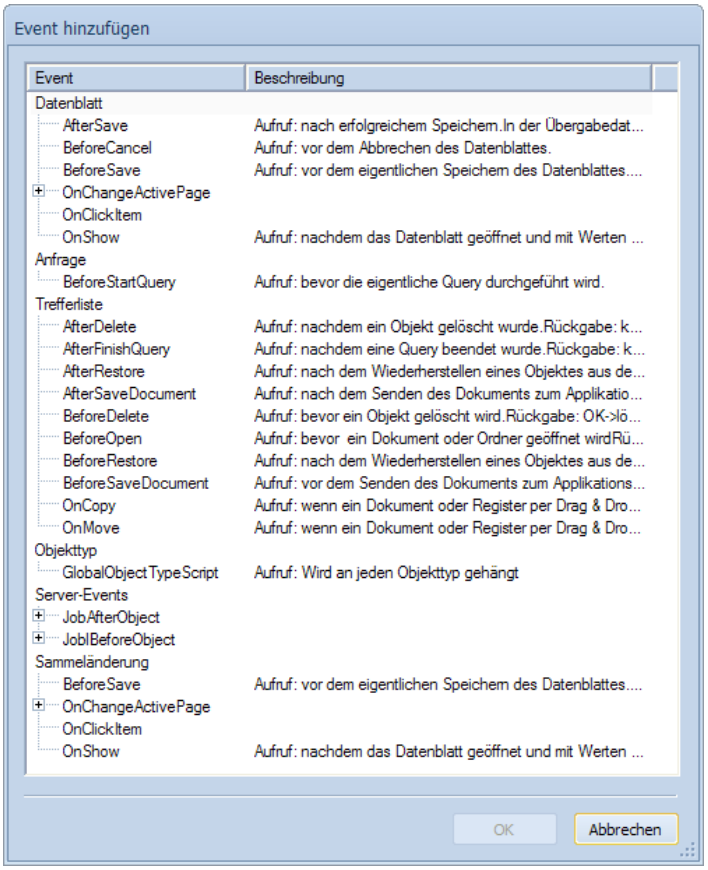
- Clientseitige Events (siehe 'Clientseitige Events') werden in enaio® client ausgelöst, beispielsweise durch die Anzeige eines Objekts.
- Serverseitige Events dagegen sind einem Serverjob zugeordnet (siehe 'Serverseitige Events').

Mit Hilfe von Events können Sie zum Beispiel beim Speichern einer Verschlagwortungsmaske gezielt einige Felder, die der Benutzer leer gelassen hat, ergänzen lassen: Falls der Bearbeiter seinen Namen nicht in das Feld 'Bearbeiter' eingetragen hat, kann dies vor dem Speichern der Verschlagwortungsmaske per Event automatisch geschehen. Hierfür können Sie das Event **BeforeValidate** nutzen.

Um ein derartiges Client-Event zu erstellen, öffnen Sie im Bereich 'Objektsuche' das Kontextmenü des DMS-Objekts und wählen den Eintrag **Event hinzufügen**:

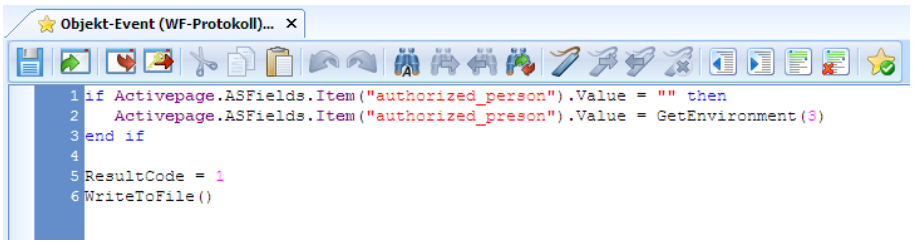


Im Dialog **Event hinzufügen** wählen Sie nun den Eintrag **BeforeValidate** aus und bestätigen mit **OK**:



Daraufhin wird **enaio® editor-for-events** geöffnet (siehe 'Für die Suche in enaio® client kann jeder Benutzer dieses Anfrageverhalten über seine benutzerspezifischen Einstellungen im Bereich 'Anfrageverhalten' festlegen.

enaio® editor-for-events'). In diesem Editor geben Sie den VBScript-Code ein, der durch das Event **BeforeValidate** ausgeführt werden soll.



Im Editor geben Sie Folgendes ein:

```
if ActivePage.ASFields.Item("Bearbeiter").Value = "" then
   ActivePage.ASFields.Item("Bearbeiter").Value = GetEnvironment(3)
end if
ResultCode = 1
WriteToFile()
```

Um abzufragen, ob der Benutzer im Feld 'Bearbeiter' der Verschlagwortungsmaske einen Eintrag gemacht hat oder nicht (Zeile 1), greift man auf den Wert des Felds über das Objekt **ActivePage** zu. Dieses Objekt ist ein Verweis auf das aktive Datenblatt der Verschlagwortungsmaske (siehe 'Objekte').

Um auf ein konkretes Feld der Verschlagwortungsmaske (hier das Feld 'Bearbeiter') zuzugreifen, kann man die Methode **Item** der Kollektion **ASFields**, die alle Felder der Maske beinhaltet, aufrufen. Um ein bestimmtes Feld der Verschlagwortungsmaske anzusprechen, übergibt man dessen Bezeichnung als Parameter. Die Bezeichnungen der Felder einer Verschlagwortungsmaske können in enaio® editor für alle Objekte eingesehen werden.

Falls kein Bearbeiter eingetragen wurde, wird der Benutzername des angemeldeten Benutzers ermittelt und in das Feld 'Bearbeiter' geschrieben (Zeile 2).

Das Ändern von Werten geschieht in der Übergabedatei (siehe 'Übergabedateien'). Damit die Änderungen aus der Übergabedatei vor dem Weiterführen der Aktion zurückgeschrieben und somit berücksichtigt werden, muss der Rückgabewert '1' geschrieben werden (Zeile 5, zu den Rückgabewerten siehe 'Clientseitige Events').

Speichern Sie das Skript durch Klick auf  **Speichern**.

Beim nächsten Aufruf der Verschlagwortungsmaske greift das VB-Skript bereits und fügt den Namen des Bearbeiters in das Feld ein, falls dieses Feld beim Speichern noch leer sein sollte.

> Wenn Sie über Skripte auf Dialogelemente Bezug nehmen, die Sonderzeichen enthalten, können Fehler auftreten. Verwenden Sie in diesem Fall die internen Namen für Bezüge auf Dialogelemente.

Clientseitige Events

enaio® client bzw. enaio® server führt VB-Skripte aus, wenn das zugeordnete Event eintrifft. Wählen Sie ein Event aus und erstellen Sie den VB-Skript-Code. Events benötigen teilweise einen Rückgabewert, den Sie in die Übergabedatei schreiben.

Die Client-Events sind in folgende Gruppen eingeteilt:

- Anwendung

  Events mit Bezug zu enaio® client.

- Datenblatt

  Events mit Bezug zu Datenblättern von DMS-Objekten, die neu angelegt werden oder deren Verschlagwortung geändert wird.

- Trefferliste

  Events mit Bezug zu Trefferlisten als Ergebnis einer Suche oder Inhaltslisten von Ordnern und Registern.

- Anfrage

  Events mit Bezug zu Suchmasken.

- Schrank

  Event für Drag&Drop von Dokumenten in einen Ordner

Die folgenden clientseitigen Events stehen zur Verfügung:

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
AfterLogin	Anwendung	nachdem sich der Benutzer angemeldet hat	keine Rückgabe	8
BeforeLogout	Anwendung	bevor sich der Benutzer ausloggt	1 = Ausloggen wird durchgeführt 0 = Ausloggen wird nicht durchgeführt	9
OnStartApp	Anwendung	nachdem enaio® client gestartet wurde, unmittelbar vor 'AfterLogin'	keine Rückgabe	14
OnCloseApp	Anwendung	nach dem Einchecken aller Dokumente direkt vor dem Beenden von enaio® client	1 = Beenden wird durchgeführt 0 = Beenden wird nicht durchgeführt	15
BeforeLink	Anwendung	nachdem in enaio® client eine Verknüpfung oder eine Relation erstellt wurde, bevor die Daten zum Server übertragen werden	0 = Verknüpfung oder Relation erstellen 1 = Relation nicht erstellen, Dialog offen halten 2 = Relation nicht erstellen, Dialog schließen Bei Verknüpfungen führt jeder Wert ungleich 0 zum Abbruch.	21
AfterLink	Anwendung	nachdem der Server die Verknüpfung oder Relation angelegt hat	keine Rückgabe	22
BeforeDeleteLink	Anwendung	nachdem in enaio® client eine Verknüpfung oder eine Relation gelöscht wurde, bevor die Daten zum Server übertragen werden	0 = Verknüpfung oder Relation löschen -2 = Komplett abbrechen Andere Werte führen zum Abbruch des aktuellen Löschvorganges, es wird mit dem nächsten Selektierten Objekt weitergemacht.	23
AfterDeleteLink	Anwendung	nachdem der Server die Verknüpfung oder Relation gelöscht hat	keine Rückgabe	24

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
StartAction	Anwendung	Aufruf durch eine Server-Benachrichtigung	keine Rückgabe	31
OnClickItem	Datenblatt	wenn eine Schaltfläche angeklickt wurde	0 = Daten werden aus der Übergabedatei in das Datenblatt gelesen 1 = Daten des Datenblatts werden nicht geändert	13
OnShow	Datenblatt	vor dem Öffnen des Datenblatts In der Übergabedatei steht 'Action=NEW', wenn ein Datenblatt zur Neuanlage geöffnet wird, 'Action=UPDATE', wenn ein Datenblatt zum Bearbeiten geöffnet wird, 'Action=READONLY', wenn ein Datenblatt schreibgeschützt geöffnet wird, 'Action=REQUEST', wenn eine Suchmaske geöffnet wird.	1 = Daten werden aus der Übergabedatei in das Datenblatt gelesen 0 = Daten des Datenblatts werden nicht geändert -1 = Datenblatt wird nicht geöffnet	1

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
BeforeValidate	Datenblatt	nachdem auf 'Speichern' geklickt wurde, noch vor der Plausibilitätsprüfung des enaio® client und dem Speichern. In der Übergabedatei steht 'Action=NEW', wenn ein Objekt neu angelegt wird oder 'Action=Update', wenn die Daten geändert werden.	1 = Daten werden aus der Übergabedatei in das Datenblatt gelesen enaio® client führt danach die Plausibilitätsprüfung durch und speichert die Daten. 0 = das Speichern des Datenblattes wird durchgeführt, die Daten aus der Übergabedatei werden dabei nicht in das Datenblatt gelesen -1 = das Neuanlegen oder das Ändern der Daten wird abgebrochen -2 = Daten werden aus der Übergabedatei in das Datenblatt gelesen, das Datenblatt bleibt offen, die Daten werden nicht gespeichert	2
AfterValidate	Datenblatt	nach der Plausibilitätsprüfung des enaio® client und vor dem Speichern In der Übergabedatei steht 'Action=NEW', wenn ein Objekt neu angelegt wird oder 'Action=Update', wenn die Daten geändert werden.	1 = Daten werden aus der Übergabedatei in das Datenblatt gelesen 0 = das Speichern des Datenblattes wird durchgeführt, die Daten aus der Übergabedatei werden dabei nicht in das Datenblatt gelesen -1 = das Neuanlegen oder das Ändern der Daten wird abgebrochen -2 = Daten werden aus der Übergabedatei in das Datenblatt gelesen, das Datenblatt bleibt offen, die Daten werden nicht gespeichert	36

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
AfterSave	Datenblatt	nach der Plausibilitätsprüfung des enaio® client und dem Speichern In der Übergabedatei steht 'Action=NEW', wenn ein Objekt neu angelegt wird oder 'Action=Update', wenn die Daten geändert werden.	keine Rückgabe	3
BeforeCancel	Datenblatt	nachdem auf einem Datenblatt auf 'Abbrechen' geklickt wurde	0 = das Datenblatt wird nicht geschlossen. Bei anderen Rückgabewerten wird das Datenblatt geschlossen.	30
OnEnterPage	Datenblatt	beim Wechseln der Seite des Dialogelements 'Pagecontrol'	1 = Daten von Feldern auf dem PageControl, die in der Übergabedatei gesetzt sind, werden in die Suchmaske zurückgelesen 0 = Daten aus der Übergabedatei werden ignoriert	25
OnLeavePage	Datenblatt	wenn eine Seite des Dialogelements 'Pagecontrol' verlassen wird	keine Rückgabe	37
OnFocusGained	Datenblatt	Wenn der Fokus auf ein Textfeld gelegt wird. Das Event kann jedem Textfeld eines Datenblatts zugeordnet werden.	keine Rückgabe	32
OnCellFocusGained	Datenblatt	Wenn der Fokus auf ein Tabellenzelle gelegt wird.	keine Rückgabe	39

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
OnValueChanged	Datenblatt	Wenn die Eingabe in einem Textfeld abgeschlossen wurde	0 = Keine Änderung -1 = Zurück zum Textfeld, der Eintrag wird nicht geändert 1 = Daten werden aus der Übergabedatei in das Textfeld eingelesen	33
OnCellValueChanged	Datenblatt	Wenn die Eingabe in einer Tabellenzelle abgeschossen wurde	0 = Keine Änderung -1 = Zurück zum Textfeld, der Eintrag wird nicht geändert 1 = Daten werden aus der Übergabedatei in das Textfeld eingelesen	40
BeforeAddRow	Datenblatt	Bevor in einer Tabelle eine neue Zeile hinzugefügt wird.	0 = Keine Änderung -1 = Zeile wird nicht hinzugefügt	34
BeforeDeleteRow	Datenblatt	Bevor in einer Tabelle eine Zeile gelöscht wird.	0 = Keine Änderung -1 = Zeile wird nicht gelöscht	35
BeforeStartQuery	Anfrage	nachdem auf der Suchmaske auf 'Anfrage starten' geklickt wurde, noch vor dem eigentlichen Durchführen der Suche	1 = Daten werden aus der Übergabedatei in die Suchmaske gelesen enaio® client startet danach die Suche. 0 = Suche ohne Änderung starten -1 = Suche wird abgebrochen	4
AfterFinishQuery	Trefferliste	nach der Suche	keine Rückgabe	5
BeforeDelete	Trefferliste	bevor ein Objekt gelöscht wird	1 = Löschen wird durchgeführt 0 = Löschen wird nicht durchgeführt	11

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
BeforeUndoCheckOut	Trefferliste	bevor in enaio® client auf einem Dokument die Benutzeraktion 'Auschecken zurücknehmen' ausgeführt wird Über die Konstante 'NumberOfSelectedDocuments' können Sie im Skript abfragen, für wie viele Dokumente das Auschecken zurückgenommen werden soll.	-2 = Auschecken rückgängig für dieses Dokument nicht zulassen -1 = Auschecken rückgängig für alle markierten Dokumente abbrechen 0 = Normaler Ablauf für Benutzeraktion 'Auschecken zurücknehmen' 1 = Auschecken rückgängig erlaubt, Benutzer wird nicht gefragt	27
AfterDelete	Trefferliste	nachdem ein Objekt gelöscht wurde	keine Rückgabe	12
BeforeOpen	Trefferliste	bevor ein Dokument geöffnet wird	0 = nicht öffnen 1 = Öffnen 2 = schreibgeschützt Öffnen -4 = Nur Workflow-Event: geöffnetes Objekt wird im Dokumenten-Viewer angezeigt	16
OnMove	Trefferliste	wenn ein Dokument oder Register innerhalb eines Schranks verschoben wird	0, 1 = Verschieben wird durchgeführt -1 = Verschieben wird nicht durchgeführt	17
OnMoveExtern	Trefferliste	wenn ein Dokument in einen anderen Schrank verschoben wird	0, 1 = Verschieben wird durchgeführt -1 = Verschieben wird nicht durchgeführt	42
OnAddLocation	Trefferliste	wenn ein Dokument oder Register per Drag & Drop einen weiteren Standort erhält	0, 1 = Standort zuordnen -1 = Standort nicht zuordnen	18
OnCreateCopy	Trefferliste	wenn ein Dokument oder Register kopiert wird	0, 1 = Kopieren wird durchgeführt -1 = Kopieren wird nicht durchgeführt	43

Event	Bezug	Zeitpunkt des Aufrufs	Beschreibung der Rückgabe	Event-Code
BeforeSaveDocument	Trefferliste	vor dem Einchecken eines Dokuments	0 = Dokument wird eingecheckt -2 = komplett abbrechen, Dokument wird nicht eingecheckt Bei anderen Rückgabewerten wird das Dokument nicht eingecheckt, sondern mit den folgenden Dokumenten fortgefahren.	19
AfterSaveDocument	Trefferliste	nach dem Einchecken eines Dokuments	keine Rückgabe	20
BeforeRestore	Trefferliste	vor dem Wiederherstellen eines Objektes aus dem Papierkorb	0 = Das Objekt wird nicht wiederhergestellt. Bei anderen Rückgabewerten wird das Objekt wiederhergestellt.	28
AfterRestore	Trefferliste	nach dem Wiederherstellen eines Objektes aus dem Papierkorb	keine Rückgabe	29
FileDrop	Schrank	nach dem Ablegen von Dateien an einem Standort	-1 = Abbruch 1 = Client aktualisiert nur die Trefferliste 0 = Client übernimmt die Dateien	200

Das Anwendungs-Event 'OnContextChange' wurde für den Contentviewer verwendet, der mit der Version 7.00 durch den Documentviewer ersetzt wurde. Das Event wird deshalb nicht mehr verwendet und ist ab der Version 7.00 deaktiviert. Es kann durch einen Eintrag in der Konfigurationsdatei `as.cfg` im Verzeichnis `\etc` des Datenverzeichnisses aktiviert werden:

```
[SYSTEM]
ENABLE_CONTEXTCHANGE_EVENT=1
```

Events für Sammeländerungen

Für Sammeländerungen sind die folgenden Datenblatt-Events eingebunden:

Event	Event-Code
BeforeValidate	100

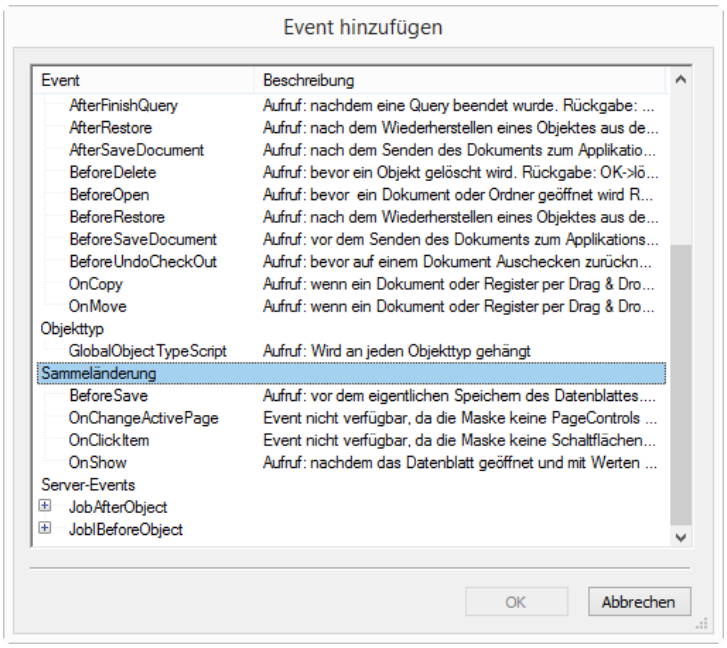
Event	Event-Code
AfteValidate	104
OnShow	101
OnClickItem	102
OnChangeActivePage	103

Übergabedateien enthalten einen zusätzlichen Abschnitt mit folgender Struktur:

```
[BATCHUPDATE]
Count=<Anzahl der Objekte>
ID0=<Objekt-ID>
IDn= …
```

Für Sammeländerungen können nur diese Events eingesetzt werden.

Events für Sammeländerungen werden im Arbeitsbereich über das Kontextmenü eines Objekttyps hinzugefügt.



Im Bereich 'Objektsuche' sind Events für Sammeländerungen mit einem Blitz gekennzeichnet.

Die Funktion entspricht derjenigen der Events für ein Objekt.

Event 'StartAction'

Das Event 'StartAction' wird über die Methode 'krn.SendMessageToClients' aufgerufen. Die Methode kann über enaio® enterprise-manager oder aus Skripten und Anwendungen aufgerufen werden. Informationen zu 'krn.SendMessageToClients' finden Sie in der Dokumentation 'enaio® Server API'.

Der Aufruf des Events erfolgt über den Wert 'StartEvent' des Parameters 'Message'. Der Wert des Parameters 'Text' wird als Event-Datei übergeben und kann durch das Skript ausgewertet werden.

Übergabedateien

enaio® client erzeugt für Events Übergabedateien, aus denen Sie Daten auslesen und ändern und in die Sie einen Rückgabewert schreiben können.

Übergabedateien erhalten die Dateiendung `*.evt` und werden clientseitig in das benutzerspezifische temporäre Verzeichnis `\temp\OSTEMP\` geschrieben.

Mit Hilfe des Codes 'MsgBox filename' können Sie sich innerhalb eines Skripts den vollständigen Pfad und den Dateinamen der Übergabedatei in einem Dialog ausgeben.

Auf die Übergabedateien erhalten Sie direkt über das ActiveX-Control `oxactive.dll` (siehe 'Das ActiveX-Control OXACTIVE') lesenden und schreibenden Zugriff. Die Übergabedateien werden automatisch gelöscht, sobald enaio® client die notwendigen Daten ausgelesen hat.

Für einige Events müssen Sie Rückgabewerte in die Übergabedateien schreiben, damit enaio® client die gewünschte Aktion durchführen kann. Schreiben Sie hierfür folgenden Code in die Übergabedatei:

```
ResultCode=1
WriteToFile()
```

Bei folgenden Events wird keine Übergabedatei erzeugt:

- AfterLogin
- OnStartApp
- AfterFinishQuery

Bei folgenden Events ist die Übergabedatei leer und wird nur erzeugt, damit Sie einen Rückgabewert dort eintragen, um die Aktion fortzuführen:

- BeforeLogout
- OnCloseApp

Die folgenden Events erzeugen Übergabedateien mit Daten, die Sie auslesen, weiterverarbeiten und in die Sie einen Rückgabewert schreiben können:

- OnClickItem
- OnShow
- BeforeValidate
- AfterValidate
- AfterSave
- BeforeStartQuery
- BeforeOpen
- BeforeDelete
- AfterDelete
- OnMove
- OnMoveExtern

- OnAddLocation
- OnCreateCopy
- BeforeLink
- BeforeDeleteLink
- BeforeSaveDocument
- BeforeUndoCheckOut
- BeforeCancel
- BeforeRestore
- OnChangeActivePage
- OnValueChanged
- OnCellValueChanged
- BeforeAddRow
- BeforeDeleteRow

Die folgenden Events erzeugen Übergabedateien mit Daten, die Sie auslesen und weiterverarbeiten können, in die Sie aber keinen Rückgabewert schreiben:

- AfterLink
- AfterDeleteLink
- AfterSaveDocument
- AfterRestore
- OnLeavePage
- OnFocusGained
- OnCellFocusGained

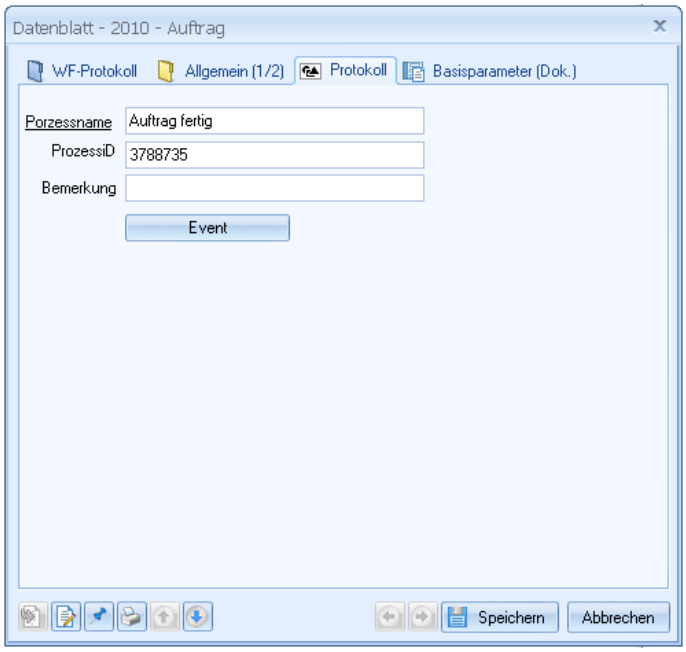
Wenn Sie über Skripte auf Dialogelemente Bezug nehmen, die Sonderzeichen enthalten, können Fehler auftreten. Verwenden Sie in diesem Fall die internen Namen für Bezüge auf Dialogelemente.

Struktur der Übergabedateien

Bei Übergabedateien handelt es sich um Textdateien, die mit jedem beliebigen Texteditor, wie beispielsweise Notepad, geöffnet werden können.

In diesen finden Sie Informationen zu Registern von Verschlagwortungsmasken, einschließlich deren Datenfeldern und Inhalten.

Die Übergabedateien sind in Abschnitte gegliedert: Deren Beginn ist jeweils durch eine Seitenangabe in eckigen Klammern gekennzeichnet. Diese Abschnitte entsprechen den Reitern der Verschlagwortungsmaske. Die Seiten sind, bei 'Null' beginnend, fortlaufend nummeriert: [PAGE00], [PAGE01], [PAGE02] usw.



Zusätzlich existiert in der Übergabedatei noch ein Abschnitt [GLOBALS], in der übergeordnete, abschnittsübergreifende Informationen gespeichert werden.

Wenn Sie auf einer Verschlagwortungsmaske ein Pagecontrol eingefügt haben, enthalten dessen Seiten in der Übergabedatei eigene Unterabschnitte, indem an die Seitenangabe der Abschnitte noch eine ID angehängt wird:

```
[PAGE00#BD5D78C3F05A45538A226667DC644C01]
[PAGECTRL#77EEBAE5F6834B20BE5A18F939191C8A]
#NAME#=PageCtrl1
#OSINTERNNAME#=PageCtrl1
#PAGENAME1#=Laufzeit
#PAGEOSINTERNNAME1#=Laufzeit
#PAGEGUID1#=71134100B5BA43DD94F81CB5A100BDD6
#PAGENAME2#=Erinnerung
#PAGEOSINTERNNAME2#=Erinnerung
#PAGEGUID2#=2C8082795C704ED0A4E79D78410E6E71
...
```

Die einzelnen Seiten des Pagecontrols können darüber hinaus über die Schlüssel #PAGENAME#, #PAGEOSINTERNNAME# und #PAGEGUID# identifiziert werden.

Pro Zeile werden ein Schlüssel und dessen Wert getrennt durch ein Gleichheitszeichen aufgelistet:

```
...
#OSMAIN#=4
#OSMODIFIED#=0
#OSFIELDMODE#=1
FULLTEXT=
VTREQUESTTYPE=0
#OSACT#=1
...
```

Art und Anzahl der Schlüssel in der Übergabedatei ist jeweils abhängig vom aufrufenden Kontext: Hierbei spielen sowohl das Event als auch der Aufbau der Verschlagwortungsmaske eine Rolle.

Bei einigen Schlüsseln setzt sich der Wert aus mehreren Teilen zusammen, die durch ein Trennzeichen voneinander separiert werden. Das Trennzeichen muss entweder das Zeichen '\021' oder ASCII-Code '17' sein.

Schlüssel der PAGE-Abschnitte

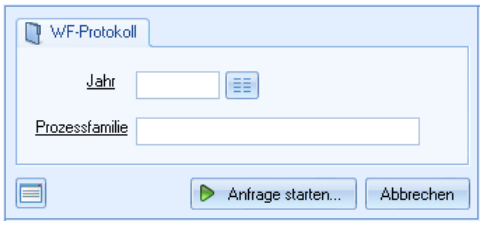
Schlüssel	Bedeutung
#OSACT#	'1' beim aktiven Register der Verschlagwortungsmaske, sonst '0'.
#OSEXP#	Falls eine Suche im Expertenmodus gestartet wurde, wird dieser Schlüssel in die Übergabedatei mit dem Wert '1' geschrieben.
#OSFIELDMODE#	GUID (=1) oder Position (=0) des Felds wird erwartet und ausgewertet
#OSIDENT#	Objekt-Indexnummer
#OSMAIN#	der Haupttyp des Dokuments: '1': X-Dokument (Graustufen-Grafik) '2': D-Dokument (SW-Grafik) '3': P-Dokument (Farb-Grafik) '4': W-Dokument (Windows-Dokument) '5': M-Dokument (Bewegtbild-Dokument) '6': Q-Dokument (Mails) '7': XML-Dokument '0': Ordner '99': Register
#OSMODIFIED#	Zeitstempel der letzten Änderung des Objekts (Verschlagwortungsmaske oder Dokument)
#OSNAME#	Bezeichnung des Registers in der Verschlagwortungsmaske
#OSPOS000#	Fortlaufend nummerierte Felder der Verschlagwortungsmaske mit ihren Inhalten
#OSTYPE#	Typ-ID des Objekts
#OSSYSTEMID#	ID des Systems, das das Objekt verwaltet. enaio® server haben immer die OSSYSTEMID '0'.
#OSFOREIGNID#	ID des Objekts in einem Fremdsystem, falls das Objekt in einem Fremdsystem verwaltet wird.
FELD0	Fortlaufend nummerierte Informationen über die Felder der Verschlagwortungsmaske, durch Trennzeichen separiert
FIELDEXT0	Erweiterung von Feld0, ebenfalls fortlaufend nummeriert
FILECOUNT	Anzahl der Dateien, die dem Objekt zugeordnet sind

Schlüssel des GLOBALS-Abschnitts

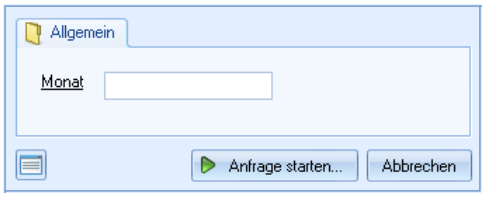
Schlüssel	Bedeutung
Action	Mögliche Werte sind: UPDATE – wenn Sie ein Datenblatt zum Bearbeiten geöffnet oder Daten geändert haben REQUEST – wenn Sie eine Suchmaske geöffnet haben NEW – wenn Sie ein Datenblatt/Objekt zur Neuanlage geöffnet haben READONLY – wenn Sie ein Datenblatt schreibgeschützt geöffnet haben BEFORELINK – bei Aufruf des gleichnamigen Events
EventCode	Numerische ID des auslösenden Events, beispielsweise '1' für das Event 'OnShow'. Für die numerischen Codes siehe die tabellarische Übersicht 'Zur Einführung'
Handle	ID des Ziehpunktes der Verschlagwortungsmaske
OrdIdent	Interne ID des Ordners
OrdType	Typ des Ordners
RegIdent	Interne ID des Registers
RegType	Typ des Registers
EXTERNDROPFILE	Pfad zu einer Datei, die per Drag&Drop übernommen wird.

Übergabedateien der clientseitigen Events

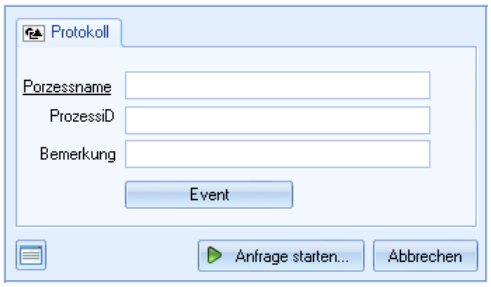
Die folgenden Beispiele für Übergabedateien beziehen sich jeweils auf ein Dokument vom Typ 'Protokoll' im Register 'Allgemein' im Ordner 'WF-Protokoll'. Abhängig vom Event finden Sie Ordnerdaten, Registerdaten und Dokumentdaten in den jeweiligen Übergabedateien, die sich auf diese DMS-Objekte beziehen.



Ordner-Maske 'WF-Protokoll'



Register-Maske 'Allgemein'



Dokumenttyp-Maske 'Protokoll'

OnClickItem

Das Event wird ausgelöst, wenn Sie auf eine Schaltfläche der Verschlagwortungsmaske eines DMS-Objekts klicken, in diesem Beispiel auf die Schaltfläche 'Event' eines Dokuments vom Typ 'Protokoll'.

enaio® client erzeugt für das Event `OnClickItem` eine Übergabedatei mit Daten zum Ordner, zum Register, zum Dokument und zu den Basisparametern.

Die Datei enthält folgende Abschnitte:

- [PAGE00]

  In diesem Abschnitt finden Sie Ordnerdaten, die Verschlagwortung des Ordners und die Felddefinitionen des Ordnertyps.

- [PAGE01]

  In diesem Abschnitt finden Sie Registerdaten, die Verschlagwortung des Registers und die Felddefinitionen des Registertyps.

- [PAGE02]

  In diesem Abschnitt finden Sie Dokumentendaten, die Verschlagwortung des Dokuments und die Felddefinitionen des Dokumenttyps.

- [PAGE03]

  In diesem Abschnitt finden Sie die Basisparameterdaten des DMS-Objekts, auf das sich das Event bezieht.

- [GLOBALS]

  Dieser Abschnitt enthält allgemeine Informationen.

Die Nummerierung der Page-Abschnitte entspricht der Datenblattfolge von links nach rechts in enaio® client.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=4
#OSIDENT#=882
#OSMAIN#=0
#OSMODIFIED#=1283779716
#OSPOS000#=2010
#OSPOS001#=Aufträge
FILECOUNT=0
FELD0=#OSPOS000#;Jahr;zahl1;9;4;0;0
FIELDEXT0=#OSPOS000#08B022760BAE042CAB627F8A2945CCJahrzahl1WF_Year94
00
FELD1=#OSPOS001#;Prozessfamilie;feld1;X;50;0;0
FIELDEXT1=#OSPOS001#D70773C5AFDB4894A739C2119093DAA0Prozessfamiliefe
ld1WF_FamilyX5000
#OSNAME#=WF-Protokoll
[PAGE01]
#OSTYPE#=6488071
#OSIDENT#=883
#OSMAIN#=99
#OSMODIFIED#=1283779767
#OSPOS000#=April
FILECOUNT=0
FELD0=#OSPOS000#;Monat;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#D622CD2C43D3490E87F36AF9ADF660D5Monatfeld1WF_Mon
thX5000
#OSNAME#=Allgemein
[PAGE02]
#OSTYPE#=131115
#OSIDENT#=884
#OSMAIN#=2
#OSMODIFIED#=1326111543
#OSFIELDMODE#=1
#OSGUID#49B4CEA9BADF49BlA3AEAC193ABC6093=Auftrag Eingang
#OSPOS000#=Auftrag Eingang
#OSGUID#DBFOC6B60C5F44A3AB37F92C8BC538A4=neu
#OSPOS001#=neu
#OSGUID#C5053863FA2D46B4B19C13BCAOC6D30C=324543
#OSPOS002#=324543
#OSACT#=1
FILECOUNT=0
FELD0=#OSPOS000#;Prozessname;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#049B4CEA9BADF49BlA3AEAC193ABC6Prozessnamefeld1WF
_ProcessNameX5000
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000;0;0
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538ABemerkungfeld2WF_
CommentX100000
FELD2=#OSPOS002#;ProzessId;feld4;X;1000;0;0
FIELDEXT2=#OSPOS002#OC5053863FA2D46B4B19C13BCAOC6D3ProzessIdfeld4WF_
ProcessIdX100000
FELD3=#OSPOS003#;Event;;K;0;0;0
FIELDEXT3=#OSPOS003#11DF4AlE9AF84A8682340807127406180EventK000
#OSNAME#=Protokoll
#OSFOREIGNID#=0
[PAGE03]
#OSTYPE#=6553600
#OSIDENT#=884
#OSMAIN#=100
#OSMODIFIED#=0
#OSPOS000#=THOMAS
#OSPOS001#=1280227291
#OSPOS003#=ADMINISTRATOR
#OSPOS004#=1326111543
#OSPOS007#=A402C5EDF25744DBA04A622986E15042
#OSPOS006#=72
FILECOUNT=0
```

```
[GLOBALS]
EventCode=30
Action=UPDATE
Handle=67004
OrdIdent=882
OrdType=4
RegIdent=883
RegType=6488071
TargetMainType=-1
```

Wird beispielsweise das Event `OnClickItem` aus einer Suchmaske ausgeführt, die im Feld 'Prozessname' mit dem Suchbegriff 'Auftrag Eingang' und im Feld 'ProzessID' mit dem Suchbegriff '324543' gefüllt ist, finden Sie im Abschnitt [GLOBALS] den Eintrag 'Action=REQUEST' an Stelle von 'Action=UPDATE':

```
[PAGE00]
#OSTYPE#=131115
#OSIDENT#=0
#OSMAIN#=2
#OSMODIFIED#=0
#OSFIELDMODE#=1
#OSGUID#49B4CEA9BADF49B1A3AEAC193ABC6093=Auftrag Eingang
#OSPOS000#=Auftrag Eingang
#OSGUID#DBFOC6B60C5F44A3AB37F92C8BC538A4=neu
#OSPOS002#=neu
#OSACT#=1
FILECOUNT=0
FELD0=#OSPOS000#;Prozessname;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#049B4CEA9BADF49B1A3AEAC193ABC6Prozessnamefeld1WF
_ProcessNameX5000
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000;0;0
FIELDEXT1=#OSPOS001#DDBFOC6B60C5F44A3AB37F92C8BC538Bemerkungfeld2WF_
CommentX100000
FELD2=#OSPOS002#;ProzessId;feld4;X;1000;0;0
FIELDEXT2=#OSPOS002#DC5053863FA2D46B4B19C13BCAOC6D3ProzessIdfeld4WF_
ProcessIdX100000
FELD3=#OSPOS003#;Event;;K;0;0;0
FIELDEXT3=#OSPOS003#D11DF4AlE9AF84A868234080712740618DEventK000
#OSNAME#=Protokoll
[GLOBALS]
EventCode=0
Action=REQUEST
Handle=3408762
OrdIdent=-1
OrdType=-1
RegIdent=-1
RegType=-1
```

enaio® client benötigt für das Event `OnClickItem` einen Rückgabewert:

- 'resultcode=0' überträgt die Daten aus der Übergabedatei in die Verschlagwortungsmaske,
- 'resultcode=1' ändert die Daten der Verschlagwortungsmaske nicht.

Mit der Methode `closedatamask` im `OnClickItem` können DMS-Masken von DMS-Objekt per Skriptaufruf geschlossen werden. Folgende Rückgabewerte stehen zur Verfügung:

- '-1' Änderungen verwerfen und DMS-Maske schließen
- '0' (Standard) DMS-Maske nicht schließen
- '1' Änderungen speichern und DMS-Maske schließen

Den Rückgabewert müssen Sie explizit in die Übergabedatei schreiben.

```
WriteProfString "GLOBALS","closedatamask","<Rückgabewert>",OSFILE
asfile.WriteToFile()
```

Über die Eigenschaft 'Enabled=true' ist es möglich, mit einem Event-Skript eine schreibgeschützte Schaltfläche einer Datenmaske oder bei einem `OnShow`-Event zu aktivieren, um beispielsweise zugehörige Dokumente zu öffnen.

Voraussetzungen sind, dass im Event-Skript eine Event-Schaltfläche aktiviert werden kann und das Datenblatt keine PageControls besitzt.

Beispiel:

```
if AsFile.EventAction = "READONLY" then
    MsgBox "try to activate Schaltflaeche"
    ActivePage.ASFields.Item("&Schaltfläche").Enabled=true
end if
ResultCode=1
WriteToFile()
```

Aus dem Eventskript kann statt des aktuellen Objekts ein Objekt über die Objekt-ID in der Inhalts-/Detaivorschau bzw. Dashlets angezeigt werden:

```
asfile.ContextObjIdent = ID
```

OnShow

Das Event lösen Sie durch Öffnen eines Datenblatts, einer Suchmaske oder beim Neuerstellen eines DMS-Objekts aus.

enaio® client erzeugt beispielsweise eine Übergabedatei mit Daten zum Ordner [PAGE00], zum Register [PAGE01], zum Dokument [PAGE02] und zu den Basisparametern [PAGE03].

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=4
#OSIDENT#=882
#OSMAIN#=0
#OSMODIFIED#=1283779716
#OSPOS000#=2010
#OSPOS001#=AdHoc V 1.02
FILECOUNT=0
FELD0=#OSPOS000#;Jahr;zahl1;9;4;0;0
FIELDEXT0=#OSPOS000#18A1529714954419924950FFFD930FDEJahrzahl1WF_Year
9400
FELD1=#OSPOS001#;Prozessfamilie;feld1;X;50;0;0
FIELDEXT1=#OSPOS001#8B022760BAE042CAB627F8A2945CC044Prozessfamiliefe
ld1WF_FamilyX5000
#OSNAME#=WF-Protokoll
[PAGE01]
#OSTYPE#=6488071
#OSIDENT#=883
#OSMAIN#=99
#OSMODIFIED#=1283779767
#OSPOS000#=April
FILECOUNT=0
FELD0=#OSPOS000#;Monat;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#D622CD2C43D3490E87F36AF9ADF660D5Monatfeld1WF_Mon
thX5000
#OSNAME#=Allgemein
[PAGE02]
#OSTYPE#=131115
#OSIDENT#=884
#OSMAIN#=2
#OSMODIFIED#=1326113515
#OSFIELDMODE#=1
#OSGUID#49B4CEA9BADF49BlA3AEAC193ABC6093=Auftrag Eingang
#OSPOS000#=Auftrag Eingang
#OSGUID#DBFOC6B60C5F44A3AB37F92C8BC538A4=neu
#OSPOS001#=neu
#OSGUID#C5053863FA2D46B4B19C13BCAOC6D30C=324543
#OSPOS002#=324543
#OSACT#=1
FILECOUNT=0
FELD0=#OSPOS000#;Prozessname;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX5000
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000;0;0
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX100000
FELD2=#OSPOS002#;ProzessId;feld4;X;1000;0;0
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX100000
FELD3=#OSPOS003#;Event;;K;0;0;0
FIELDEXT3=#OSPOS003#11DF4AlE9AF84A868234080712740618EventK000
#OSNAME#=Protokoll
#OSFOREIGNID#=0
[PAGE03]
#OSTYPE#=6553600
#OSIDENT#=884
#OSMAIN#=100
#OSMODIFIED#=0
FILECOUNT=0
[GLOBALS]
EventCode=1
Action=UPDATE
Handle=1442014
OrdIdent=882
OrdType=4
```

```
RegIdent=883
RegType=6488071
CANRESETFIELDS=1
CANINSERTFIELDS=1
```

Legen Sie ein DMS-Objekt neu an, wird ebenfalls das Event `OnShow` ausgelöst. Im Abschnitt [GLOBALS] finden Sie dann den Eintrag 'Action=NEW'. Legen Sie ein Dokument per Drag&Drop an, finden Sie den Pfad zur Datei über den Eintrag 'EXTERNDROPFILE'. Öffnen Sie die Suchmaske für den DMS-Objekttyp, lautet der Eintrag 'Action=REQUEST'. Wird ein Datenblatt schreibgeschützt geöffnet, lautet der Eintrag 'Action=READONLY'. Skripte, die diese unterschiedlichen Action-Modi nicht beachten, führen zu Fehlern.

Wird ein Objekt durch Kopieren erzeugt, enthält die Übergabedatei einen zusätzlichen Eintrag, der die ID des Quellobjekts enthält. Der Eintrag hat folgende Struktur: 'CopyFrom=ID'

Wird ein Objekt aus einem Register geöffnet, enthält die Übergabedatei die Standortangabe. Der Eintrag hat folgende Struktur:

```
[GLOBALS]
EventLocationFolderIdent=81
EventLocationFolderType=0
EventLocationRegisterIdent=293
EventLocationRegisterType=6488065
```

Wird ein Objekt aus einem Ordner auf oberster Ebene geöffnet, enthält die Übergabedatei die Standortangabe in folgender Struktur:

```
[GLOBALS]
EventLocationFolderIdent=81
EventLocationFolderType=0
EventLocationRegisterIdent=4294967295
EventLocationRegisterType=4294967295
```

Wird ein Objekt aus einem Standort geöffnet, der kein Ordner ist, enthält die Übergabedatei die folgende Struktur:

```
[GLOBALS]
EventLocationFolderIdent=4294967295
EventLocationFolderType=4294967295
EventLocationRegisterIdent=4294967295
EventLocationRegisterType=4294967295
```

4294967295 bedeutet 'keine Angabe'.

enaio® client benötigt einen Rückgabewert für das Event `OnShow`:

'resultcode=1' überträgt die Daten aus der Übergabedatei in die Verschlagwortungsmaske,

'resultcode=0' ändert die Daten nicht,

'resultcode=-1' bricht das Öffnen der Verschlagwortungsmaske ab.

Mit dem Event `OnShow` können Sie beispielsweise die Eigenschaft 'Dialogelement sichtbar' ändern:

```
Activepage.AsFields.Item("Bemerkung").Visible = false
ResultCode=1
WriteToFile()
```

Analog können Sie über die Eigenschaft 'Enabled=False' einen Schreibschutz für einzelne Felder einrichten.

Schreibgeschützte Schaltflächen auf einer Datenmaske oder bei einem `OnShow`-Event können über die Eigenschaft auch aktiviert werden, um beispielsweise zugehörige Dokumente zu öffnen.

Voraussetzungen sind, dass im Event-Skript eine Event-Schaltfläche aktiviert werden kann und das Datenblatt keine PageControls besitzt.

Beispiel:

```
if AsFile.EventAction = "READONLY" then
    MsgBox "try to activate Schaltflaeche"
    ActivePage.ASFields.Item("&Schaltfläche").Enabled=true
end if
ResultCode=1
WriteToFile()
```

Ist ein Datenblatt über das AddOn 'Quickfinder' geöffnet worden, enthält die Übergabedatei im Abschnitt [GLOBALS] folgende Daten:

`QUICKFINDER=1`	Informiert, dass das Datenblatt über das AddOn 'Quickfinder' geöffnet wurde.
`QUICKPARENTOBJECT=Objekttyp-ID`	Gibt die ID des Dokumenttyps an, über den das AddOn 'Quickfinder' ausgeführt wurde.

Die Einträge 'CANRESETFIELDS=1' und 'CANINSERTFIELDS=1' im Abschnitt [GLOBALS] beziehen sich auf die Kontextmenüfunktionen 'Zurücksetzen' und 'Einfügen' auf einer Maske. 'Zurücksetzen' leert alle Felder, 'Einfügen' fügt kopierte Einträge eines anderen Objekts ein.

Wenn diese Funktionen nicht zur Verfügung stehen sollen, kann jeweils der Wert auf '0' gesetzt werden. Dann sind diese Funktionen deaktiviert.

Aus dem Eventskript kann statt des aktuellen Objekts ein Objekt über die Objekt-ID in der Inhalts-/Detaivorschau bzw. Dashlets angezeigt werden:

```
asfile.ContextObjIdent = ID
```

BeforeValidate

Das Event wird beim Ändern der Verschlagwortung durch die Funktion 'Speichern' auf der Verschlagwortungsmaske eines DMS-Objekts, hier ein Dokument vom Typ 'Protokoll', ausgelöst. Beim 'Speichern' eines neu angelegten DMS-Objekts wird das Event ebenfalls ausgeführt, dann finden Sie im Abschnitt [GLOBALS] den Eintrag 'Action=NEW' statt 'Action=UPDATE'.

enaio® client erzeugt eine Übergabedatei mit Daten zum Ordner [PAGE00], zum Register [PAGE01], zum Dokument [PAGE02] und zu den Basisparametern [PAGE03].

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=4
#OSIDENT#=882
#OSMAIN#=0
#OSMODIFIED#=1283779767
#OSPOS000#=2010
#OSPOS001#=Aufträge
FILECOUNT=0
FELD0=#OSPOS000#;Jahr;zahl1;9;4;0;0
FIELDEXT0=#OSPOS000#18A1529714954419924950FFFD930FDEJahrzahl1WF_Year
9400
FELD1=#OSPOS001#;Prozessfamilie;feld1;X;50;0;0
FIELDEXT1=#OSPOS001#8B022760BAE042CAB627F8A2945CC044Prozessfamiliefe
ld1WF_FamilyX5000
#OSNAME#=WF-Protokoll
[PAGE01]
#OSTYPE#=6488071
#OSIDENT#=883
#OSMAIN#=99
#OSMODIFIED#=1283779767
#OSPOS000#=April
FILECOUNT=0
FELD0=#OSPOS000#;Monat;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#D622CD2C43D3490E87F36AF9ADF660D5Monatfeld1WF_Mon
thX5000
#OSNAME#=Allgemein
[PAGE02]
#OSTYPE#=131115
#OSIDENT#=884
#OSMAIN#=2
#OSMODIFIED#=1326113973
#OSFIELDMODE#=1
#OSGUID#49B4CEA9BADF49BlA3AEAC193ABC6093=Auftrag Eingang
#OSPOS000#=Auftrag Eingang
#OSGUID#DBFOC6B60C5F44A3AB37F92C8BC538A4=neu und bearbeitet
#OSPOS001#=neu und bearbeitet
#OSGUID#C5053863FA2D46B4B19C13BCAOC6D30C=324543
#OSPOS002#=324543
#OSACT#=1
FILECOUNT=1
FELD0=#OSPOS000#;Prozessname;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX5000
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000;0;0
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX100000
FELD2=#OSPOS002#;ProzessId;feld4;X;1000;0;0
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX100000
FELD3=#OSPOS003#;Event;;K;0;0;0
FIELDEXT3=#OSPOS003#11DF4AlE9AF84A868234080712740618EventK000
#OSNAME#=Protokoll
#OSFOREIGNID#=0
[PAGE03]
#OSTYPE#=6553600
#OSIDENT#=884
#OSMAIN#=100
#OSMODIFIED#=0
#OSPOS000#=THOMAS
#OSPOS001#=1280227291
#OSPOS003#=ADMINISTRATOR
#OSPOS004#=1326113973
#OSPOS007#=A402C5EDF25744DBA04A622986E15042
#OSPOS006#=72
FILECOUNT=0
```

```
[GLOBALS]
EventCode=2
Action=UPDATE
Handle=590516
OrdIdent=882
OrdType=4
RegIdent=883
RegType=6488071
TargetMainType=-1
```

Wird ein Objekt durch Kopieren erzeugt, enthält die Übergabedatei einen zusätzlichen Eintrag, der die ID des Quellobjekts enthält. Der Eintrag hat folgende Struktur: 'CopyFrom=ID'

enaio® client benötigt für das Event einen Rückgabewert:

'resultcode=1' speichert die Daten aus der Übergabedatei in der Verschlagwortungsmaske.

'resultcode=0' Daten werden nicht geändert.

'resultcode=-1' übernimmt keine Daten und lässt die Verschlagwortungsmaske offen.

'resultcode=-2' überträgt die Daten aus der Übergabedatei in die Verschlagwortungsmaske, die Verschlagwortungsmaske bleibt in enaio® client mit den geänderten Daten offen.

enaio® client prüft die Daten und lässt bei 'resultcode=1' die Verschlagwortungsmaske mit entsprechenden Benutzerhinweisen offen, wenn Daten nicht den Vorgaben entsprechen.

Die Katalogprüfung kann abgeschaltet werden. Dazu schreiben Sie im Abschnitt [GLOBALS] den Eintrag `CHECKCATALOGVALUES=0`:

```
oxhelp.writeprofstring "GLOBALS", "CHECKCATALOGVALUES", 0, osfile
Resultcode = 1
WriteToFile()
```

AfterValidate

Das Event entspricht BeforeValidate, wird aber nach der Plausibilitätsprüfung des enaio® client ausgeführt.

enaio® client benötigt für das Event einen Rückgabewert:

'resultcode=1' speichert die Daten aus der Übergabedatei in der Verschlagwortungsmaske ohne weitere Validierung durch den enaio® client.

'resultcode=0' Daten werden nicht geändert.

AfterSave

Das Event wird nach dem Speichern der Daten eines Datenblatts ausgeführt. enaio® client erwartet keine Rückgabe und liest die Daten aus der Übergabedatei nicht aus.

Die Übergabedatei enthält Daten zum Ordner [PAGE00], zum Dokument [PAGE01] und zu den Basisparametern [PAGE02]. Das Dokument liegt nicht in einem Register.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=4
#OSIDENT#=882
#OSMAIN#=0
#OSMODIFIED#=1283779716
#OSPOS000#=2010
#OSPOS001#=Aufträge
FILECOUNT=0
FELD0=#OSPOS000#;Jahr;zahl1;9;4;0;0
FIELDEXT0=#OSPOS000#18A1529714954419924950FFFD930FDEJahrzahl1WF_Year
9400
FELD1=#OSPOS001#;Prozessfamilie;feld1;X;50;0;0
FIELDEXT1=#OSPOS001#8B022760BAE042CAB627F8A2945CC044Prozessfamiliefe
ld1WF_FamilyX5000
#OSNAME#=WF-Protokoll
[PAGE01]
#OSTYPE#=6488071
#OSIDENT#=883
#OSMAIN#=99
#OSMODIFIED#=1283779767
#OSPOS000#=April
FILECOUNT=0
FELD0=#OSPOS000#;Monat;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#D622CD2C43D3490E87F36AF9ADF660D5Monatfeld1WF_Mon
thX5000
#OSNAME#=Allgemein
[PAGE02]
#OSTYPE#=131115
#OSIDENT#=884
#OSMAIN#=2
#OSMODIFIED#=1326114350
#OSFIELDMODE#=1
#OSGUID#49B4CEA9BADF49BlA3AEAC193ABC6093=Auftrag Eingang
#OSPOS000#=Auftrag Eingang
#OSGUID#DBFOC6B60C5F44A3AB37F92C8BC538A4=neu und bearbeitet
#OSPOS001#=neu und bearbeitet
#OSGUID#C5053863FA2D46B4B19C13BCAOC6D30C=324543
#OSPOS002#=324543
#OSACT#=1
FILECOUNT=1
FELD0=#OSPOS000#;Prozessname;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX5000
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000;0;0
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX100000
FELD2=#OSPOS002#;ProzessId;feld4;X;1000;0;0
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX100000
FELD3=#OSPOS003#;Event;;K;0;0;0
FIELDEXT3=#OSPOS003#11DF4AlE9AF84A868234080712740618EventK000
#OSNAME#=Protokoll
#OSFOREIGNID#=0
[PAGE03]
#OSTYPE#=6553600
#OSIDENT#=884
#OSMAIN#=100
#OSMODIFIED#=0
#OSPOS000#=THOMAS
#OSPOS001#=1280227291
#OSPOS003#=ADMINISTRATOR
#OSPOS004#=1326114350
#OSPOS007#=A402C5EDF25744DBA04A622986E15042
#OSPOS006#=72
```

```
FILECOUNT=0
[GLOBALS]
EventCode=2
Action=UPDATE
Handle=458922
OrdIdent=882
OrdType=4
RegIdent=883
RegType=6488071
TargetMainType=-1
```

BeforeStartQuery

Das Event wird durch die Funktion 'Anfrage starten' aus einer Suchmaske ausgeführt. Die Übergabedatei enthält die Daten der Suchmasken. Diese Daten können Sie über das VB-Skript ändern.

enaio® client liest die Daten aus der Übergabedatei in die Suchmasken ein und startet die Suche, wenn Sie 'resultcode=1' in die Übergabedatei schreiben.

'resultcode=0' startet die Suche ohne Änderungen.

'resultcode=-1' führt zum Abbruch der Suche.

Die Übergabedatei enthält Daten zu den Suchmasken und zu den eingegebenen Suchbegriffen. Im Beispiel wurde der Suchbegriff '324543' in das Feld 'ProzessID' der Suchmaske zum Dokument eingetragen und der Suchbegriff '2010' in das Feld 'Jahr' der Ordner-Suchmaske.

Die Nummerierung der Page-Abschnitte entspricht bei kombinierten Anfragen der Datenblattfolge von links nach rechts in enaio® client. Das Event ist einem DMS-Objekt zugeordnet und wird nur ausgeführt, wenn das Datenblatt dieses DMS-Objekts bei kombinierten Anfragen aktiv ist.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
#OSIDENT#=0
#OSMAIN#=2
#OSMODIFIED#=0
#OSFIELDMODE#=1
#OSGUID#C5053863FA2D46B4B19C13BCAOC6D30C=324543
#OSPOS002#=324543
#OSACT#=1
FILECOUNT=0
FELD0=#OSPOS000#;Prozessname;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX5000
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000;0;0
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX100000
FELD2=#OSPOS002#;ProzessId;feld4;X;1000;0;0
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX100000
FELD3=#OSPOS003#;Event;;K;0;0;0
FIELDEXT3=#OSPOS003#11DF4AlE9AF84A868234080712740618EventK000
#OSNAME#=Protokoll
[PAGE01]
#OSTYPE#=6488071
#OSIDENT#=884
#OSMAIN#=99
#OSMODIFIED#=1283779716
#OSFIELDMODE#=1
FILECOUNT=0
FELD0=#OSPOS000#;Monat;feld1;X;50;0;0
FIELDEXT0=#OSPOS000#D622CD2C43D3490E87F36AF9ADF660D5Monatfeld1WF_Mon
thX5000
#OSNAME#=Allgemein
[PAGE02]
#OSTYPE#=131115
#OSIDENT#=884
#OSMAIN#=2
#OSMODIFIED#=1326114350
#OSFIELDMODE#=1
#OSGUID#DDF9D11554884EA6BA217F99189CDF01=2010
#OSPOS000#=2010
FILECOUNT=0
FELD0=#OSPOS000#;Jahr;zahl1;9;4;0;0
FIELDEXT0=#OSPOS000#DDF9D11554884EA6BA217F99189CDF01Jahrzahl1WF_Year
9400
FELD1=#OSPOS001#;Prozessfamilie;feld1;X;50;0;0
FIELDEXT1=#OSPOS001#D70773C5AFDB4894A739C2119093DAA0Prozessfamiliefe
ld1WF_FamilyX5000
#OSNAME#=WF-Protokoll
[GLOBALS]
EventCode=4
Action=
Handle=1905136
OrdIdent=-1
OrdType=-1
RegIdent=-1
RegType=-1
```

BeforeOpen

Das Event wird ausgeführt, nachdem Sie für ein Dokument die Funktion 'Öffnen' ausgeführt haben. Die Übergabedatei enthält die Daten der Verschlagwortung des Dokuments, aber keine weiteren Daten wie beispielsweise Basisparameter.

enaio® client öffnet das Dokument, wenn Sie 'resultcode=1' in die Übergabedatei schreiben.

'resultcode=0' öffnet das Dokument nicht. Ohne Rückgabewert wird das Dokument ebenfalls nicht geöffnet.

'resultcode=-4' beinhaltet als Rückgabewert die Objekt-ID und den Objekt-Typ. Sobald zwischen verschiedenen Workflow-Vorgängen, Trefferlisten und geöffneten Standorten gewechselt wird, bleibt das Dokument im Dokument-Viewer angezeigt.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
FILECOUNT=1
#OSPOS000#=Auftrag Eingang
FELD0=#OSPOS000#;Prozessname;feld1;X;50
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX50
#OSPOS001#=neu und bearbeitet
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX1000
#OSPOS002#=324543
FELD2=#OSPOS002#;ProzessId;feld4;X;1000
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX1000
#OSIDENT#=884
#OSNAME#=Protokoll
[GLOBALS]
EventCode=16
Action=READONLY
OrdIdent=882
OrdType=4
RegIdent=883
RegType=6488071
CHECKOUT=0
```

BeforeDelete

Das Event wird ausgeführt, nachdem Sie für ein Dokument, Ordner oder Register die Funktion 'Löschen' ausgeführt haben. Die Übergabedatei enthält die Daten der Verschlagwortung des DMS-Objekts, aber keine weiteren Daten wie beispielsweise Basisparameter.

enaio® client löscht das Dokument, wenn Sie 'resultcode=1' in die Übergabedatei schreiben.

'resultcode=0' löscht das Dokument nicht.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
FILECOUNT=1
#OSPOS000#=Auftrag Eingang
FELD0=#OSPOS000#;Prozessname;feld1;X;50
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX50
#OSPOS001#=kopiert
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX1000
#OSPOS002#=324543
FELD2=#OSPOS002#;ProzessId;feld4;X;1000
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX1000
#OSIDENT#=2189
#OSNAME#=Protokoll
[GLOBALS]
EventCode=11
OrdIdent=0
OrdType=4
```

BeforeUndoCheckOut

Das Event wird ausgeführt, nachdem Sie für ein oder mehrere Dokumente die Funktion 'Auschecken zurücknehmen' ausgeführt haben. Die Übergabedatei enthält die Daten der Verschlagwortung des Dokuments, aber keine weiteren Daten wie beispielsweise Basisparameter.

Bei 'resultcode=1' wird das Auschecken ohne Rückfrage des Benutzer rückgängig gemacht.

Der enaio® client nimmt das Auschecken des Dokuments zurück, wenn Sie 'resultcode=0' in die Übergabedatei schreiben.

Sind mehrere Dokumente markiert, wird bei 'resultcode=-1' das Auschecken für alle markierten Dokumente nicht rückgängig gemacht.

'resultcode=-2' nimmt das Auschecken des aktuellen Dokuments nicht zurück.

Über die Konstante 'NumberOfSelectedDocuments' können Sie im Skript abfragen, für wie viele Dokumente das Auschecken zurückgenommen werden soll.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
FILECOUNT=1
#OSPOS000#=Auftrag Eingang
FELD0=#OSPOS000#;Prozessname;feld1;X;50
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX50
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX1000
#OSPOS002#=324543
FELD2=#OSPOS002#;ProzessId;feld4;X;1000
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX1000
#OSIDENT#=2189
#OSNAME#=Protokoll
[GLOBALS]
EventCode=27
OrdIdent=882
OrdType=4
RegIdent=883
RegType=6488071
```

AfterDelete

Das Event wird ausgeführt, nachdem Sie ein Dokument, Ordner oder Register gelöscht haben. Die Übergabedatei enthält die Daten der Verschlagwortung des DMS-Objekts, aber keine weiteren Daten wie beispielsweise Basisparameter.

enaio® client erzeugt die Übergabedatei, führt das VB-Skript aus, liest aber keine Daten aus der Übergabedatei.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=4
FILECOUNT=0
#OSPOS000#=2010
FELD0=#OSPOS000#;Jahr;zahl1;9;4
FIELDEXT0=#OSPOS000#18A1529714954419924950FFFD930FDEJahrzahl1WF_Year
9400
FELD1=#OSPOS001#;Prozessfamilie;feld1;X;50
FIELDEXT1=#OSPOS001#8B022760BAE042CAB627F8A2945CC044Prozessfamiliefe
ld1WF_FamilyX5000
#OSIDENT#=1805
#OSNAME#=WF-Protokoll
[GLOBALS]
EventCode=12
```

OnMove

Das Event wird ausgeführt, wenn Sie ein DMS-Objekt innerhalb eines Schranks an einen anderen Standort verschieben. Die Übergabedatei enthält die Daten der Verschlagwortung des DMS-Objekts.

Im Abschnitt [MOVEINFO] stehen Informationen zum Ursprungs- und Zielort. Den Zielort können Sie beispielsweise durch das VB-Skript überprüfen und ändern.

- SOURCEINFO

  ```
  SOURCEINFO=OrdnerID,OrdnerTyp,RegisterID,RegisterTyp
  ```

- DESTINFO

  ```
  DESTINFO=OrdnerID,RegisterID
  ```

Das Verschieben wird mit den Daten aus der Übergabedatei ausgeführt, wenn Sie 'resultcode=1' zurückschreiben.

'resultcode=-1' führt das Verschieben nicht aus.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
FILECOUNT=1
#OSPOS000#=Auftrag Eingang
FELD0=#OSPOS000#;Prozessname;feld1;X;50
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX50
#OSPOS001#=neuer Auftrag Januar
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX1000
#OSPOS002#=334543
FELD2=#OSPOS002#;ProzessId;feld4;X;1000
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX1000
#OSIDENT#=2196
#OSNAME#=Protokoll
[GLOBALS]
EventCode=17
OrdIdent=2190
OrdType=4
RegIdent=0
RegType=0
[MOVEINFO]
SOURCEINFO=2190,4,0,0
DESTINFO=2190,2193
DESTINFO2=2190,2193,6488071
```

OnMoveExtern

Das Event wird ausgeführt, wenn Sie ein DMS-Objekt in einen anderen Schrank verschieben. Die Übergabedatei enthält die Daten der Verschlagwortung des DMS-Objekts und entspricht der des Events 'OnMove'.

OnAddLocation

Das Event wird ausgeführt, wenn Sie ein Register oder Dokument per Drag & Drop einen weiteren Standort erhält.

Die Übergabedatei enthält die Daten der Verschlagwortung des DMS-Objekts.

Im Abschnitt [COPYINFO] stehen Informationen zum Ursprungs- und Zielort. Den Zielort können Sie beispielsweise durch das VB-Skript überprüfen und ändern.

- SOURCEINFO

  ```
  SOURCEINFO=OrdnerID,OrdnerTyp,RegisterID,RegisterTyp
  ```

- DESTINFO

  ```
  DESTINFO=OrdnerID,RegisterID
  ```

Das Hinzufügen eines Standorts wird mit den Daten aus der Übergabedatei ausgeführt, wenn Sie 'resultcode=1' zurückschreiben.

'resultcode=-1' führt das Hinzufügen eines Standorts nicht aus.

Beispiel für Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
FILECOUNT=1
#OSPOS000#=Auftrag Bearbeitung
FELD0=#OSPOS000#;Prozessname;feld1;X;50
FIELDEXT0=#OSPOS000#49B4CEA9BADF49BlA3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX50
#OSPOS001#=Auftrag wird jetzt weiter bearbeitet
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000
FIELDEXT1=#OSPOS001#DBFOC6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX1000
#OSPOS002#=3345439
FELD2=#OSPOS002#;ProzessId;feld4;X;1000
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCAOC6D30CProzessIdfeld4WF
_ProcessIdX1000
#OSIDENT#=2197
#OSNAME#=Protokoll
[GLOBALS]
EventCode=18
OrdIdent=2190
OrdType=4
RegIdent=2194
RegType=6488071
[COPYINFO]
SOURCEINFO=2190,4,2194,6488071
DESTINFO=2190,2195
DESTINFO2=2190,2195,6488071
```

OnCreateCopy

Das Event wird ausgeführt, wenn ein Register oder Dokument kopiert wird. Die Übergabedatei enthält die Daten der Verschlagwortung des DMS-Objekts und entspricht der des Events 'OnAddLocation'.

BeforeLink

Das Event wird ausgeführt, wenn Sie über das Notiz-Fenster und den Notizbereich des Ordner-Fensters eine Verbindung oder eine Relation anlegen.

Die Übergabedatei enthält die Daten der Verschlagwortung der beiden DMS-Objekte in den Abschnitten [PAGE00] und [PAGE01].

Im Abschnitt [GLOBALS] sind über folgende Struktur die verbundenen Objekte angegeben:

```
[GLOBALS]
EventCode=21
Action=BEFORELINK
LINKOBJECTID1=1868
LINKOBJECTTYPE1=4
LINKOBJECTID2=1808
LINKOBJECTTYPE2=4
```

Bei Relationen sind ebenfalls die Relationsdaten angegeben.

Bei 'resultcode=2' wird die Relation nicht erstellt und der Relationsdialog geschlossen.

Bei Relationen wird durch 'resultcode=1' die Relation nicht erstellt, der Relationsdialog aber offen gehalten.

Schreiben Sie 'resultcode=0' zurück, wird die Verknüpfung oder Relation erstellt.

Für Verknüpfungen führen andere Werte als '0' zum Abbruch. Die Verknüpfung wird nicht erstellt.

BeforeDeleteLink

Das Event wird ausgeführt, wenn Sie über das Notiz-Fenster und den Notizbereich des Ordner-Fensters eine Verbindung oder eine Relation löschen.

Die Übergabedatei entspricht der Übergabedatei beim Event 'BeforeLink'.

Schreiben Sie 'resultcode=0' zurück, wird die Verknüpfung oder Relation gelöscht.

Andere Werte führen zum Abbruch, die Verbindung oder Relation wird nicht gelöscht.

AfterLink

Das Event wird ausgeführt, nachdem Sie eine Verbindung oder eine Relation angelegt haben.

Die Übergabedatei entspricht der Übergabedatei beim Event 'BeforeLink'. enaio® client liest keine Daten zurück.

AfterDeleteLink

Das Event wird ausgeführt, nachdem Sie eine Verbindung oder eine Relation gelöscht haben.

Die Übergabedatei entspricht der Übergabedatei beim Event 'BeforeLink'. enaio® client liest keine Daten aus der Übergabedatei.

BeforeSaveDocument

Das Event wird ausgeführt, bevor Sie ein Dokument eingecheckt haben.

Ausgecheckte Dokumente werden ebenfalls eingecheckt, wenn enaio® client beendet wird. Auch dann wird das Event ausgeführt.

Die Übergabedatei enthält im Abschnitt [PAGE00] die Verschlagwortungsdaten des Dokuments. Angegeben sind ebenfalls Bezeichnung und Pfad der Dokumentdateien im lokalen Cachebereich des Arbeitsplatzes.

Beispiel:

```
FILE0=...\LOKALE~1\TEMP\OSTEMP\00000791\CACHE\03\11\2D\00000C2D.000
FILE1=...\LOKALE~1\TEMP\OSTEMP\00000791\CACHE\03\11\2D\00000C2D.001
```

Im Abschnitt [GLOBALS] sind zusätzlich ID und Typ für den Ordner und das Register angegeben.

Schreiben Sie 'resultcode=0' zurück, wird das Dokument eingecheckt.

Schreiben Sie 'resultcode=-2', wird komplett abgebrochen.

Bei anderen Rückgabewerten wird das aktuelle Dokument nicht eingecheckt, sondern mit dem folgenden Dokument fortgefahren.

AfterSaveDocument

Das Event wird ausgeführt, nachdem Sie ein Dokument eingecheckt haben.

Die Übergabedatei entspricht der Übergabedatei beim Event 'BeforeSaveDocument'. Bezeichnung und Pfad der Dokumentdateien sind nicht enthalten, wurde eine Datei per Drag&Drop übernommen, ist der Ausgangspfad zu dieser Datei als Wert von 'EXTERNDROPFILE' in der Übergabedatei angegeben.

enaio® client liest keine Daten aus der Übergabedatei.

BeforeCancel

Das Event wird ausgeführt, nachdem auf einem nicht schreibgeschützten Datenblatt auf den Button **Abbrechen** geklickt wurde.

Die Übergabedatei entspricht der Übergabedatei beim Event 'OnShow'.

'resultcode=0' führt dazu, dass das Datenblatt offen bleibt.

Alle anderen Rückgabewerte schließen das Datenblatt.

BeforeRestore

Das Event wird ausgeführt, wenn im Papierkorb ein Objekt markiert ist und auf den Button **Wiederherstellen** geklickt wird.

'resultcode=0' bricht das Wiederherstellen ab.

Alle anderen Rückgabewerte führen zum Wiederherstellen.

Die Übergabedatei enthält einen Page-Abschnitt mit den Daten des Objekts. Im Global-Abschnitt sind gegebenenfalls Ordner und Register des ehemaligen Standorts über die ID angegeben.

Beispiel für eine Übergabedatei:

```
[PAGE00]
#OSTYPE#=131115
FILECOUNT=1
#OSPOS000#=Auftrag Eingang
FELD0=#OSPOS000#;Prozessname;feld1;X;50
FIELDEXT0=#OSPOS000#49B4CEA9BADF49B1A3AEAC193ABC6093Prozessnamefeld1
WF_ProcessNameX50
#OSPOS001#=nicht archiviert
FELD1=#OSPOS001#;Bemerkung;feld2;X;1000
FIELDEXT1=#OSPOS001#DBF0C6B60C5F44A3AB37F92C8BC538A4Bemerkungfeld2WF
_CommentX1000#OSPOS002#=3827982
FELD2=#OSPOS002#;ProzessId;feld4;X;1000
FIELDEXT2=#OSPOS002#C5053863FA2D46B4B19C13BCA0C6D30CProzessIdfeld4WF
_ProcessIdX1000
#OSIDENT#=888
#OSNAME#=Protokoll
[GLOBALS]
EventCode=28
OrdIdent=882
OrdType=4
RegIdent=883
RegType=6488071
```

AfterRestore

Das Event wird ausgeführt, nachdem ein Objekt aus dem Papierkorb wiederhergestellt wurde.

Die Übergabedatei entspricht der Übergabedatei von 'BeforeRestore'.

Rückgaben werden nicht ausgewertet.

OnChangeActivePage

Das Event wird ausgeführt, wenn Sie beim Dialogelement 'Pagecontrol' die aktive Seite gewechselt haben.

Eine Rückgabe von Werten erfolgt nicht. Beim Wechsel auf einer Suchmaske steht im Abschnitt [Globals] `Action=REQUEST`, beim Wechsel auf einem Datenblatt steht dort `Action=UPDATE`.

Der folgende Übergabedatei-Ausschnitt zeigt den grundsätzlichen Aufbau:

```
[PAGE00]
#OSTYPE#=3
#OSIDENT#=0
#OSMAIN#=0
#OSMODIFIED#=0
#OSFIELDMODE#=1
FULLTEXT=
VTREQUESTTYPE=0
#OSACT#=1
FILECOUNT=0
FELD0=#OSPOS000#;SUPFEST;feld22;X;3;0;1
FIELDEXT0=#OSPOS000#12DD22481B2C45848E5BC0CDE959F271SUPFESTfeld22SUP
FESTX301
FELD1=#OSPOS001#;PageCtrl27;;C;0;0;0
FIELDEXT1=#OSPOS001#65DD459E62BE4DE99932ED6EA9753438PageCtrl27C000
#OSNAME#=Kunde
[PAGE00#BD5D78C3F05A45538A226667DC644C01]
#OSTYPE#=3
#OSFIELDMODE#=1
#OSGUID#6BE24A2D207F48FF8CD7965EB840035C=
#OSPOS000#=
#OSGUID#821A6F059F02418F875FB6F2D3496694=
#OSPOS001#=
#OSGUID#173CFBCFBF244B03B0645B005F11F5E0=
#OSPOS002#=
#OSGUID#910C438E76534801B66C12DDDDF73353=
#OSPOS003#=
#OSGUID#7C68B08185704C1DB1AD7C88B17CF529=
#OSPOS004#=
#OSGUID#58FCCDA78AA44B6093B8B2940531E835=
#OSPOS005#=
#OSGUID#8914CE6CE13942D3A6BCFFC6DF1627C2=
#OSPOS006#=
#OSGUID#BD1779FB5DD7496C9A79F91EF4A71517=
#OSPOS007#=
#OSGUID#CE5DFE7C5006443BBC2E0ACF994890AC=
#OSPOS008#=
#OSGUID#AA808EFBA7F04905A48092E30FF92403=
#OSPOS009#=
#OSGUID#EFD8FBAF228241BF983E4F1973375CE7=
#OSPOS010#=
#OSGUID#6FBFBC01EBAC406A8A048997F3C33351=
#OSPOS011#=
#OSACT#=0
#OSENABLED#=1
#FIELDPOS#=#OSPOS001#
#OSNAME#=Kundendaten
FELD0=#OSPOS000#;Firmenname;feld1;X;60;0;0
FIELDEXT0=#OSPOS000#6BE24A2D207F48FF8CD7965EB840035CFirmennamefeld1F
irmennameX6000
FELD1=#OSPOS001#;Zusatz;feld2;X;60;0;0
FIELDEXT1=#OSPOS001#821A6F059F02418F875FB6F2D3496694Zusatzfeld2Zusat
zX6000
FELD2=#OSPOS002#;Straße;feld3;X;40;0;0
FIELDEXT2=#OSPOS002#173CFBCFBF244B03B0645B005F11F5E0Straßefeld3Stras
seX4000
FELD3=#OSPOS003#;Land;feld4;A;3;0;0
FIELDEXT3=#OSPOS003#910C438E76534801B66C12DDDDF73353Landfeld4LandA30
0
FELD4=#OSPOS004#;PLZ;feld5;Z;5;0;0
FIELDEXT4=#OSPOS004#7C68B08185704C1DB1AD7C88B17CF529PLZfeld5PLZZ500
FELD5=#OSPOS005#;Ort;feld6;X;30;0;0
FIELDEXT5=#OSPOS005#58FCCDA78AA44B6093B8B2940531E835Ortfeld6OrtX3000
FELD6=#OSPOS006#;Bundesland;feld7;X;50;0;0
```

```
FIELDEXT6=#OSPOS006#8914CE6CE13942D3A6BCFFC6DF1627C2Bundeslandfeld7B
undeslandX5000
FELD7=#OSPOS007#;Telefon;feld8;X;20;0;0
FIELDEXT7=#OSPOS007#BD1779FB5DD7496C9A79F91EF4A71517Telefonfeld8Tele
fonX2000
FELD8=#OSPOS008#;Fax;feld9;X;20;0;0
FIELDEXT8=#OSPOS008#CE5DFE7C5006443BBC2E0ACF994890ACFaxfeld9TelefaxX
2000
FELD9=#OSPOS009#;E-Mail;feld10;X;80;0;0
FIELDEXT9=#OSPOS009#AA808EFBA7F04905A48092E30FF92403E-
Mailfeld10eMail_AdresseX8000
FELD10=#OSPOS010#;Internet;feld11;X;80;0;0
FIELDEXT10=#OSPOS010#EFD8FBAF228241BF983E4F1973375CE7Internetfeld11I
nternetX8000
FELD11=#OSPOS011#;Klasse;feld29;X;30;0;0
FIELDEXT11=#OSPOS011#6FBFBC01EBAC406A8A048997F3C33351Klassefeld29Kla
sseX3000
[PAGE00#BD85F472CDF646CC82245ECCF4D03EBE]
#OSTYPE#=3
#OSFIELDMODE#=1
#OSGUID#328DD5567B05420893A94090FAE565B0=
#OSPOS000#=
#OSGUID#53A47363F74D4BCD8C944EBD502BE725=
#OSPOS001#=
#OSGUID#EF5B3ACFA2FF418A98048CBECFB2F337=
#OSPOS002#=
#OSGUID#2ED784B358B04467A969EBBCC1995ADC=
#OSPOS003#=
#OSGUID#0003EDBB81654525A9638298CCC4F546=
#OSPOS004#=
#OSGUID#9A2D322CEA694684A0AC210B05F6B848=
#OSPOS005#=
#OSGUID#9504BB059BDB419AAFCF5EC49F59BF0B=
#OSPOS006#=
#OSGUID#136F0270745A4BE480EE89DB0CD4376E=
#OSPOS007#=
#OSACT#=1
#OSENABLED#=1
#FIELDPOS#=#OSPOS001#
#OSNAME#=Beschreibung
FELD0=#OSPOS000#;Branche;feld15;X;20;0;0
FIELDEXT0=#OSPOS000#328DD5567B05420893A94090FAE565B0Branchefeld15Bra
ncheX2000
FELD1=#OSPOS001#;Herkunft;feld16;X;20;0;0
…
```

Die aktive Seite erkennen Sie am Eintrag `#OSACT#=1`.

Die PAGE00, auf der das Pagecontrol eingefügt wurde, wird untergliedert in die definierten Seiten des Pagecontrols, die jeweils eine eigenständige ID erhalten:

- [PAGE00#BD5D78C3F05A45538A226667DC644C01]
- [PAGE00#BD85F472CDF646CC82245ECCF4D03EBE]
- …

FileDrop

Das Event wird nach dem Ablegen von Dateien per Drag&Drop an einem Standort ausgeführt.

'resultcode=-1' bricht die Aktion ab und übernimmt keine Dateien.

'resultcode=1' führt nur zu einer Aktualisierung der Trefferliste.

'resultcode=0' führt zum Ablegen der in der Übergabedatei angegebenen Daten durch enaio® client.

Beispiel für eine Übergabedatei:

```
[GLOBALS]
EventCode=200
OrdIdent=673
OrdType=3
RegIdent=967
RegType=6488070
[FILES]
COUNT=2
FILE1=C:\Users\thomas\Documents\Auftragsbest.tif
FILE2=C:\Users\thomas\Documents\Schreiben.tif
```

Übergabedaten bei Tabellen

Die Daten aus Tabellen von Verschlagwortungsmasken werden in folgender Form in die Übergabedatei geschrieben:

```
[object75list1]
ZEILE0={1234Verkauf567842,13333,522}
ZEILEDB0='1234','Verkauf','5678','42,13','3','33,5','22'
ZEILE1={5678Verkauf356432,673,545,233}
ZEILEDB1='5678','Verkauf','3564','32,67','3,5','45,2','33'
REQFIELDS=feld1X50,feld2X50,feld3X5,feld4X20,feld5X10,feld6X20,feld7
X20
FIELDS=feld1,feld2,feld3,feld4,feld5,feld6,feld7
[LISTCONTROL]
TABLES=object75list1
```

Tabelle der Beispieldatei:

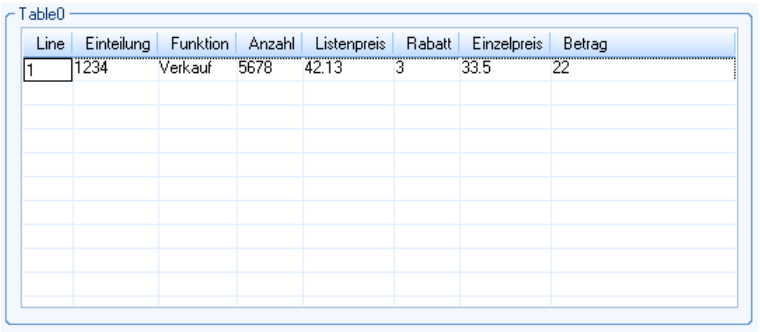
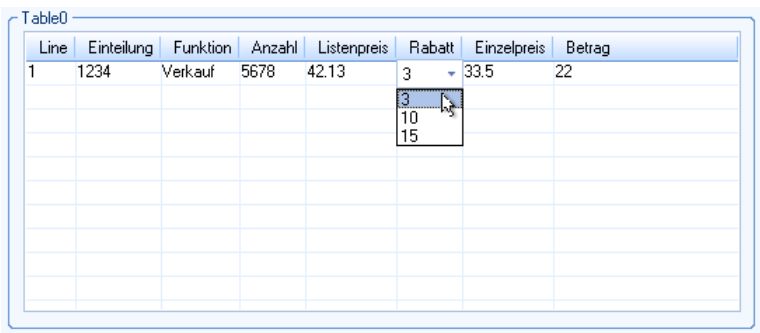


Tabelle der Beispieldatei mit einem Auswahlkatalog:



Beispiel für eine Änderung (Schreiben der Werte der ersten Zeile der Tabelle):

```
a = "{1.Wert" & chr(17) & "2.Wert" & chr(17) & "3.Wert" & chr(17) _
& "4.Wert" & chr(17) & "5.Wert" & chr(17) & "6.Wert" & chr(17) _
& "7.Wert}"
oxhelp.writeprofstring "object75list1", "zeile0", a, osfile
```

Die Änderungen werden nur geschrieben, falls Sie 'resultcode=1' in die Übergabedatei schreiben.

Das ActiveX-Control OXACTIVE.DLL

Über das ActiveX-Control `oxactive.dll` stehen Ihnen Methoden und Objekte zur Verfügung, mit denen Sie Daten aus der Übergabedatei und aus Konfigurationsdateien leicht auslesen und ändern können.

Das Control wird bei der Installation von enaio® automatisch installiert und registriert.

Das Control können Sie direkt über das VBScript-AddOn im VBScript-Editor benutzen. Aus anderen Umgebungen können Sie das ActiveX-Control `oxactive.dll` folgendermaßen einbinden:

`CoxHelp` Schnittstelle zur Unterstützung der VBScript-Programmierung

Die Objekterstellung in VBScript:

```
Dim x
Set x = CreateObject('oxactive.CoxHelp')
```

Methoden

GetProfString()

```
HRESULT GetProfString(BSTR bstrSec, BSTR bstrKey, BSTR bstrDefault,
VARIANT* pvarReturn, BSTR bstrFile)
```

Diese Funktion stellt die Funktionalität der Windows-API Funktion `GetPrivateProfileString` zur Verfügung.

Input:	`BSTR bstrSec`	der Abschnittsname in der Übergabe-Datei
	`BSTR bstrKey`	der Schlüsselname in diesem Abschnitt
	`BSTR bstrDefault`	Default-Rückgabewert, wenn kein Wert ermittelt werden kann
	`BSTR bstrFile`	Dateiname der Übergabe-Datei
Output:	`VARIANT* pVarReturn`	ermittelter Schlüsselwert als String

Beispiel:

```
oxhelp.GetProfString "PAGE00","#OSEXP#",-1,rvalue,osfile
```

Falls der Wert ermittelt werden kann, wird der Wert des Schlüssels '#OSEXP#' des Abschnitts 'PAGE00' in die Variable 'rvalue' geschrieben, andernfalls '-1'.

WriteProfString()

```
HRESULT WriteProfString(BSTR bstrSection, BSTR bstrKey, BSTR
bstrValue, BSTR bstrFile)
```

Diese Funktion stellt Ihnen die Funktionalität der Windows-API Funktion `WritePrivateProfileString` zur Verfügung.

Input:	`BSTR bstrSection`	der Abschnittsname in der Übergabe-Datei
	`BSTR bstrKey`	der Schlüsselname in der Übergabe-Datei
	`BSTR bstrValue`	Der Wert des Schlüssels
	`BSTR bstrFile`	Pfad und Dateiname der Übergabe-Datei
Output:	-	

Bevor die Methode `oxhelp.WriteProfString()` verwendet werden kann, rufen Sie die Methode `WriteToFile()` auf.

ExtractString()

```
HRESULT ExtractString(BSTR strVal, VARIANT* pFieldName, VARIANT*
pFieldValue)
```

Mit dieser Funktion teilen Sie einen String in zwei Hälften. Das Trennzeichen muss entweder das Zeichen '\021' oder ASCII-Code '17' sein.

Input:	`BSTR strVal`	String, der in zwei Hälften geteilt werden soll
Output:	`VARIANT* pFieldName`	linke Hälfte des Strings bis zum Trennzeichen
	`VARIANT* pFieldValue`	rechte Hälfte des Strings ab dem Trennzeichen

WinExec()

```
HRESULT WinExec(BSTR strFile, BSTR strParams)
```

Über diese Funktion können Sie Anwendungen starten und diesen Parameter übergeben.

Input:	BSTR strFile	Dateiname der Anwendung
	BSTR strParams	Kommandozeilenparameter
Output:	-	

Objekte

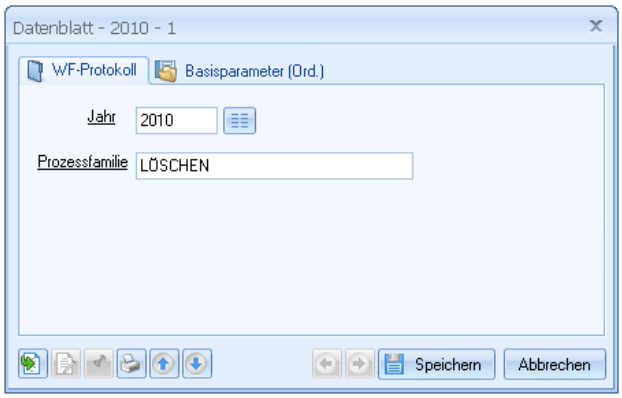
Folgende Objekte stellt das ActiveX-Control `oxactive.dll` zur Verfügung:

- ASFile-Objekt: bietet Zugriff auf die AddOn- bzw. Event-Übergabedatei
- RequestPages-Objekt: umfasst alle Register einer Verschlagwortungsmaske (Kollektion von RequestPage-Objekten)
- RequestPage-Objekt: ein einzelnes Datenblatt
- ActivePage-Objekt: hierbei handelt es sich um ein spezielles RequestPage-Objekt, nämlich einen Verweis auf das aktive Datenblatt

- ASFields-Objekt: umfasst alle Felder einer Verschlagwortungsmaske (Kollektion aller ASField-Objekte)
- ASField-Objekt: ein einzelnes Feld einer Maske

Über diese Objekte lesen Sie Daten der Übergabedatei und verändern diese.

Beispiel:



Sie können den Wert des Felds 'Jahr' über eine der beiden folgenden Varianten auslesen:

```
ActivePage.ASFields.Item("Jahr").Value
```

```
RequestPages.Item("WF-Protokoll").ASFields.Item("Jahr").Value
```

Bei a) wird das Feld über das ActivePage-Objekt ausgelesen, bei b) über die RequestPages-Kollektion.

ASFile-Objekt

Das ASFile-Objekt bietet Ihnen Zugriff auf die AddOn- bzw. Event-Übergabedatei. Diese Übergabedatei enthält Informationen zur Verschlagwortungsmaske.

Eigenschaften:

L – nur Lesezugriff

S – nur Schreibzugriff

L/S – Lese- und Schreibzugriff

`ActivePage`	Liefert ein RequestPage-Objekt des gerade aktiven (oben liegenden) Datenblattes.	L
`AddonFields`	Nur für AddOns. Liefert eine Kollektion von ASField-Objekten der Felder, die mit dem AddOn verbunden sind. Das erste Element ist genau das Feld, dem die AddOn-Schaltfläche zugeordnet ist. Über die anderen Elemente dieser Kollektion erhalten Sie Zugriff auf alle Felder, die mit dem AddOn verbunden sind.	L
`ErrorMessage`	Für die Events 'BeforeValidate' und 'OnItemClick' können Sie hier in Verbindung mit der Eigenschaft	L/S

	'ASField.IsErrorField' einen Fehlertext angeben, der dann in einem Info-Fenster angezeigt wird. Voraussetzung ist, dass der ResultCode ungleich 0 ist (siehe Beispiel 4).	
`EventAction`	Nur für Events. Mögliche Werte sind: NEW – es wird ein neues Objekt angelegt. UPDATE – ein bestehendes Objekt wird bearbeitet. REQUEST – es wird eine Suche durchgeführt READONLY – ein bestehendes Objekt ist schreibgeschützt geöffnet.	L
`EventCode`	Gibt den numerischen EventCode zurück (siehe 'Clientseitige Events').	L
`Filename`	Dateiname der zu verwendenden Datei einschließlich vollständigem Pfad, normalerweise der Dateiname der jeweiligen Übergabedatei.	L/S
`FOLDERID`	Liefert die ID des Ordners, in dem das Objekt liegt, zurück.	L
`FOLDERTYPE`	Liefert den numerischen Ordnertyp zurück.	L
`Handle`	Ziehpunkt der Verschlagwortungsmaske.	L
`IsErrorField`	Gibt an, ob dieses Feld als fehlerhaft auf der Verschlagwortungsmaske markiert wird.	L/S
`REGISTERID`	Liefert die ID des Registers, in dem das Objekt liegt, zurück.	L
`REGISTERTYPE`	Liefert den numerischen Registertyp zurück.	L
`RequestPages`	Kollektion von RequestPage-Objekten aller Datenblätter der aktuellen Verschlagwortungsmaske.	L
`ResultCode`	Nur für Events. Wird der Wert auf '1' gesetzt, werden alle Änderungen übernommen. Standardwert: '0'	L/S
`TARGETMAINTYPE`	Liest bzw. schreibt den Haupttyp des Dokuments.	L/S

Methoden:

`WriteToFile()`	Schreibt alle Objekte und Eigenschaften in die Übergabedatei zurück. Der Dateiname wird der Eigenschaft `Filename` entnommen. Rückgabewerte: '10' – Datei existiert nicht '11' – keinen Dateinamen angegeben

RequestPages-Objekt

Über dieses Objekt erhalten Sie Zugriff auf eine Kollektion von RequestPage-Objekten.

Eigenschaften:

L – nur Lesezugriff

S – nur Schreibzugriff

L/S – Lese- und Schreibzugriff

`Count`	Anzahl der Elemente dieser Kollektion	L

Methoden:

`Item(VARIANT Item)`	Liefert ein Element aus dieser Kollektion. Parameter: `Item` – hier können Sie entweder den Index oder den Namen des gewünschten Elements angeben.

RequestPage-Objekt

Dieses Objekt erlaubt Ihnen Zugriff auf ein einzelnes Datenblatt der Verschlagwortungsmaske.

Eigenschaften:

L – nur Lesezugriff

S – nur Schreibzugriff

L/S – Lese- und Schreibzugriff

`Active`	'1' wenn diese Seite die oben liegt, sonst '0'	L
`ASFields`	ASField-Kollektion aller Felder dieser Seite	L
`FileCount`	Anzahl der Dateien zu einem Dokument	L
`ID`	ID des Objekts	L
`Name`	Name des Datenblattes	L
`ObjectType`	Objekttyp dieser Seite	L

ASFields-Objekt

Dieses Objekt stellt eine Kollektion von ASField-Objekten dar.

Eigenschaften:

L – nur Lesezugriff

S – nur Schreibzugriff

L/S – Lese- und Schreibzugriff

`Count`	Anzahl der Elemente dieser Kollektion	L

Methoden:

`Item(VARIANT Item)`	Liefert ein Element aus dieser Kollektion Parameter: `Item` – hier können Sie entweder den Index oder den Namen des gewünschten Elements angeben.

ASField-Objekt

Über dieses Objekt erhalten Sie Zugriff auf ein einzelnes Feld in einer Suchmaske.

Eigenschaften:

L – nur Lesezugriff

S – nur Schreibzugriff

L/S – Lese- und Schreibzugriff

`CtrlPages`	Kollektion aller Register eines Pagecontrols.	L
`DBName`	Name des Felds in der Datenbank	L
`Enabled`	Auf FALSE gesetzt, wird das entsprechende Feld in der Maske schreibgeschützt.	L/S
`GUID`	Liefert die GUID des entsprechenden Felds in der Maske zurück.	L
`InternalName`	Der interne Name des entsprechenden Felds in der Maske.	L
`IsErrorField`	Gibt an, ob dieses Feld der Maske als fehlerhaft (gelb) auf der Verschlagwortungsmaske markiert werden soll ('TRUE') oder nicht ('FALSE'). (siehe Beispiel 4)	L/S
`Length`	Max. Länge des Felds	L
`Name`	Name des Felds	L
`Type`	Datentyp des Felds (s. u.)	L
`Value`	Wert des Felds	L/S
`Visible`	Auf FALSE gesetzt, wird das entsprechende Feld in der Maske ausgeblendet.	L/S

Mögliche Datentypen sind:

Typ	Beschreibung	Datenbank
#	Realzahlen	DECIMAL
0	Kontrollkästchen	SHORT
1	Optionsschaltfläche	SHORT
9	Zahlen	INTEGER
A	alle Zeichen ohne Ziffern	CHAR
C	Pagecontrol	CHAR
D	Datum	DATE

G	Großbuchstaben	CHAR
I	Volltextindex	INTEGER
L	alle Zeichen	CHAR
M	alle Zeichen	CHAR
P	Patientenart	CHAR
Q	ja/nein	CHAR
S	männlich/weiblich	CHAR
T	links/rechts	CHAR
W	Tabellenelement	CHAR
X	alle Zeichen	CHAR
Z	nur Ziffern	CHAR

Anwendungsbeispiele in VBScript

Werden die oben beschriebenen Objekte in enaio® client durch einen Event oder vom VBScript-AddOn verwendet, brauchen Sie das ASFile-Objekt und das CoxHelp-Objekt nicht explizit erstellen. Auch die Eigenschaft 'Filename' des ASFile-Objekts wird automatisch gesetzt.

Beispiel 1

Ausgabe der Namen aller Datenblätter:

```
for b = 0 to RequestPages.Count-1
  MsgBox(RequestPages.Item(b).Name)
Next
```

Beispiel 2

Ausgabe aller Feldnamen des gerade aktiven Datenblattes, sowie Änderung eines Feldinhaltes für einen Event:

```
for b = 0 to ActivePage.ASFields.Count-1
  MsgBox(ActivePage.ASFields.Item(b).Name)
Next
ActivePage.ASFields.Item(1).Value = 'neuer Wert'
ResultCode = 1
WriteToFile()
```

Beispiel 3

Ausgabe des Namens des AddOn-Felds, sowie dessen Inhalt im Datenblatt und anschließende Änderung des Inhalts des AddOn-Felds:

```
MsgBox (AddonFields.Item(0).Name + ' = ' +
AddonFields.Item(0).Value)
AddonFields.Item(0).Value = 'neuer Wert'
WriteToFile()
```

Beispiel 4

Farbliches Markieren eines Datenblatt-Felds auf einem Pagecontrol mit Aktivierung der entsprechenden Seite und Ausgabe einer Fehlermeldung als Tooltip für das Event 'OnClickItem' oder 'BeforeValidate':

```
ActivePage.ASFields("PageCtrl").CtrlPages(2).Fields.Item(2).IsErrorF
ield = true
ErrorMessage = "Fehlerhafte Eingabe, bitte korrigieren."
ResultCode = -1
WriteToFile()
```

Beispiel 5

Unterscheiden, ob eine Abfrage im Expertenmodus durchgeführt wurde oder nicht durch Auslesen des Schlüssels #OSEXP# der Übergabedatei:

```
oxhelp.GetProfString "PAGE00","#OSEXP#",-1,rvalue,osfile
if ( rvalue = "1") then
  'Code für Expertenmodus
else
  'Code für nicht-Expertenmodus
end if
```

Serverseitige Events

Server-Events sind Skripte, die einem Serverjob zugeordnet sind. Vor oder nach der Ausführung eines Serverjobs kann das zugeordnete VB-Skript ausgeführt werden.

Zwei Typen von Server-Events können eingebunden werden:

- KernelDrivenEvents (KDE)

  KernelDrivenEvents können vor oder nach dem Ausführen des Jobs aufgerufen werden. Vor dem Ausführen können durch das Skript alle Eingabeparameter geändert werden, nach dem Ausführen alle Rückgabeparameter.

- JobDrivenEvents (JDE)

  JDEs bieten im Gegensatz zu KDEs den Vorteil, dass die im Job implementierte Fachlogik größtenteils vom Event genutzt werden kann und nicht erneut implementiert werden muss.

  Das erste implementierte JDE, das mit enaio® ausgeliefert wurde, ist im DoArchive-Job (Archivierung von Objekttypen). Hier sind KDE in objektbezogenen Szenarien schlecht nutzbar, da der Job in einer internen Schleife über alle zu archivierenden Dokumente eines Objekttyps iteriert. Eine Ausführung eines KDE müsste diese Logik (übergeben würde nur der Parameter 'Objekttyp') und diverse Plausibilisierungschecks selbst abbilden.

  Der JDE im DoArchive-Job kann verwendet werden, um beispielsweise zusätzlich zum bisher im WORK-Bereich gehaltenen Dokument, eine Rendition zu archivieren.

  Dieses ist mit der Motivation eines rechtsicheren, revisionssicheren Langzeitarchivs immer dann angezeigt, wenn– unter anderem bei der Nutzung von DMS-Funktionalitäten – dazu weniger geeignete Formate (.doc, .xls) verwendet werden, die im Archiv als PDF gespeichert werden sollen.

Server-Events werden, wie Objekt- und Applikations-Events, über den Bereich 'Objektsuche' des enaio® client erstellt.

Um KDEs einzubinden, markieren Sie im Bereich 'Objektsuche' **allgemeine Events** und wählen **Event hinzufügen** aus dem Kontextmenü. Aus der Job-Liste **KernelBeforeJob** wählen Sie den entsprechenden Job, dessen Eingabeparameter über ein Skript bearbeitet werden, aus der Job-Liste **KernelAfterJob** wählen Sie den Job, dessen Rückgabeparameter bearbeitet werden. Das Skript erstellen oder importieren Sie über das Editor-Fenster.

Um JDEs einzubinden, markieren Sie im Bereich 'Objektsuche' einen Objekttyp und wählen **Event hinzufügen** aus dem Kontextmenü. Aus der Job-Liste im Bereich **Server-Events** wählen Sie dann den gewünschten Job.

Die Jobs in der Job-Liste sind nach den folgenden Namespaces gruppiert:

Namespace	Beschreibung
abn	Funktionen zur Einrichtung und Steuerung von Abonnements (Benachrichtigungen bei Änderungen am Dokumentenbestand)
adm	administrative Jobs (Funktionen zur Verwaltung von Systemdateien und einige Konfigurationsaufgaben am Server)
ado	Datenbankzugriff über ADO
cnv	Konvertierungen von Bilddateien
dms	Jobs zum Anfragen und Bearbeiten von Indexdaten, DMS Objekten, Relationen und Mappen unter Berücksichtigung des Sicherheitssystems
dtr	Serverseitiger Aufruf des Datenübernahmeservers
krn	Kernel-Jobs (Funktionen in den Bereichen Administration, Lizenzierung, Sitzungsmanagement, Engine-Verwaltung und interne Steuerungen)
mng	Jobs für die Verwaltung von Gruppen und Benutzern des enaio®
std	DMS-Jobs und Archivierung (Work-, Cache-, Datei- und Archiv-Verwaltung)
tst	Test-Exekutor
vtx	Bearbeitung von Volltextanfragen enaio® client
wfm	Bearbeitung und Verwaltung von Workflowprozessen und Modellen

Die Dokumentation der Serverjobs erhalten Sie auf Anfrage.

Folgende JobDrivenEvents stehen darüber hinaus zur Verfügung:

- OnSessionLogin

  Über dieses Event ist eine Protokollierung von Anmeldeversuchen möglich.

- OnObjectHistoryEntry

  Über dieses Event ist eine pauschale Überwachung aller relevanten Änderungen an Objekten möglich.

Server-Events für die Archivierung

Folgende Events stehen für Server-Events im Rahmen der revisionssicheren Archivierung zur Verfügung:

- BeforeOpenMedia

  Bevor ein Medium innerhalb eines Archivlaufes beschrieben wird.

  Medium-Name und Medium-Nummer werden zurückgegeben.

- BeforeStartArchiveBatch

  Bevor ein Archivlauf beginnt und nachdem die Anzahl der zu archivierenden Dokumente berechnet wurde.

  Zurückgegeben werden Objekttypnamen, Objekttypnummern, Anzahl der zu archivierenden Dokumente.

- OnCloseMedia

  Nachdem die Archivierung auf einem Medium innerhalb eines Archivlaufs abgeschlossen ist.

  Medium-Name und Medium-Nummer werden zurückgegeben.

- OnFinishArchiveBatch

  Nachdem ein Archivlauf beendet wurde.

  Zurückgegeben werden Objekttypnamen, Objekttypnummern, Anzahl der erfolgreich/fehlerhaft archivierten Dokumente, alle Statistik-Daten aus der Report-Datei.

- OnFinishMedia

  Nachdem ein Medium endgültig abgeschlossen wurde. D. h. im Fehlerfalle, wenn es voll ist oder an allen Stellen, wenn es als gesperrt markiert wird.

  Medium-Name und Medium-Nummer werden zurückgegeben.

Als Rückgaben sind, kontextabhängig 'BreakJob' (Jobabbruch) und 'BreakMedium' (Medium nicht verwenden) möglich.

Serverseitige Skripte

Skripte, die auf dem Server laufen, müssen unter Umständen auf Kernel-Objekte des Servers zugreifen können. Zu diesem Zweck wird vom Server im Skript das Objekt 'Running Context' (RC) zur Verfügung gestellt. Soweit es von den JobDrivenEvents nicht anders definiert ist, lautet die Variablen-Instanz 'RC'. Darüber hinaus können von den verschiedenen Jobs in JobDrivenEvents auch proprietäre Objekte zur Verfügung gestellt werden. Diese werden dann in der Dokumentation der entsprechenden Jobs gesondert vermerkt. Auch die Art der Bereitstellung dieser Objekte ist jobspezifisch.

Entwicklung von Skripten

Alle Kernel-Objekte sind dem Running Context zugeordnet und sind niemals direkt im globalen Namensraum des Skripts sichtbar, im Gegensatz zu Objekten, die der Job selbst hinzufügt. Diese können auch im globalen Namensraum sichtbar sein. Auf jeden Fall sind sie zusätzlich auch dem Running Context zugeordnet. Somit unterscheiden sich die Kernel-Objekte von jobspezifischen Objekten, da die Kernel-Objekte nur über den Running Context sichtbar sind und niemals direkt, wobei die jobspezifischen Objekte immer über Running Context und möglicherweise auch direkt als Variablen sichtbar sein können. Auch auf diese jobspezifischen Objekte kann über den Running Context zugegriffen werden.

Es gibt zwei Möglichkeiten, auf die Objekte über Running Context zuzugreifen, entweder über Eigenschaft 'Item', z. B. 'RC.Item("SessionData")', oder mit dem Punkt vom RC getrennt, z. B. 'RC.SessionData'.

Die erste Möglichkeit funktioniert sowohl mit Namen, als auch mit Nummern. Die Nummer des Objektes kann als Parameter der Eigenschaft 'Item' übergeben werden: RC.Item(2).

Die gesamte Anzahl der Objekte im RC kann mit dem Eigenschaft 'RC.Count' ermittelt werden. Die Kernel-Objekte stehen in der Liste immer hinter allen jobspezifischen Objekten, d. h. die Nummerierung ist nicht bei jedem Aufruf die gleiche.

RC

Der Running Context hat außer allen Objekten noch die Eigenschaft 'GUIAvailable', mit der man aus dem Skript heraus ermitteln kann, ob die GUI-Aufrufe im Skript erlaubt sind, und die Methode 'NewJobsParams', mit der man eine neue Instanz der Parameterliste erstellen kann, die dann für die Jobsaufrufe verwendet wird.

Die dritte Methode des RC ist 'IncludeFile(BSTR FileName)' und kann dafür benutzt werden, ein großes Skript auf mehrere aufzuteilen.

Eine weitere Eigenschaft des RC ist 'UseNewDB'. Sie wird verwendet, um zu steuern, ob Jobs in einer Transaktion aufgerufen werden oder einen eigenen Transaktionskontext bekommen.

Objekte

Es gibt folgende Kernel-Objekte des ersten Levels:

- SessionData,
- ServerData,
- General,
- Logger,
- Jobs,
- Actions,
- Registry.

Im Folgenden ist die Beschreibung der einzelnen Objekte aufgeführt.

SessionData

Dieses Objekt liefert Daten über das Konto des Benutzers, in dessen Kontext das Skript ausgeführt wird. Momentan stehen folgende Eigenschaften zur Verfügung:

Name	Beschreibung
SessGUID	GUID der Client-Sitzung
UserName	Name des Benutzers
StatName	Name des Rechners, an dem enaio® client läuft

Name	Beschreibung
InstName	Name des Programms, aus dem das Login erfolgte
UserGUID	GUID des Benutzers
StatGUID	GUID der Station des enaio® client Das ist keine MAC-Adresse, sondern die GUID des Datensatzes aus der DB.
UserID	ID des Benutzers
Supervisor	Das Supervisor-Flag aus der Benutzer-Tabelle für den Benutzer
LangID	Sprache-ID aus der Benutzer-Tabelle für den Benutzer

ServerData

Dieses Objekt liefert unterschiedliche Konfigurationsdaten aus der Laufzeitumgebung des Servers. Momentan stehen folgende Eigenschaften zur Verfügung:

Name	Beschreibung
DataDir	Pfad zum Datenbereich der Servergruppe
RootDir	Pfad, über den der Server gestartet wurde
TempDir	Pfad zum ostemp-Verzeichnis des Servers
ConfDir	Pfad zum etc-Verzeichnis der Servergruppe
UserDir	Pfad zum user-Verzeichnis der Servergruppe
Service	Name des Dienstes
LogDir	Pfad zum log-Verzeichnis, in das der Server protokolliert
Connect	Computername und TCP-Port, durch '#' getrennt
ServerID	ID des Servers
GroupID	ID der Servergruppe

General

Dieses Objekt liefert einige allgemeine Daten des Jobs. Momentan stehen folgende Eigenschaften zur Verfügung:

Name	Beschreibung
JobNumber	Nummer des Jobs, fortlaufend seit dem Serverstart gezählt
ThreadID	ID des Threads, in dem der Jobs gerade ausgeführt wird

Logger

Dieses Objekt gibt die Möglichkeit im Kernel-Kontext zu protokollieren. Das Skript kann auch ein eigenes Protokoll führen, indem er die `oxrpt.dll` per COM direkt anspricht. Dabei wird eine eigene Konfiguration genommen. Alternativ kann das Skript über das Objekt Logger protokollieren. Das Logger-Objekt stellt folgende Methoden zur Verfügung:

Name	Beschreibung
Flow	Flow-Protokoll auf Level 5
FlowEx	Level und die Stelle im Skript können festgelegt werden
Info	Flow-Protokoll mit Facility 'informational' auf Level 3
InfoEx	Wie Info, die Stelle im Skript kann festgelegt werden
Warning	Flow-Protokoll mit Facility 'warning' auf Level 3
WarningEx	Wie Warning, die Stelle im Skript kann festgelegt werden
Error	Error-Protokoll mit Facility 'error' auf Level 0
errorEx	Wie Error, die Stelle im Skript kann festgelegt werden

Der Aufruf eines Protokolleintrags wird im Protokoll mit dem Text 'SCRIPT: […]' erscheinen, wobei in der Klammer der übergebene Text protokolliert wird. Je nach Methodenaufruf erscheint der Eintrag als 'Flow', 'Info', 'Warning' oder 'Error'.

Alle Methoden haben einen Parameter 'Text', der den eigentlichen Protokolleintrag darstellt, und je nach Methode weitere Parameter:

```
Flow(BSTR Text);
Warning(BSTR Text);
Info(BSTR Text);
Error(BSTR Text);
FlowEx(BSTR Text, UINT Level, BSTR Function, UINT Line);
WarningEx(BSTR Text, BSTR Function, UINT Line);
InfoEx(BSTR Text, BSTR Function, UINT Line);
ErrorEx(BSTR Text, BSTR Function, UINT Line);
```

Die Protokollierung über Methode 'Flow' wird auf dem Level 5 durchgeführt, die Methoden 'Info' und 'Warning' benutzen Level 3, und die Methode 'Error' benutzt Level 0.

Jobs

Dieses Objekt gibt die Möglichkeit unterschiedliche Server-Jobs ausführen zu lassen. Die Jobs werden synchron ausgeführt, d. h. die Aufrufmethode kehrt genau dann zurück, wenn der Job beendet ist, erfolgreich oder nicht. Der Aufruf eines Jobs sieht beispielsweise folgendermaßen aus:

```
1    Dim PrmIn
2    Dim PrmOut

3    Set PrmIn = RC.NewJobsParams
4    Set PrmOut = RC.NewJobsParams

5    PrmIn.Clear()
6    PrmIn.Value("Flags") = 0
7    res = RC.Jobs.krn.GetNextIndex(PrmIn,PrmOut)
8    if res = 0 then
9    MsgBox("krn.GetNextIndex succeeded: " & NextIndex)
     'Zur Ermittlung des Parameters siehe nächstes Kapitel.
10   else
11   MsgBox("krn.GetNextIndex failed: " & res)
12   end if
13   PrmOut.Clear()
```

In Zeilen 1 bis 4 erstellt man zwei Objekte, die als Eingabe- und Ausgabeparametern für den Job verwendet werden. Diese Objekte kann man nicht

nur für einen Job, sondern für mehrere Jobs verwenden. Man kann zwar die Elemente hinzufügen und abfragen, aber nicht einzeln löschen. Deswegen wird die Methode 'Clear' eingeführt, mit der die Liste leer gemacht wird (Zeilen 5 und 13). Dann kann die Liste von Eingabeparametern gefüllt werden. Die Eigenschaft, die den jeweiligen Parameter darstellt, heißt 'Value' und hat den Namen des Parameters als Argument. Der Typ eines Parameters wird analysiert und dabei wird ein XMLRPC-Parameter erstellt, der den entsprechenden Typ hat. 'BSTR' wird nach 'XMLRPC-String' konvertiert. Verschiedene 'Integer-Werte' werden 'XMLRPC-Integer', und 'BOOL' wird 'XMLRPC-Boolean'. Andere 'XMLRPC-Typen' werden in der vorliegenden Version nicht unterstützt.

Danach erfolgt der Aufruf selbst, und zwar in Form:

```
RC.Jobs.namespace.jobname(in-list,out-list)
```

Hier ist 'namespace' die Bezeichnung eines Exekutors bzw. Namespaces (wie immer drei Buchstaben) und darauf folgt der Job-Name. Der Jobaufruf liefert einen Integer-Wert zurück, der normalerweise auf '0' gesetzt wird, falls der Job erfolgreich war. Dieser Wert stellt 'dwResult' da, was die eigentliche Job-Funktion zurückgibt. Sollten Probleme auftreten, die dazu führen, dass der Job überhaupt nicht aufrufbar ist (z. B. eine Exception oder der Job bzw. Namespace sind nicht bekannt), wird kein Rückgabewert zurückgegeben, sondern eine COM-Exception generiert. Wenn also der Aufruf 'Job.krn.MyJob' zurückkehrt, ohne eine COM-Exception auszulösen, kann davon ausgegangen werden, dass MyJob aufgerufen wurde und wenn der Aufruf Job.krn.MyJob z. B. den Wert '-1' zurückgibt, so kommt dieser Wert aus der Jobfunktion, und nicht aus dem Kernel heraus.

Um dem Job Dateien zu übergeben, fügt man einen weiteren Parameter hinzu. Dieser Parameter heißt JobFiles$ und ist eine Zeichenkette. Sie besteht aus Dateinamen, die durch Semikolon getrennt angegeben werden.

Nach der Rückkehr kann man die Ausgabeparameter auswerten, sie befinden sich in unserem Beispiel in der Output-Parameter Liste 'PrmOut'. Selbstverständlich muss man die Namen der Parameter wissen. Die Ausgabedateien werden erhalten, indem der Job ausgewertet und gesplittet wird.

Dabei ist zu beachten, dass jedem Dateinamen ein Fragezeichen vorangestellt sein kann. Dies ist der Fall, wenn vom Job temporäre Dateien im ostemp-Verzeichnis angelegt werden, z. B. eine verschlüsselte Datei aus dem WORK-Verzeichnis wird in das ostemp Verzeichnis entschlüsselt.

Ist dem Dateinamen kein Fragezeichen vorangestellt, handelt es sich um keine temporäre Datei, und wurde somit z. B. direkt im CACHE oder WORK-Verzeichnis abgelegt.

Wird beispielsweise eine nicht vorhandene Variable angefragt, liefert der Performer den Parameter auch zurück, nur ist die Variable dabei nicht initialisiert, sie hat also den Typ 'VT_EMPTY'. Im Skript kann es mit der Funktion 'IsEmpty' geprüft werden. Die Namen der Jobs bzw. Namen und Typen der Jobparameter sind in Dokumentation der Server-API aufgeführt.

Sollen dem Job BASE64 Parameter übergeben werden, bzw. von dem Job BASE64 Parameter ausgelesen werden, so sind die Details dem folgenden Abschnitt

'Parameters' zu entnehmen. Ist beispielsweise die Variable Dom ein MSXML.DOMDocument-Objekt, in das ein XML geladen wurde, so kann man mit dem Befehl

PrmIn.Value("Buffer") = Dom

einen Input Parameter vom Typ BASE64 dem Job zuweisen. Umgekehrt kann man einen Ausgabeparameter folgendermaßen auslesen:

B64Result = PrmOut.Value("Result")

Dom.load(B64Result)

Wenn Sie einen Job im Skript am Server aufrufen und diesem den $$$Server-ID$$$-Parameter übergeben, dann wird dieser Job an den übergebenen Server weitergeleitet und dort ausgeführt. Aus Sicherheitsgründen gilt dieses Verfahren nur, wenn der Job per RC in einem Skript am Server aufgerufen wird.

Actions

Mit diesem Objekt sind einige Funktionsaufrufe möglich, die der Kernel selbst implementiert. Momentan stehen folgende Methoden zur Verfügung:

Name	Beschreibung
SendMail	Sendet eine E-Mail mit Eingabe von Ziel und Quelladressen, Thema, Text und die Liste von Dateien.
SendAdminMail	Sendet eine E-Mail an den Administrator, falls seine Adresse in der Registrierung konfiguriert ist.
StreamReset	Leert einen Istream.

Die Liste mit Dateinamen ist eine Zeichenkette, in der die Dateinamen mit Semikolon getrennt sind. Die Methoden haben folgende Parameter:

```
SendMail(BSTR To, BSTR From, BSTR Subject, BSTR Text, BSTR Files);
SendAdminMail(BSTR From, BSTR Subject, BSTR Text, BSTR Files);
StreamReset (VARIANT stream); // VARIANT ist vom Typ IStream oder
BYREF | VARIANT und dann IStream
```

Registry

Dieses Objekt bietet die Möglichkeit die Elemente in der Server-Registrierung zu lesen und zu schreiben. Es stehen folgende Eigenschaften zur Verfügung:

Name	Beschreibung
Element	Diese Eigenschaft kann gelesen und geschrieben werden, und hat einen Parameter von Typ 'BSTR', der den vollständigen Pfad zum Element darstellt. Der Wert des Elementes hat ebenfalls den Typ 'BSTR'.

Der Pfad zum Element fängt mit dem aktuellen Schema an. Beispielsweise wird auf das Element 'ComString', das unter dem aktuellen Schema liegt, so zugegriffen: RC.Registry("ComString").

Auf das Element 'Schema' das unter dem Schlüssel 'DataBase' liegt, wird mit 'RC.Registry("DataBase\Schema")' zugegriffen. Achten Sie dabei auf die Groß-/Kleinschreibung.

Parameters

Dem Skript können verschiedene Parameter übergeben werden. Genauso können diese nach der Ausführung des Skripts zurückgegeben werden.

```
Dim InputNames
InputNames = RC.InputParams.Names
MsgBox("Input params: " & InputNames)

Dim NameArray
NameArray = Split(InputNames,";",-1,1)
MsgBox("VarType((NameArray) = " & VarType(NameArray))

Dim sMsg, i, sName
Dim Param
for i = LBound(NameArray) to UBound(NameArray)
   sName = NameArray(i)
   if (len(sName) > 0) then
        Param = RC.InputParams.Value(sName)
        if (IsEmpty(Param)) then
         sMsg = sMsg & "Parameter " & sName & " not found"
       else
         sMsg = sMsg & sName
          sMsg = sMsg & ": "
          sMsg = sMsg & Param
       end if
          sMsg = sMsg & vbCrLf
       end if
  next
  MsgBox(sMsg)

  RC.OutputParams.Value("hallo") = "Welt"
```

Zwei Objekte 'RC.InputParams' und 'RC.OutputParams' ermöglichen die Arbeit mit Parametern. Die Eigenschaft 'Names' gibt die Namen von allen Parametern aus der Liste zurück, durch Semikolon getrennt. Die Eigenschaft 'Value' ermöglicht, den Parameter zu lesen oder zu schreiben.

Ein besonderer Fall ist die Ausführung von Skripten bei KernelDrivenEvents. Die KDEs werden vor und nach dem Job aufgerufen. Sie brauchen deshalb Zugriff auf die Job-Parameter. Das wird durch die Variablen 'InputParams' und 'OutputParams' gewährleistet. Im Before Skript eines KDEs werden alle Parameter dem Skript als Input Parameter zur Verfügung gestellt. Nach der Ausführung des Skripts wird diese Liste von Input Parametern vollständig die originale Parameterliste des ursprünglichen Jobs ersetzen, denn es können auch einzelne Parameter aus der Liste entfernt werden. Im After Skript eines KDEs werden alle Input Parameter des Jobs (bzw. des Before Skripts) zur Verfügung gestellt, sowie alle Ausgabeparameter des Jobs. Die Ausgabeparameter können nun manipuliert werden. Im Gegensatz zum Before Skript werden die Output-Parameter nicht komplett ersetzt, sondern lediglich die Werte der vorhandenen Parameter geändert.

Bei Parametern, die nicht vom Type 'BASE64' sind, können die Werte wie oben beschrieben abgefragt bzw. geändert werden. Bei BASE64-Parametern wird anders vorgegangen: Solche Parameter werden als 'IStreams' dargestellt. Das Skript kann

nun solche IStreams an diejenigen Schnittstellen weiterleiten, die diese auch verstehen, so z. B. an ein XMLDOMDocument, denn die Implementierung des 'msxml' erlaubt das Laden von XML aus IStreams. Anschließend kann ein mögliches DOM erneut in einen IStream gespeichern werden. Allerdings wird dabei der Inhalt von 'DOMDocument' anschließend dem IStream hinzugefügt, deswegen muss der IStream vorher geleert werden. Das kann mit 'RC.Actions.StreamReset(param)' erreicht werden.

Files

RunningContext hat Methoden, die es erlauben, Dateilisten dem Skript zu übergeben bzw. zurückzubekommen.

Mit 'RC.InputFiles' kann man im Skript die Dateiliste auslesen, die das Skript als Eingabeliste bekommt.

Mit 'RC.OutputFiles' kann man im Skript die Ausgabeliste von Dateien festlegen.

Beispiel: RC.OutputFiles = "c:\temp\1.txt;c:\temp\2.txt"

Die einzelnen Dateinamen müssen dabei mit Semikolon getrennt werden.

RunScript

Die beschriebene Funktionalität kann mit dem Job 'krn.RunScript' getestet werden. Dieser Job kann außerdem für administrative Zwecke benutzt werden, um beispielsweise periodisch bestimmte Skripte ausführen zu lassen.

Der Job 'RunScript' benötigt folgende Parameter:

Name	Typ	Beschreibung
Flags	Integer (2)	Wenn der Wert '1' ist, werden die dem Job übergeben Dateien nicht gelöscht.
Script	String (1)	Der Text des Skripts. Wenn dieser Parameter eine leere Zeichenkette ist, wird geprüft, ob mit dem Job mindestens eine Datei ankommt. Wenn nicht, so wird ein Fehler zurückgegeben. Wenn ja, so wird der Dateiinhalt als Skripttext verwendet. Nach dem Job werden alle angekommenen Dateien gelöscht, falls Flags den Wert '0' hat.
GUI	Boolean (3)	Wird weiter dem Performer als 'bGUIAvailable' übergeben. Mit dem Wert '0' wird die Möglichkeit abgeschaltet, eine Nachrichtenfenster anzuzeigen. Der Wert '1' kann diese Möglichkeit einschalten, es sei denn sie ist per Registrierung generell abgeschaltet worden.
CtxName	String (1)	Name des RunningContext, unter dem er im Skript sichtbar gemacht wird. Wenn dieser Parameter leer ist, wird 'RC' benutzt.

Main	String (1)	Name der Funktion, die im Skript aufgerufen werden soll, kann aber auch leer sein. Wenn der Parameter fehlt, wird 'Main' als Funktion angenommen.
Eval	Boolean (3)	Gibt an ob der Skripttext eine Expression darstellt.

Der Rückgabewert der serverseitig ausgeführten Skript-Funktion (mit leerem Job-Parameter 'Main' heißt die Funktion 'Main') wird nach Jobausführung als Rückgabeparameter '$ScriptResult$' (vom Datentyp String) zurückgeliefert.

Beispiel:

```
Set oServer = CreateObject( "OxSvrSpt.Server" )
set oSession = oServer.Login("<User>" , "<PWD>" , "<Server>",
"<Port>" )

sScript= _
"Function Main" & vbcrlf & _
"Main=now" & vbCrLf & _
"End Function"
MsgBox _
"Lokale Zeit: " & vbTab & now & vbCrLf & _
"~Zeit am Server: " & vbTab & _
RunScript(oSession, sScript), vbInformation, _
"Zeitdifferenz lokal/server"

Function RunScript(oSession, sScript)
     set oJob = oSession.NewJob("krn.RunScript")
     oJob.InputParameters.AddNewStringParameter "Script", sScript
     oJob.InputParameters.AddNewStringParameter "CtxName", ""
     oJob.InputParameters.AddNewIntegerParameter "Flags", 0
     oJob.InputParameters.AddNewBooleanParameter "GUI", 0
     oJob.InputParameters.AddNewBooleanParameter "Eval", 0
     oJob.Execute
     RunScript = oJob.OutputParameters("$ScriptResult$").Value
End Function
```

Als Ergebnis wird folgendes Nachrichtenfenster zurückgeliefert:

Um dem Skript eine Parameterliste übergeben zu können, kann man dem Job weitere optionale Parameter übergeben. Diese Parameter müssen sich von normalen Jobparametern unterscheiden und müssen daher Namen besitzen, die bei diesem Job nicht vorhanden sind. Außerdem müssen beim Aufruf die Namen und die Werte wie folgt getrennt werden:

Name	Typ	Beschreibung
SP_1_name	String (1)	Der Name des ersten Skript-Parameters
SP_1_value	Beliebig	Der Wert des ersten Skript-Parameters
...		
SP_xx_name	String (1)	Der Name des letzten Skript-Parameters

SP_xx_value	Beliebig	Der Wert des letzten Skript-Parameters

Diese Parameter müssen fortlaufend durchnummeriert werden.

Globale Skripte

Globale Skripte dienen dazu, Konstanten, Subprogramme und Funktionen zu hinterlegen. Sie können ein globales Client-Skript und ein globales Server-Skript einrichten.

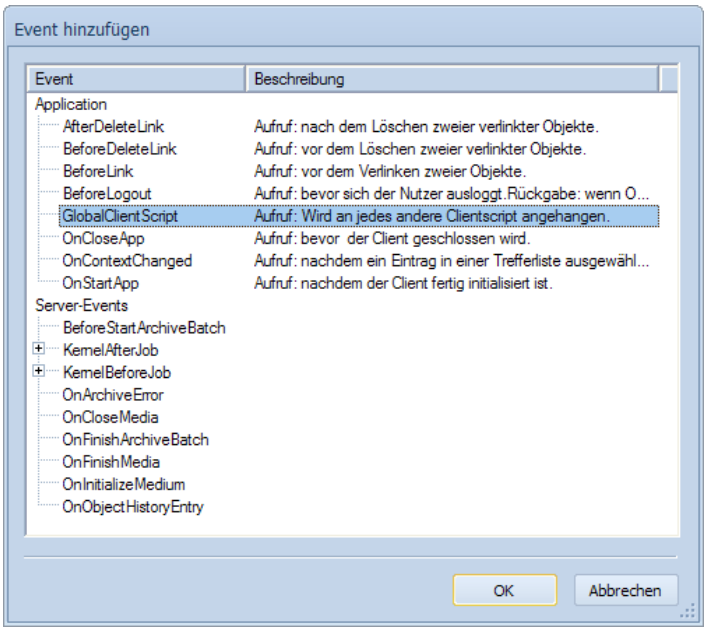
An die Skripte der Client-Events werden vor der Ausführung immer die Daten des globalen Client-Skripts angehängt, an die Skripte der Serverevents immer die Daten des globalen Server-Skripts.

Weiter kann für jeden Objekttyp ein globales Objekttyp-Skripte angelegt werden. Dort werden Daten hinterlegt, die nur für diesen Objekttyp verwendet werden. Globale Objekttyp-Skripte richten Sie über das Kontextmenü eines Objekttyps im Bereich 'Objektsuche' ein.

Durch diese Organisation kann die Pflege und Verwaltung von Skript-Code deutlich vereinfacht werden.

Das Hinzufügen von Funktionen über Execute bzw. ExecuteGlobal in Skripten ist technisch möglich, wird aber aus Gründen der Sicherheit und Performance in Produktivsystemen nicht empfohlen. Ebenso wird die Fehlersuche deutliche erschwert.
Für Skripte mit Execute bzw. ExecuteGlobal kann der Support ausgeschlossen werden.

Markieren Sie im Arbeitsbereich **Allgemeine Events** und wählen Sie über das Kontextmenü die Funktion **Event hinzufügen**.



Markieren Sie **GlobalClientScript** oder **GlobalServerScript** und bestätigen Sie mit **OK**.

Das Editor-Fenster wird geöffnet. Sie können das Skript editieren.

Enthält ein globales Skript selbst ausführbaren Skript-Code, wird dieser bei jedem entsprechenden Event ausgeführt. Wird Skript-Code ausgeführt und ist das Ergebnis 'falsch', dann wird ein noch ausstehender Job nicht ausgeführt.

Beispiel für Event-Skript-Code mit Bezug auf ein GlobalClientScript:

```
MsgBox TextForString
useGlobalScript

Sub useGlobalScript
  Dim ret
  call globalHello
  ret = Dummy("String aus aufrufender Fkt")
  MsgBox ret
End Sub
```

Beispiel für das entsprechende GlobalClientScript:

```
const TextString = "String aus global Script"
const TextForDummy = "Funktion erfolgreich ausgeführt Input="

Sub globalHello()
  MsgBox"Hallo aus dem globalen Script"
End Sub

Function Dummy (text)
  Dummy = TextForDummy + text
End Function
```

Steuerung des Info-Fensters

Neben dem Bereich 'Objektsuche' und der Suchleiste steht in enaio® client ein drittes Fenster zur Verfügung, das Info-Fenster. Dieses Fenster kann nicht durch den Benutzer gestaltet werden, sondern nur administrativ über Events.

Dem Info-Fenster wird eine URL übergeben oder ein HTML-String. Es verfügt über eine COM-Schnittstelle, die von jedem Event-Skript aus unter dem Namen 'InfoWindow' angesprochen werden kann.

Folgende Eigenschaften und Methoden sind in dieser Schnittstelle implementiert:

Eigenschaft	Bedeutung
ID	eindeutige ID des Fensters
Visible	gibt an oder legt fest, ob das Fenster sichtbar ist
Caption	Text in der Titelleiste des Fensters
URL	gibt die aktuelle URL an, die zurzeit angezeigt wird oder angezeigt werden soll
Closeable	gibt an oder legt fest, ob das Fenster durch den Benutzer geschlossen werden kann

EnableContextMenu	legt fest, ob das Kontextmenü des Internet Explorer angezeigt werden soll
HtmlDocument	gibt das aktuelle HTML-Dokument an

Methode	Bedeutung
ShowHtml(String html)	zeigt den übergebenen HTML-String im Fenster an
Refresh()	aktualisiert die Anzeige
GoBack()	entspricht der 'Zurück'-Schaltfläche im Internet Explorer
GoForward()	entspricht der 'Vorwärts'-Schaltfläche im Internet Explorer

Administration der Events

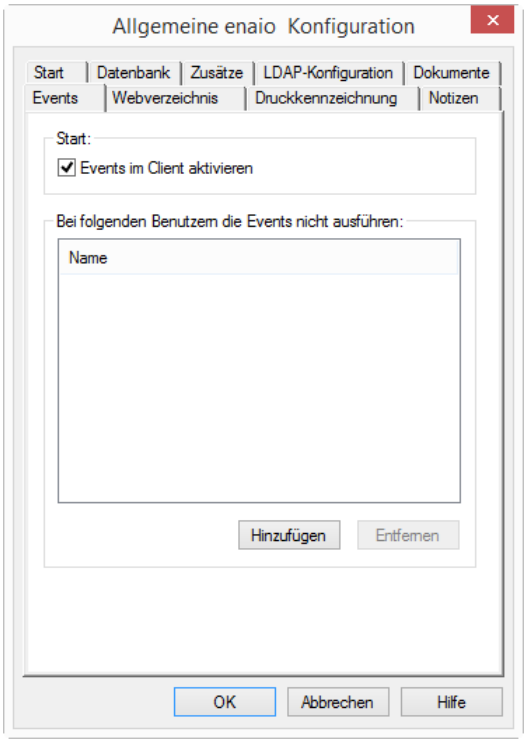
Sie können einstellen, ob und für welche Benutzer Events in enaio® client ausgeführt werden. In enaio® webclient werden Events immer für alle Benutzer ausgeführt.

Dazu öffnen Sie den Dialog **Allgemeine enaio Konfiguration** mit einem Klick auf **Gesamtsystem** im Menüband.

Wenn Sie auf der Registerkarte **Events** die Option **Events im Client aktivieren** wählen, werden Events für alle Benutzer ausgeführt, die nicht in die Benutzerliste eingetragen werden.

Wählen Sie die Option nicht, werden die Events nur für die Benutzer ausgeführt, die in die Benutzerliste eingetragen werden.

Die Liste erstellen Sie über die Schaltfläche **Hinzufügen**.

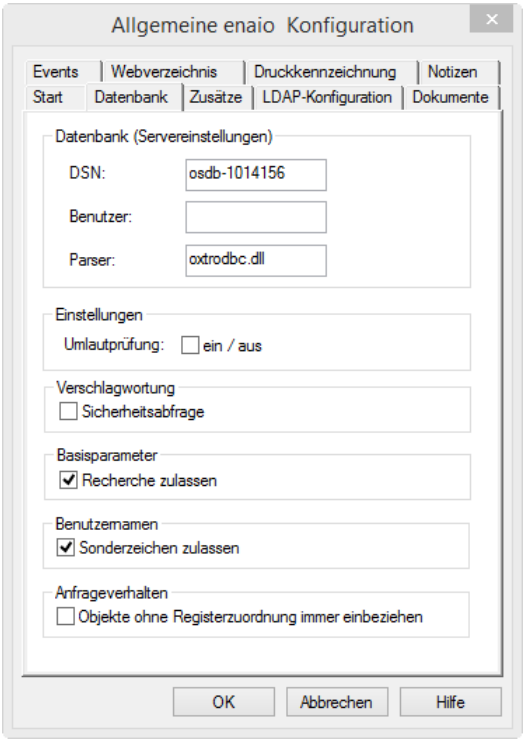


Falls Sie Events ändern, werden diese in enaio® client für andere Benutzer erst aktualisiert, wenn enaio® client neu gestartet wird oder Benutzer die Events über die Tastenkombination **Umschalt+Strg+F5** aktualisieren.

Über enaio® administrator kann dann das Anfrageverhalten für eingebundene Skripte konfiguriert werden.

Auf der Registerkarte **Gesamtsystem/Datenbank** markieren Sie das Kontrollkästchen **Objekte ohne Registerzuordnung immer einbeziehen**, wenn bei einer Suche über Register- und Dokumentendaten Dokumente ohne Registerzuordnung – Dokumente, die nicht in einem Register liegen – mit in die Trefferliste aufgenommen werden sollen.

Diese Einstellung wird nur ausgewertet, wenn das Verhalten nicht im Skript selbst festgelegt ist.



Für die Suche in enaio® client kann jeder Benutzer dieses Anfrageverhalten über seine benutzerspezifischen Einstellungen im Bereich 'Anfrageverhalten' festlegen.

enaio® editor-for-events

enaio® editor-for-events – Einführung

Mit enaio® editor-for-events ordnen Sie ein Skript einem DMS-Objekt und einem Event zu und speichern das Event in der Datenbank.

enaio® editor-for-events ist in enaio® client integriert. Verfügt ein Benutzer über die notwendigen Systemrollen und Lizenzen, werden die entsprechenden Funktionen im enaio® client freigeschaltet.

Zum Testen der Events kann in enaio® client der Debug-Modus (siehe 'Debugging'.) eingestellt werden.

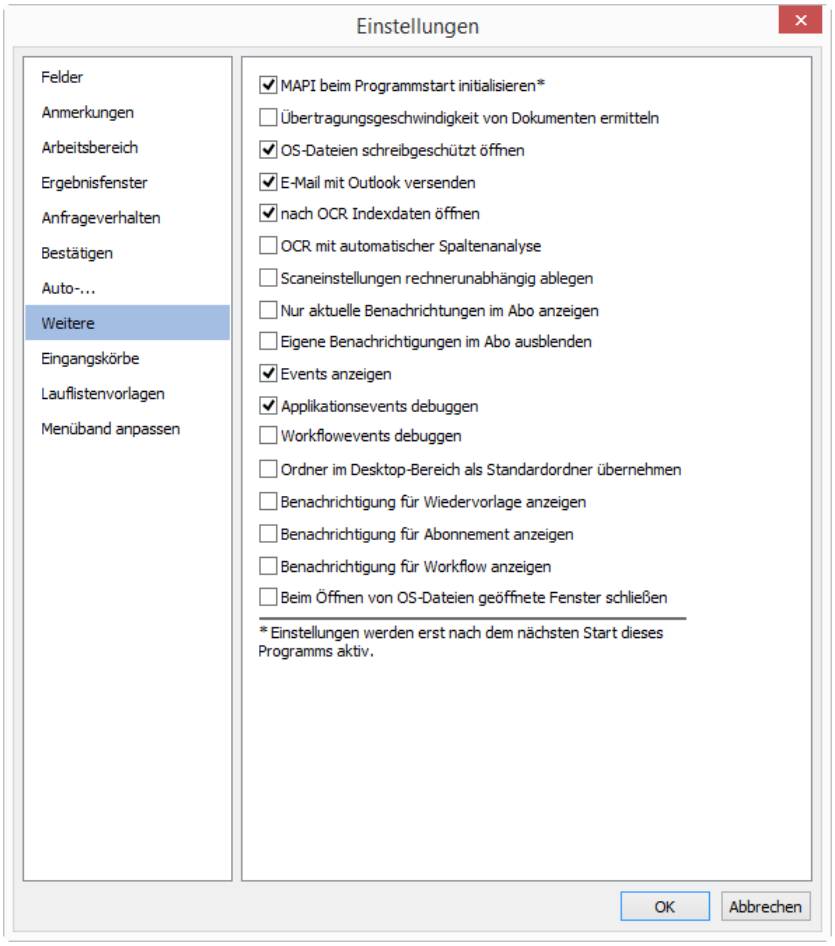
Mit den folgenden Schritten erstellen Sie Events:

- Sie wählen im Bereich 'Objektsuche' ein DMS-Objekt oder enaio® client aus,
- wählen ein Event,
- tragen den Script-Code ein,
- speichern das Event in der Datenbank.

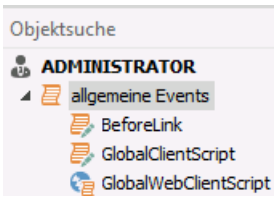
Das Event wird danach sofort für alle administrierten Benutzern (siehe 'Administration der Events') ausgeführt, die enaio® client neu starten oder in enaio® client über die Tastenkombination **Umschalt+Strg+F5** das Sicherheitssystem aktualisieren.

Events erstellen

Den Zugriff auf die Funktionen schalten Sie über die **Einstellungen** ein.



Markieren Sie **Events anzeigen** im Bereich **Weitere**, können Sie über den Arbeitsbereich in enaio® client auf die Funktionen zugreifen. Eingerichtete Events werden dort angezeigt.



Oben im Bereich 'Objektsuche' finden Sie den Eintrag **allgemeine Events**. Zugeordnet sind dort alle Events, die durch das Starten des enaio® client, das Beenden, das Ein- und Ausloggen ausgelöst werden.

Die anderen Events werden durch Aktionen mit DMS-Objekten ausgelöst und sind innerhalb der Baumstruktur diesen Objekttypen zugeordnet.

Serverseitige Events

Clientseitige Events

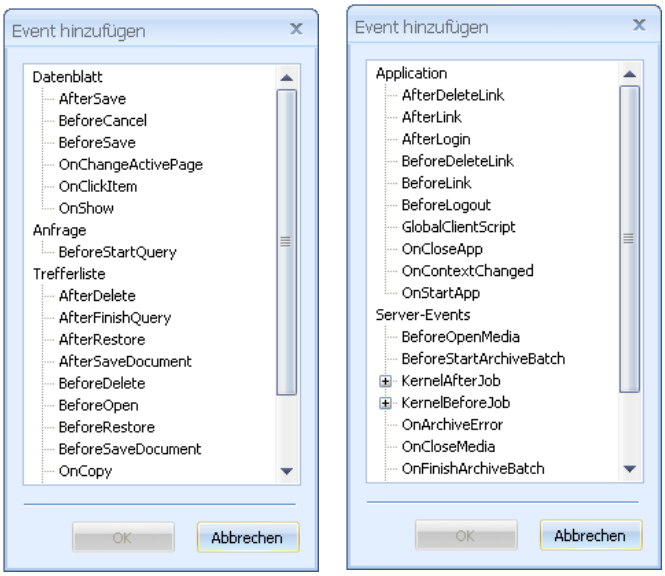
Webclientseitiges Event

Clientseitige Sammeländerungsevents

So erstellen Sie ein Event:

- Markieren Sie den Eintrag **allgemeine Events** oder ein DMS-Objekt.
- Wählen Sie **Event hinzufügen** aus dem Kontextmenü.

  Der Dialog **Event hinzufügen** wird geöffnet.



Aufgelistet sind jeweils die Events, die Sie zuordnen können, entweder Objekt-Events, Applikations-Events oder Server-Events.

- Wählen Sie durch Doppelklick das gewünschte Event.

  Das Editorfenster wird geöffnet.

- Tragen Sie in das Editorfenster das VB-Skript ein.
- Klicken Sie auf die Schaltfläche **Skript speichern**.

Das Event wird in der Datenbank gespeichert.

Jedes Event kann nur einmal einem Objekt zugeordnet werden.

> Wenn Sie über Skripte auf Dialogelemente Bezug nehmen, die Sonderzeichen enthalten, können Fehler auftreten. Verwenden Sie in diesem Fall die internen Namen für Bezüge auf Dialogelemente.

Skripte importieren

Events, die OPTIMAL SYSTEMS für Sie schreibt, erhalten Sie als verschlüsselte Datei, die Sie importieren, einem DMS-Objekt oder der Applikation zuordnen und in der Datenbank speichern.

Sie können ebenfalls Events in enaio® editor-for-events als verschlüsselte Datei speichern und importieren.

> Unverschlüsselte Dateien können nicht importiert werden.

So importieren Sie ein Event:

- Markieren Sie den Eintrag **allgemeine Events** oder ein DMS-Objekt.

- Wählen Sie **Event importieren** aus dem Kontextmenü.
- Wählen Sie über den Dateiauswahldialog die Datei.

  Verschlüsselte Event-Dateien führen die Endung 'evc'.

  Das Event wird im Arbeitsbereich angezeigt und das Editorfenster mit dem VB-Skript geöffnet.

Sie erhalten einen Hinweis, wenn die Bezeichnung des zugeordneten DMS-Objekts nicht mit der Bezeichnung des DMS-Objekts in der Event-Datei übereinstimmt.

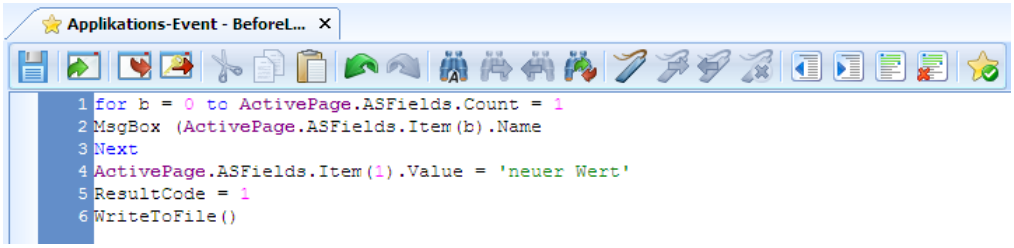
- Klicken Sie im Editorfenster auf die Schaltfläche **Skript speichern**.

Das Event wird in der Datenbank gespeichert.

Das Editorfenster

Wenn Sie ein Event erstellen, dann tragen Sie das VB-Skript in das Editorfenster ein und bearbeiten es dort. Importieren Sie ein Event, wird das VB-Skript unverschlüsselt im Editorfenster angezeigt.

Öffnen Sie ein Event aus der Datenbank, werden Datum, Uhrzeit und Benutzername der letzten Änderung Statusleiste angezeigt.



Auf der Symbolleiste des Editorfensters finden Sie folgende Schaltflächen:

Skript speichern

Sie speichern das Skript in der Datenbank (**STRG-S**).

Skript Importieren

Sie importieren ein Skript aus einer Datei.

Skript Exportieren

Sie exportieren das Skript in eine Datei.

Verschlüsselter Export

Sie exportieren das aktuelle Skript verschlüsselt in eine Datei.

Ausschneiden

Sie schneiden den markierten Text aus.

Kopieren

Sie kopieren den markierten Text in die Zwischenablage.

Einfügen

Sie fügen den Text aus der Zwischenablage an der aktuellen Cursorposition ein.

Rückgängig

Sie machen die letzte Aktion rückgängig.

Wiederherstellen

Stellt die vorherige Version wieder her, nachdem eine Aktion rückgängig gemacht wurde.

Suchen

Sie geben einen Suchbegriff und eine Suchrichtung ab.

Abwärts

Sie suchen das nächste Auftreten des Ausdrucks.

Aufwärts

Sie suchen das vorherige Auftreten des Ausdrucks.

Ersetzen

Sie geben einen Suchbegriff und einen Begriff zum Ersetzen an.

Lesezeichen ein-/ausschalten

Setzt in der aktuellen Zeile ein Lesezeichen bzw. entfernt es.

Nächstes Lesezeichen

Geht zur Zeile mit dem nächsten Lesezeichen.

Vorhergehendes Lesezeichen

Geht zur Zeile mit dem vorhergehenden Lesezeichen.

Alle Lesezeichen löschen

Löscht alle Lesezeichen.

Einzug verkleinern

Verkleinert die Einzugsebene der markierten Zeilen.

Einzug vergrößern

Vergrößert die Einzugsebene der markierten Zeilen.

Block auskommentieren

Kommentiert die markierten Zeilen aus.

Auskommentierung des Blocks aufheben

Hebt die Kommentierung der markierten Zeilen auf.

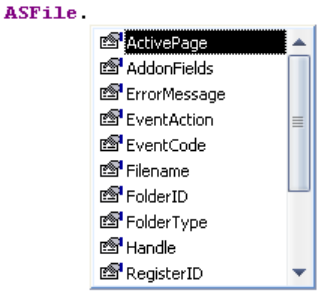
Syntaxprüfung

Startet die Syntaxprüfung.

Die meisten Funktionen stehen auch im Kontextmenü zur Verfügung.

Mit **Strg+G** geben Sie eine Zeilennummer an, zu der gesprungen wird.

enaio® editor-for-events unterstützt Intellisense: Haben Sie den Namen eines Objekts getippt und drücken die Punkttaste, erscheint ein Kontextmenü mit allen Methoden und Eigenschaften dieses Objekts:



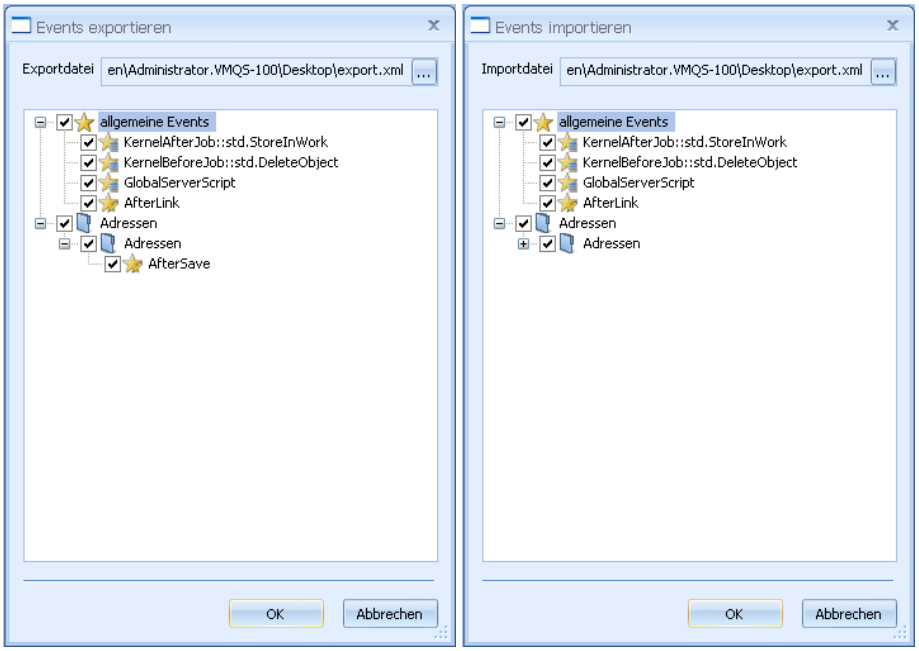
Export/Import

Über das Editorfenster können Sie einzelne Skripte importieren und exportieren (auch verschlüsselt).

Über den Bereich 'Objektsuche' in enaio® client können Sie Events exportieren und importieren.

Die so exportierten Skriptdateien erhalten die Dateiendung *.evc.

Wenn Sie im Bereich 'Objektsuche' Ihren Benutzernamen markieren, finden Sie im Kontextmenü die Funktionen **Eventexport** und **Eventimport**.



Der Eventexport erzeugt mehrere Dateien: In einer XML-Datei werden die Zuordnungen von Objekttyp und Skript gespeichert. Die eigentlichen Skripte dagegen werden einzeln und verschlüsselt in das gleiche Verzeichnis als *.evc-Dateien exportiert.

Sie markieren die Events, die exportiert werden sollen und bestätigen Ihre Auswahl per Mausklick auf **OK**.

Beim Eventimport wählen Sie die XML-Datei mit den Zuordnungen und geben an, welche Events importiert werden sollen. Die Skripte müssen im gleichen Verzeichnis liegen wie die XML-Datei.

Sie erhalten entsprechende Hinweise, falls den Objekttypen bereits Skripte zugeordnet sind und können entscheiden, ob diese überschrieben werden sollen.

Debugging

Zum Testen der Events schalten Sie in enaio® client den Debug-Modus ein. Im Debug-Modus werden die Events nicht ausgeführt, sondern bei der entsprechenden Aktivität das Skript im Eventeditor geöffnet. Dort kann dann das Skript schrittweise ausgeführt werden, wobei die aktuellen Werte der Variablen angezeigt werden, die Werte können geändert werden.

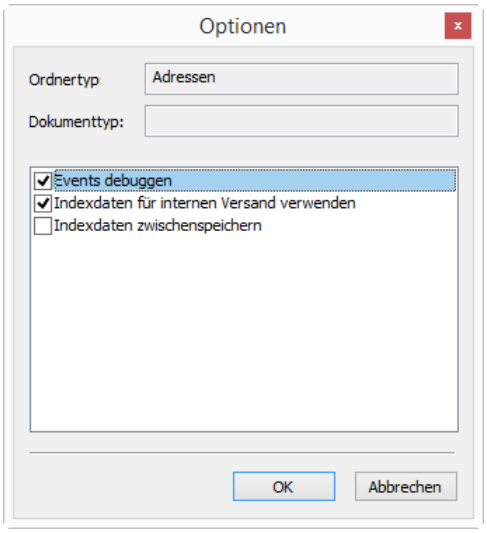
> Im Debug-Modus kann das Skript nicht editiert werden.

Neben den Applikations- und Objekt-Events können ebenfalls die Skripte, die in einen Workflow-Vorgang eingebunden sind, im Eventeditor geöffnet und schrittweise ausgeführt werden.

Den Debug-Modus für die Applikations-Events schalten Sie über die **Einstellungen** ein. Im Bereich **Weitere** markieren Sie **Applikationsevents debuggen**.

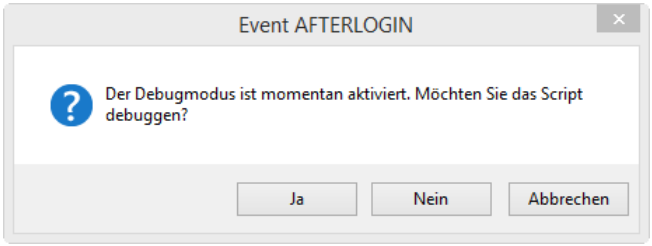
Für die einzelnen Objekttypen schalten Sie den Debug-Modus über den Optionen-Dialog aus dem Kontextmenü ein.

Markieren Sie hier **Events debuggen**.



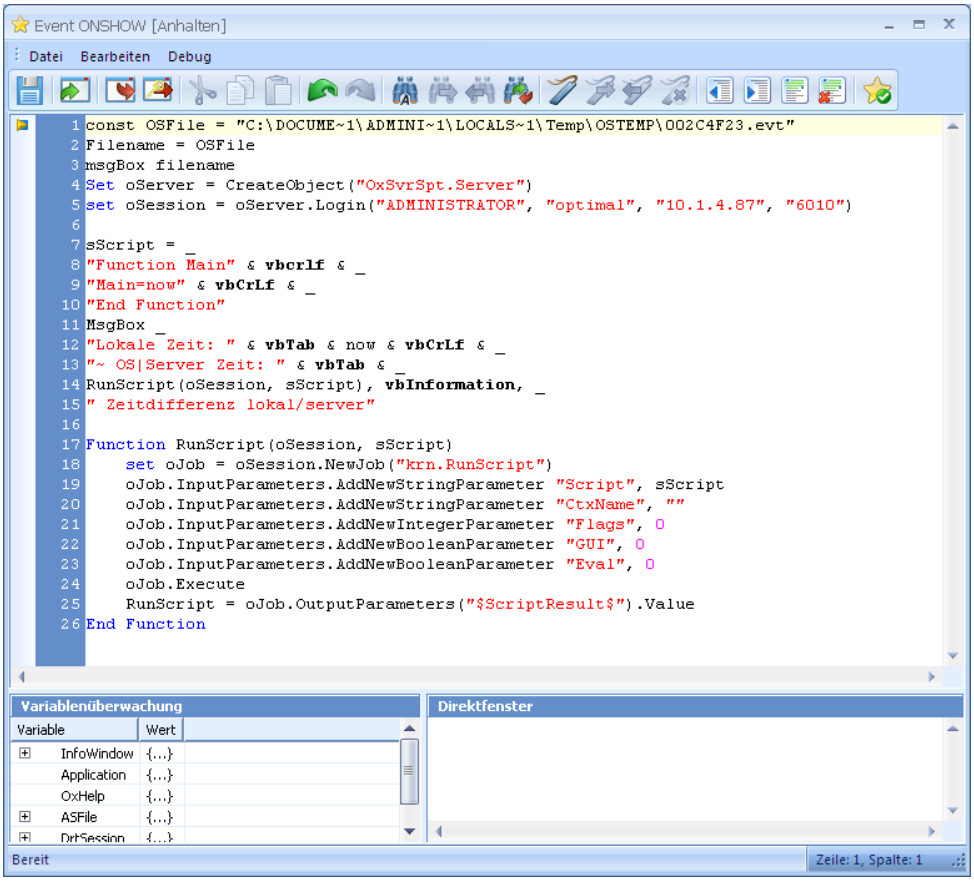
Wollen Sie Workflow-Skripte im Debug-Modus testen, markieren Sie über die **Einstellungen** im Bereich **Weitere** die Option **Workflowevents debuggen**.

Ist der Debug-Modus eingeschaltet, wird, sobald ein entsprechendes Event eintritt, zuerst ein Rückfrage-Dialog angezeigt.



Die können mit **Ja** den Debug-Modus bestätigen, mit **Nein** das Event ausführen lassen und mit **Abbrechen** den Vorgang ohne Event weiterführen.

Bestätigen Sie den Debug-Modus, wird das Skript geöffnet.



Im Debug-Modus ist das Fenster in drei Bereiche unterteilt, den Skript-Bereich, den Bereich **Variablenüberwachung** und das **Direktfenster**.

Im Bereich **Variablenüberwachung** sind die aktuelle im Skript verfügbaren Variablen und Objekte und deren Werte aufgelistet. Die Daten sind nicht editierbar.

Im **Direktfenster** können Sie Ausdrücke prüfen und Werte von Variablen ändern:

- Tragen Sie einen Ausdruck ein und bestätigen Sie mit **Enter**, wird der Ausdruck geprüft.
- Weisen Sie einer Variablen einen Wert zu und bestätigen Sie mit **Umschalt+Enter**, wird der Wert der Variable geändert.

Das Event-Skript testen Sie über die folgenden Schaltflächen:

F9	Haltepunkt ein/aus
F5	Debug starten bis zum nächsten Haltepunkt oder bis zum Ende
Umschalt F5	Debugging beenden
F8	In Aufruf oder Funktion springen

	**Umschalt F8**	Führt die nächste Anweisung in einem Schritt aus
	**Umschalt Strg F8**	Rückschritt, aus der Funktion herausspringen

Über **Strg+B** wird der Dialog **Haltepunkte** geöffnet, in dem alle definierten Haltepunkte aufgelistet werden.

Index