

# Softwaredokumentation

## enaio® server-api

Version 8.50

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

Copyright 1992 – 2018 by      OPTIMAL SYSTEMS GmbH

01.03.2018

Version 8.50

# Inhalt

Inhalt .....	iii
Einleitung .....	4
Engine .....	4
Schnittstellenbibliotheken .....	5
Realisierung einer Archivanbindung .....	6
Testmöglichkeiten .....	7
Glossar .....	8
enaio® Server-API Engine-Verzeichnis .....	9
enaio® Server-API Engine-Dokumentation .....	11
Abonnement-Engine (Namespace abn) .....	11
ADO-Datenbank-Engine (Namespace ado) .....	29
Convert-Engine (Namespace cnv) .....	31
DMS-Engine (Namespace dms) .....	38
Medizin-Engine (Namespace med) .....	127
MNG-Engine (Namespace mng) .....	152
OCR-Engine (Namespace ocr) .....	165
Standard-Engine (Namespace std) .....	166
Volltext-Engine (Namespace vtx) .....	190
Workflow-Engine (Namespace wfm) .....	196
Core Services .....	304
OxSvrSpt .....	343
Allgemeine Beschreibung .....	343
Module .....	344
Datenstrukturen .....	360
Klassenhierarchie .....	401
Index .....	403

# Einleitung

enaio® ist ein leistungsfähiges Dokumentenmanagement-, Workflow- und Archivsystem. Ein besonderes Kennzeichen der Produktfamilie ist die hohe Konfigurierbarkeit und ihre Schnittstellenstärke, die es erlaubt, an den unterschiedlichsten Stellen eine Integration in andere Systeme vorzunehmen.

Im nachfolgenden Dokument werden die durch die enaio® Server-API bereitgestellten Funktionen detailliert beschrieben. Das Dokument hat den Charakter einer Funktionsreferenz. Nähere Informationen über den Aufbau von enaio® und der einzelnen Komponenten sind den entsprechend verfügbaren Dokumentationen zu entnehmen.

Der enaio® Server dient als Laufzeitumgebung für diverse Engines im Archiv, DMS und Workflow-Umfeld. So werden durch den Server verschiedene Aufgaben zum dokumentenorientierten Informationsfluss wahrgenommen. Dies sind etwa die Aufnahme, Verwaltung und Bearbeitung von Dokumenten und den dazugehörigen Indexdaten, die Volltextrecherche, das Workflowmanagement, die Archivierung und viele weitere Funktionen.

## Engine

Eine Engine – auch Executor genannt – wird durch den Applikationsserver geladen und dadurch befähigt, Jobs auszuführen. Jeder Executor verwaltet einen oder mehrere Namespaces, in denen die Serverjobs organisiert sind. Folgende Engines werden standardmäßig ausgeliefert:

- § DMS-Engine: zur Recherche und zur Manipulation von Index- und Dokumentdaten
- § Workflow-Engine: Bearbeitung und Verwaltung von Workflowprozessen und Modellen
- § Standard-Engine: Sammlung von diversen Funktionen zur Archivierung, zum Datei- und Dokumententransfer
- § Volltext-Engine: Abfrage der Volltextengines (Microsoft SQL Volltext Server, Convera RetrievalWare)
- § OCR-Engine: optische Zeichenerkennung aus Bildern oder gescannten Belegen
- § Abonnement-Engine: Benachrichtigungsfunktionen bei Änderungen am Dokumentenbestand
- § Core-Services: Initialisierung, Lizenzierung und Sessionmanagement
- § Datenbankpiping: ermöglicht einen Zugriff auf die Datenbank über die Serverschnittstelle in Form eines internen Formates oder als Active Data Objects

## Serverjob

Jobs sind Aufgaben, die durch den Server ausgeführt werden sollen, und eine bestimmte Akquise, Manipulation von Daten oder Steuerungsfunktionen abbilden. Somit lassen sich Jobs auch mit Funktionen vergleichen. Ein Serverjob hat folgenden allgemeinen Aufbau:

- § Name des Jobs: Der Name setzt sich aus der Namespace – in den der Job implementiert ist – und der Jobbezeichnung zusammen. (z. B. dms.XMLInsert)
- § Eingabeparameter
- § Eingabe-Dateiliste: Wenn dem Job Dateien übergeben werden müssen, wird hier der absolute Pfad und Name der jeweiligen Datei übergeben.

## § Ausgabeparameter

§ Ausgabe-Dateiliste: Wenn der Job Dateien zurückliefert, wird hier der absolute Pfad und Name der jeweiligen Datei übergeben.

§ Rückgabewert: Jeder Job generiert einen Rückgabewert. Dieser ist im Erfolgsfall stets '0'. Ist ein Fehler aufgetreten, so kann er über den Rückgabewert grob qualifiziert werden.

Zur Ausführung der Jobs muss zwischen der anfordernden Instanz -auch Client genannt- und dem Server eine so genannte Session erzeugt werden. Diese besteht einerseits aus einer Verbindung über ein technisches Protokoll (z. B. TCP) und andererseits aus verschiedenen Informationen zu dieser Verbindung, wie z. B. Authentifizierungs-, Stations- und Lizenzdaten. Über das technische Protokoll wurde zur Formulierung der Anfragen und Ergebnisse der Bearbeitung das Protokoll XML-RPC gelegt. Dieses Protokoll legt fest, wie Anfragen, Parameter und Ergebnisse dargestellt werden müssen. Den Verlauf Kommunikation zwischen Client und Server entspricht folgendem Schema:

§ Aufbau einer TCP-Verbindung mit dem enaio®-Server

§ Initialisierung einer Session mit entsprechenden Parametern

§ Jobaufrufe und Empfang der Antwortdaten

§ Beenden der Verbindung

Der Client ruft einen Job auf, indem die Jobbezeichnung und die dazugehörigen Parameter in XML verpackt und zum Server gesandt werden. Danach wartet der Client auf eine Antwort. Der Server-Kernel hingegen interpretiert die empfangenen Daten. Es wird ermittelt, welche Engine den Job ausführen kann und an dessen Queue-Mechanismus zur Bearbeitung übergeben. Das Ergebnis wird dann von der Engine über den Server-Kernel an den Client übermittelt. Die Jobs, die durch die Engines bereitgestellt werden, sind in der nachfolgenden Referenz beschrieben.

Das XML-RPC-Format sieht vor, dass alle Parameter innerhalb einer XML-Struktur mit entsprechenden Typen übertragen werden. Bei der Übertragung von Dateien (also Binärdaten) wäre es deshalb notwendig, die Dateien durch eine MIME- Kodierung in einen String umzuwandeln. Aus Gründen der Performance und des Speicherbedarfs wurde hier vom Standard abgewichen. Die Dateien werden nach dem Versand der Jobparameter als TCP-Stream gesondert übertragen. Einige Parameter, insbesondere XML-Strukturen, werden zur Übertragung aber in das MIME-Format konvertiert, um evtl. Steuerzeichen zu entfernen.

## Schnittstellenbibliotheken

Um die interne Kommunikation vor den fachlichen Anforderungen zu verbergen, stehen mehrere Möglichkeiten offen, das Protokoll auch ohne tiefere Kenntnisse der Kommunikationsabfolge zu benutzen. Zu diesem Zwecke existieren Klassen und Bibliotheken, die von OPTIMAL SYSTEMS bereitgestellt werden.

Dies sind:

§ COM – Schnittstelle der Kommunikationsbibliothek oxsvrspt für Multithreadumgebungen

§ Java-Interface zur Kommunikation mit dem enaio® Server

§ COM-Schnittstelle der Kommunikationsbibliothek oxmljsc

§ Diverse weitere Bibliotheken, die Serveraufrufe kapseln

Vom prinzipiellen Aufbau gleichen sich die Schnittstellen in der Benutzung. Nach einer Initialisierung und der Herstellung einer Verbindung zum Applikationsserver werden Jobobjekte aufgebaut, denen Eingabeparameter und Eingabedateien übergeben werden. Nach der Ausführung können Ausgabeparameter und Ausgabedateien vom Objekt gelesen werden. Darüber hinaus wird ein

Fehlerstack bereitgestellt, der mögliche fachliche und technische Fehlermeldungen aufnehmen kann. Eingabe- und Ausgabelisten werden je nach verwendeter Programmiersprache als Hashlists, Arrays oder sonstige Objekte dargestellt.

Vorrangig sollte die oxsvrspt.dll als Bibliothek verwendet werden. Ausführliche Informationen zur Verwendung der Bibliotheken finden Sie im Abschnitt 'OxSvrSpt'.

## Java-Schnittstelle

Die von OPTIMAL SYSTEMS bereitgestellten Java-Schnittstellenbibliotheken unterstützen das Erstellen von Client-Anwendungen, die mit enaio®-Server kommunizieren. Insgesamt sind es drei Bibliotheken:

§ Applikationsserver Proxy

§ Java DRT Layer (JDL)

§ Java Objekt Layer

Die drei Schnittstellenbibliotheken decken unterschiedliche Anwendungsbereiche ab und stellen verschiedene Abstraktionsebenen in der Sicht auf das OS-System dar. Während Applikationsserver Proxy und JDL zwei aufeinander aufbauende Stufen in der Kommunikation mit enaio® Server sind, ermöglicht der Java Objekt Layer eine objekt-orientierte Arbeitsweise mit den Ein- und Ausgangsdaten der Jobs. Eine voll ständige Dokumentation der Java- Schnittstelle ist separat verfügbar.

## Realisierung einer Archivanbindung

### Allgemeines

Ziel einer Archivanbindung für verschiedene Applikationen ist es, Dokumente, die mit Struktur- und Indexmerkmalen versehen sind, im DMS abzulegen, im Datenbestand zu recherchieren, Dokumente zu downloaden und ggf. zu löschen. Diese Szenarien sollen an Hand von nachfolgenden Beispielen demonstriert werden. Wie bereits beschrieben, werden die Funktionen durch den Aufruf von Serverjobs durchgeführt, die in der DMS-Engine und der Standard-DMS-Engine implementiert wurden.

Zunächst eine kurze Einführung zur Struktur des Systems. Weitergehende Informationen sind den Administrationshandbüchern von enaio® zu entnehmen. enaio® verwendet folgende DMS-Objekte für die Abbildung von Informationsstrukturen:

#### § **Schrank:**

Der Begriff Schrank wurde als Analogie zu einem Aktenschrank gewählt. Ein Aktenschrank enthält Ordner, Register und Dokumente. Er stellt die oberste Verwaltungsebene im DMS dar. Alle weiteren DMS-Objekte können nur als Subobjekt eines Schrankes existieren. Ein Schrank erhält als einziges Attribut einen Namen.

#### § **Ordner:**

Ordner werden durch Indexdaten beschrieben und sind Container für darunter liegende Objekte. Sie sind vergleichbar mit Ordnern im Dateisystem auf Wurzelebene. Pro Schrank existiert immer nur ein Ordnertyp.

#### § **Register:**

Register dienen der weiteren Feingliederung der Informationsstruktur. Pro Schrank kann es mehrere Registertypen geben, die sich nach der Struktur der Indexdaten unterscheiden.

#### § **Dokumente:**

Dokumente zeichnen sich neben den Indexdaten zusätzlich dadurch aus, dass mit ihnen Dokumentdateien assoziiert sind. Jeder Dokumenttyp besitzt die Eigenschaft eines Haupttyps, wodurch gekennzeichnet ist, ob es sich bei den assoziierten Dateien um Images, Windows-Quelldokumente oder andere, von ihrem Wesen nach unterschiedlichen Dateitypen handelt. Der enaio® client verhält sich je nach Haupttyp beim Erfassen und Ausgeben der Dokumente unterschiedlich.

Die Klassen von Ordnern, Registern und Dokumenten heißen Objekttypen. So ist beispielsweise ein Kundenordner oder eine Rechnungsart ein Objekttyp. Alle Instanzen der Klasse, also die einzelnen Ordner oder Belege selbst verfügen über die gleichen Felder zur Indexierung. Bei Dokumenten wird über den Objekttypen auch der Haupttyp festgelegt. Die Struktur der Indexdaten wird durch das Objektmodell vorgegeben. So kann für jeden Objekttypen eine eigene Menge von Feldern definiert werden, die zur Verschlagwortung und zur Recherche dienen.

Nachfolgend werden Dokumente, Register und Ordner als Objekte bezeichnet, sofern eine genauere Spezifikation unerheblich ist. Jedes Objekt ist durch Indexdaten und so genannte Basisparameter gekennzeichnet. Basisparameter werden automatisch vom System vergeben und sind zur internen Verwaltung der Objekte vorgesehen. Sie beinhalten Informationen über den Anleger eines Objektes, seine ObjektID, Änderungsdaten usw.

Jedes Objekt im DMS wird durch seinen Objekttypen und seine ObjektID eindeutig beschrieben. Für die meisten Jobs werden die Objekttypen und ObjektIDs als Eingabeparameter erwartet.

Bei einer Recherche sollen anhand von Suchparametern, Indexdaten und Basisparameter von Objekten zur weiteren Verarbeitung, ermittelt werden. Über die ermittelten ObjektIDs und Objekttypen können dann weitere Recherchen durchgeführt werden oder etwa die Dokumentdateien selbst gedownloadet werden.

Bei einem Import werden durch die aufrufende Applikation Indexdaten und ggf. Dokumentdateien vorgegeben und daraufhin im System neue Objekte angelegt.

## Szenarien und zugehörige Jobs

Nachfolgend eine kurze Zusammenstellung der Jobs, die für eine Archivanbindung benötigt werden.

- § **Session erzeugen:** [krm.SessionAttach](#)
- § **User anmelden:** [krm.SessionLogin](#)
- § **Objektdefinition lesen:** [dms.GetObjDef](#)
- § **Recherche nach Ordnern, Registern, Dokumenten:** [dms.GetResultList](#)
- § **Dokument downloaden:** [std.StoreInCacheByID](#)
- § **Objekte einfügen:** [dms.XMLInsert](#)
- § **Objekt löschen:** [dms.XMLDelete](#)
- § **Session beenden:** [krm.SessionLogout](#)

## Testmöglichkeiten

### Testprogramm axlabjobs.exe

Für Tests einzelner Jobs des enaio®-Servers und seiner Engines stellt OPTIMAL SYSTEMS das Programm `axlabjobs.exe` zur Verfügung. Die Jobs können mit beliebigen Parametern und beliebig oft ausgeführt werden. Über einen Connect-String kann angegeben werden, an welchem Server sich das TestLab anmelden soll. Es ist möglich, mit vielen TestLabs an einem Server zu arbeiten, um somit Belastungstests durchzuführen. Das Programm wird standardmäßig im Serververzeichnis installiert.

Der Referenz zu den einzelnen Jobs kann jeweils entnommen werden, welche Parameter für das Testprogramm anzugeben sind.

Zur Überwachung von Jobaufrufen steht der enaio® enterprise-manager zur Verfügung. Dort können sowohl Rechner wie auch Jobs spezifiziert werden, die überwacht werden sollen. Zu Jobs, zu denen Dateien mitgeschickt werden, können auch diese Dateien über ein temporäres Verzeichnis zugänglich gemacht werden.

Auf die Funktionen zur Jobüberwachung wird im enaio® enterprise-manager über den Bereich 'Erweiterte Administration / Überwachung / Jobaufrufe' zugegriffen.

## Glossar

Dieser Abschnitt soll zur Erklärung einiger Begriffe dienen, die in der Dokumentation der Serverjobs auftauchen.

**Cache-Verzeichnis** – ist ein Verzeichnis unterhalb des Serverpfades (..\server\CACHE), welches dazu benutzt wird archivierte Dokumente, die aus Archivierungsmedien gelesen wurden für weitere Zugriffe bereit zu haben, damit der nächste Benutzer nicht auch zeitaufwendig auf das Archivierungsmedium zugreifen muss.

**Flag** - ist ein Parameter, der eine bestimmte Eigenschaft aktivieren bzw. markieren soll.

**OSTEMP-Verzeichnis** – ist ein Verzeichnis, deklariert in der Umgebungsvariable OSTEMP, dass von einigen Jobs zur Zwischenlagerung oder Erzeugung für temporäre Dateien benutzt wird.

**Work-Verzeichnis** – ist ein Verzeichnis unterhalb des Serverpfades (..\server\WORK), welches als Ablageort für alle nicht archivierten Dokumente genutzt wird. Aus Gründen der Performance erfolgt die Ablage in diesem Verzeichnis und nicht in der Datenbank.

Parameter und Rückgabewerte, die in eckige Klammern '[Parameter]' eingeschlossen sind, sind optionale Parameter. Diese Parameter müssen, wenn sie nicht benötigt werden, beim Jobaufruf nicht angegeben werden.



# enaio® Server-API Engine-Verzeichnis

Engine	Beschreibung	Bereiche
<a href="#">Abonnement</a> (abn)	Einrichtung und Steuerung von Abonnements zur Information über Veränderung an DMS-Objekten	
<a href="#">ADO-Datenbank</a> (ado)	Zugriff auf die Datenbank	
<a href="#">Convert</a> (cnv)	Konvertierung und Zugriff auf Bilddateien	
<a href="#">DMS</a> (dms)	Anfragen und Bearbeiten von Indexdaten, DMS Objekten, Relationen und Mappen	<a href="#">XML Import</a> <a href="#">XML Export (Recherche)</a> <a href="#">Sicherheitssystem</a> <a href="#">Relationen und Relationstexte</a> <a href="#">Mappen (Portfolios)</a> <a href="#">Benutzerbezogene Daten</a> <a href="#">Sonstige Jobs</a>
<a href="#">Medizin</a> (med)	Zugriff auf medizinische Informationen	
<a href="#">Benutzer/Gruppen</a> (mng)	Zugriff auf Gruppen und Benutzer von enaio®	
<a href="#">OCR</a> (ocr)	Optische Zeichenerkennung	
<a href="#">Standard</a> (std)	Work-, Cache-, Datei- und Archiv-Verwaltung	<a href="#">Work-, Cache- und Archiv-Verwaltung</a> <a href="#">Datei-Verwaltung</a> <a href="#">Interne Jobs</a> <a href="#">Sonstige Jobs</a>
<a href="#">Volltext</a> (vtx)	Bearbeitung von Volltextanfragen von enaio® client	
<a href="#">Workflow</a> (wfm)	Bearbeitung und Verwaltung von Workflowprozessen und Modellen	<a href="#">Organisationsstruktur</a> <a href="#">Workflowmodell</a> <a href="#">Workflowprozess und Arbeitsschritt</a> <a href="#">Workflow-Maske, Event und Skript</a> <a href="#">Administration und Historienverwaltung</a>

		<a href="#">Administration</a> <a href="#">Historienverwaltung</a> <a href="#">Sonstige Jobs</a> <a href="#">Serverinterne Jobs</a>
<b><a href="#">Core-Services</a></b> (nachfolgende Engines sind direkt im Serverkern implementiert)		
<b><a href="#">Administration</a></b> (adm)	Verwaltung von Systemdateien	
<b><a href="#">Kernel</a></b> (krn)	Batch-Verwaltung, Serverüberwachung, Registry-Verwaltung und Administration geladener Engines zur Laufzeit	<a href="#">Registry-Verwaltung</a> <a href="#">Batch-Verwaltung</a> <a href="#">Server-Verwaltung</a> <a href="#">Session-Verwaltung</a> <a href="#">Engine-Verwaltung</a> <a href="#">Sonstige Jobs</a>
<b><a href="#">Lizenz</a></b> (lic)	Lizenzverwaltung für das gesamte enaio® – System	
Datenübernahme-Services (Namespace dtr)	Serverseitiger Aufruf des Datenübernahmeservers	

# enaio® Server-API Engine-Dokumentation

## Abonnement-Engine (Namespace abn)

In der Abonnement-Engine sind Funktionen zur Einrichtung und Steuerung von Abonnements implementiert. Diese dienen dazu, den Anwender über eine Veränderung an DMS-Objekten zu informieren.

In Mehrserversystemen kann ein Server nur die Clients benachrichtigen, die mit dem Server verbunden sind.

- § abn.Add
- § abn.CheckOsrevisit
- § abn.GetAboGrpList
- § abn.GetDocList
- § abn.GetGroupList
- § abn.GetRequestList
- § abn.GetUserList
- § abn.NotifyRequestAbo
- § abn.NotifyAbonnement
- § abn.Remove
- § abn.RemoveAboIdent
- § abn.UpdateReqAboGrp
- § abn.RemoveAllObjNotifyFromUser
- § abn.RemoveObjNotifyFromUser
- § abn.ConfirmAboRead
- § abn.RemoveObjRevisitNotifyFromUser
- § abn.SetObjRevisitClosed
- § abn.SetObjRevisitOpen
- § abn.ChangeRevisitUser
- § abn.AddRevisit
- § abn.UpdateRevisit
- § abn.GetSubscriptions
- § abn.GetRevisits
- § abn.SetOsInformed
- § abn.ResetOsInformed

§ abn.GetRecentObjects

§ ado.ExecuteSQL

## abn.Add

### Beschreibung:

Dieser Job erstellt ein Abonnement für das angegebene Objekt.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts, über das benachrichtigt werden soll

ObjectType (INT): Typ des Objekts

ActionFlags (INT): Aktion, die mit dem Dokument durchgeführt wird und zur Benachrichtigung führt

§ 2 = Benachrichtigung, wenn ein Dokument angelegt wurde (nur für abonnierte Anfrage)

§ 3 = Benachrichtigung, wenn Indexdaten geändert wurden (nur für abonnierte Dokumente)

§ 4 = Benachrichtigung, wenn das Dokument geändert wurde

§ 27 = Benachrichtigung, wenn das Dokument gelöscht wurde

§ 39 = Benachrichtigung, wenn ein Standort hinzugefügt wurde

Channel (INT): Benachrichtigungs-Kanal (die Art der Benachrichtigung)

§ 0 - Benachrichtigung via internem Kanal (oxmljsc)

§ 1 - Benachrichtigung via Email

AboGrpID (STRING): diese ID fasst alle Aktionen eines Abonnements zusammen

[Benutzer] (STRING): Name des Benutzers, der benachrichtigt werden soll

[Alias] (STRING): Infotext zum Abonnement (max. 255 Zeichen)

[Product] (STRING): String, der die Programm-Instanz beschreibt, z. B. 'ax.exe'

[Confirm] (INT):

§ 1 = wurde Benutzer/Gruppe benachrichtigt, muss vor dem Löschen der Nachricht, diese als gelesen gekennzeichnet werden;

§ 2 = bevor diese als gelesen gekennzeichnet werden kann, wird das Systempasswort des Benutzers abgefragt; ansonsten 0

[Station] (STRING): Name der Station, die ausschließlich die Benachrichtigung erhalten soll

[Mail] (STRING): Email-Adresse (max. 255 Zeichen) für Benachrichtigung (mehrere Adressen werden durch Semikolon getrennt)

[AboType] (INT): Unterscheidet zwischen Anfrage-Abo und Dokument-Abo

§ 0 = Dokument-Abo

§ 1 = Anfrage-Abo

[RequestFormat] (STRING): Format der Anfrage. Default ist 'ABN' für das native SQL-Format. In diesem Fall wird die Anfrage im Parameter 'AboRequest' ausgewertet. Wird 'XML' angegeben, wird eine DMSQuery-Anfrage im 'XmlRequest' Parameter erwartet.

[AboRequest] (STRING): SQL-Statement für Anfrage-Abonnements

Die Anfrage muss aus Kleinbuchstaben bestehen und mit 'select count(distinct d.id)' oder mit 'select distinct d.id' beginnen, sonst wird sie zurückgewiesen.

Je nach Haupttyp des angefragten Objektes muss 'o.id' für Ordner, 'r.id' für Register oder 'd.id' für Dokumente als Bezeichner verwendet werden.

[XmlRequest] (STRING/Base64 ): Abo Anfrage im [DMSQuery](#) XML Format

[GroupID] (STRING): GUID der Gruppe, die benachrichtigt werden soll

[UserID] (STRING): Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Siehe auch:**

[abn.Remove](#)

## [abn.CheckOsrevisit](#)

**Beschreibung:**

Dieser Job führt eine Prüfung der Wiedervorlagen (Tabelle 'osrevisit') durch und sendet eine Benachrichtigung an die Benutzer, für die Wiedervorlagen definiert sind, falls die entsprechenden Zeitstempel dies erfordern. Für die Benachrichtigung der betroffenen Benutzer verwendet dieser Job den Job 'abn.Osrevisit'. Dieser Job wird periodisch intern aufgerufen. Dafür muss ein Batch in der Registry eingerichtet sein.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Siehe auch:**

[Batch-Verwaltung](#)

## [abn.GetAboGrpList](#)

**Beschreibung:**

Dieser Job liefert über die AboGrp-ID Informationen zum Abonnement.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

AboGrpID (STRING): ID, fasst alle definierten Aktionen des Abonnements zusammen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Rückgabewerte:**

Abo[1..n] (STRING): Semikolon-separierte Informationen zum Abonnement

§ ID der Dokumentes

§ Typ des Dokumentes

- § Infotext
- § ID des Abonnements
- § Confirm: 1 = wurde Benutzer/Gruppe benachrichtigt, muss vor dem Löschen der Nachricht, diese als gelesen gekennzeichnet werden; ansonsten 0
- § UserID: Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden
- § GroupID: ID der Gruppe, die benachrichtigt werden soll
- § Abotyp: 0 = Dokument-Abo; 1 = Anfrage-Abo
- § Aktion, die eine Benachrichtigung auslöst
  - § 2 = Benachrichtigung, wenn ein Dokument angelegt wurde (nur für abonnierte Anfrage)
  - § 3 = Benachrichtigung, wenn Indexdaten geändert wurden (nur für abonnierte Dokumente)
  - § 4 = Benachrichtigung, wenn das Dokument geändert wurde
  - § 27 = Benachrichtigung, wenn das Dokument gelöscht wurde
  - § 39 = Benachrichtigung, wenn ein Standort hinzugefügt wurde
- § Benachrichtigungsart
  - § 0 = Benachrichtigung via internem Kanal (oxmljsc)
  - § 1 = Benachrichtigung via Email; E-Mail Adresse
- § Name des Benutzers, der benachrichtigt werden soll
- § SQL-Statement für Anfrage-Abo

## abn.GetDocList

### Beschreibung:

Dieser Job ermittelt alle Objekte, die der angegebene Benutzer abonniert hat.

### Parameter:

Flags (INT): Steuert den Umfang der Ausgabe

- § 0 = Es werden die Dokumente für den angegebenen Benutzernamen ermittelt
- § 1 = Zusätzlich werden die Dokumente ermittelt, die für Benutzergruppen angelegt wurden in denen sich der angegebene Benutzer z. Z. befindet

Benutzer (STRING): Name des Benutzers

[Product] (STRING): Suche wird für die Programm-Instanz eingeschränkt

[Station] (STRING): Suche wird für die Bezeichnung der Benutzer-Station eingeschränkt

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### Rückgabewerte:

Document[1..n] (STRING): Semikolon-separierte Informationen zum abonnierten Dokument

- § ID des Dokumentes
- § Typ des Dokumentes
- § Infotext

- § ID des Abonnements
- § Confirm: 1 = wurde Benutzer/Gruppe benachrichtigt, muss vor dem Löschen der Nachricht, diese als gelesen gekennzeichnet werden; ansonsten 0
- § UserID: Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden
- § GroupID: ID der Gruppe, die benachrichtigt werden soll
- § Abotyp: 0 = Dokument-Abo; 1 = Anfrage-Abo
- § Aktion, die eine Benachrichtigung auslöst
  - § 2 = Benachrichtigung, wenn ein Dokument angelegt wurde (nur für abonnierte Anfrage)
  - § 3 = Benachrichtigung, wenn Indexdaten geändert wurden (nur für abonnierte Dokumente)
  - § 4 = Benachrichtigung, wenn das Dokument geändert wurde
  - § 27 = Benachrichtigung, wenn das Dokument gelöscht wurde
  - § 39 = Benachrichtigung, wenn ein Standort hinzugefügt wurde
- § Benachrichtigungsart
  - § 0 = Benachrichtigung via internem Kanal (oxmljsc)
  - § 1 = Benachrichtigung via Email; E-Mail Adresse
- § AboGrp: fasst alle Datensätze der verschiedenen Aktionen eines Abonnements zusammen

## abn.GetUnreadAboCount

### Beschreibung:

Dieser Job ermittelt die Anzahl der gelesenen und ungelesenen Abo-Benachrichtigungen eines Benutzers.

### Parameter:

Flags (INT): z. Z. nicht unterstützt -> 0 übergeben

UserID (INT): Benutzer-ID des Benutzers, dessen Abo-Benachrichtigungen ermittelt werden sollen.

Mode (INT): Legt fest, welche Abo-Benachrichtigungen ermittelt werden (optional):

- § 0: Alle Abo-Benachrichtigungen werden ermittelt (Default).
- § 1: Benachrichtigungen aus Abonnements, die der Benutzer selbst angelegt hat, werden nicht berücksichtigt.
- § 2: Zu jedem Abonnement wird nur die jeweils letzte Abo-Benachrichtigung ermittelt.
- § 3: Benachrichtigungen aus Abonnements, die der Benutzer selbst angelegt hat, werden nicht berücksichtigt und zu jedem Abonnement wird nur die jeweils letzte Abo-Benachrichtigung ermittelt (Kombination aus Mode 1 und Mode 2).

### Rückgabe:

Read (INT): Anzahl der gelesenen Abo-Benachrichtigungen

Unread (INT): Anzahl der ungelesenen Abo-Benachrichtigungen

## abn.GetGroupList

### Beschreibung:

Dieser Job ermittelt die Benutzergruppen, die das angegebene Objekt abonniert haben.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des zu überprüfenden Objekts

ObjectType (INT): Typ des Objekts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Rückgabewerte:**

Group[1..n] (STRING): durch Semikolon getrennte Informationen zur Gruppe und Abo

§ ID der Gruppe

§ Name der Gruppe

§ GUID der Gruppe

§ Beschreibung zur Gruppe

§ Infotext zum Abonnement

§ Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden

§ AboGrp: fasst alle Datensätze der verschiedenen Aktionen eines Abonnements zusammen

§ Aktion, die eine Benachrichtigung auslöst

§ 2 = Benachrichtigung, wenn ein Dokument angelegt wurde (nur für abonnierte Anfrage)

§ 3 = Benachrichtigung, wenn Indexdaten geändert wurden (nur für abonnierte Dokumente)

§ 4 = Benachrichtigung, wenn das Dokument geändert wurde

§ 27 = Benachrichtigung, wenn das Dokument gelöscht wurde

§ 39 = Benachrichtigung, wenn ein Standort hinzugefügt wurde

§ Benachrichtigungsart

§ 0 = Benachrichtigung via internem Kanal (oxmljsc)

§ 1 = Benachrichtigung via Email; E-Mail Adresse

§ ID des Abonnements

## abn.GetRequestList

**Beschreibung:**

Dieser Job ermittelt alle Anfrage-Abonnements, die der angegebene Benutzer abonniert hat.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Benutzer (STRING): Name des Benutzers

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Rückgabewerte:**

Request[1..n] (STRING): durch Semikolon getrennte Informationen zur Anfrage

§ Infotext zum Abo



- § Typ des Objektes
- § ID des Abonnements
- § 1 = Bestätigung der Kenntnisnahme anfordern, ansonsten 0
- § Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden
- § GUID der Gruppe, die benachrichtigt werden soll
- § Abotyp: 0 = Dokument-Abo; 1 = Anfrage-Abo
- § Aktion, die eine Benachrichtigung auslöst
  - § 2 = Benachrichtigung, wenn ein Dokument angelegt wurde (nur für abonnierte Anfrage)
  - § 3 = Benachrichtigung, wenn Indexdaten geändert wurden (nur für abonnierte Dokumente)
  - § 4 = Benachrichtigung, wenn das Dokument geändert wurde
  - § 27 = Benachrichtigung, wenn das Dokument gelöscht wurde
  - § 39 = Benachrichtigung, wenn ein Standort hinzugefügt wurde
- § Benachrichtigungsart
  - § 0 = Benachrichtigung via internem Kanal (oxmljsc)
  - § 1 = Benachrichtigung via Email; E-Mail Adresse
- § AboGrp: fasst alle Datensätze der verschiedenen Aktionen eines Abonnements zusammen
- § SQL-String für Anfrage- Abonnement

## abn.GetUserList

### Beschreibung:

Der Job liefert eine Liste aller Abonnement-Besitzer (Anfragen/Dokumente) für das angegebene Objekt.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des abonnierten Dokuments oder 0 = abonnierte Anfragen, für das eine Benutzerliste erzeugt werden soll

ObjectType (INT): Typ des Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### Rückgabewerte:

User[1..n] (STRING): Liste der erhaltenen Abonnement-Benutzer

- § Name des Benutzers, der benachrichtigt werden soll
- § Infotext zum Abonnement
- § Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden
- § AboGrp: fasst alle Datensätze der verschiedenen Aktionen eines Abonnements zusammen
- § Aktion, die eine Benachrichtigung auslöst
  - § 2 = Benachrichtigung, wenn ein Dokument angelegt wurde (nur für abonnierte Anfrage)
  - § 3 = Benachrichtigung, wenn Indexdaten geändert wurden (nur für abonnierte Dokumente)

- § 4 = Benachrichtigung, wenn das Dokument geändert wurde
  - § 27 = Benachrichtigung, wenn das Dokument gelöscht wurde
  - § 39 = Benachrichtigung, wenn ein Standort hinzugefügt wurde
  - § Benachrichtigungsart
    - § 0 = Benachrichtigung via internem Kanal (oxmljsc)
    - § 1 = Benachrichtigung via E-Mail; E-Mail Adresse
  - § ID des Abonnements
- abn.NotifyRequestAbo

**Beschreibung:**

Dieser Job wird vom Server-Kernel aufgerufen und überprüft, ob Benutzer über neue Abonnements benachrichtigt werden müssen. Damit der Kernel diesen Job aufruft, muss ein Batch 'RegAbo' eingerichtet sein.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Siehe auch:**

[krm.Batch-Verwaltung](#)

## abn.NotifyAbonnement

**Beschreibung:**

Dieser Jobs sendet die Benachrichtigung an einem Benutzer gemäß der angegebenen Parameter. Wenn Action = 1 (das Dokument wurde gelöscht) wird außer der Benachrichtigung auch das Abonnement für dieses Dokument aus der Datenbank gelöscht.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Dokuments, das geändert oder gelöscht wurde

ObjectType (INT): Typ des Objekts

Action (INT): 1, wenn das Dokument gelöscht wurde; 0, wenn das Dokument geändert wurde

User (STRING): Name des Benutzers, der das Dokument geändert oder gelöscht hat

UserID (INT): ID des Benutzers, der das Dokument geändert oder gelöscht hat

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.Remove

**Beschreibung:**

Dieser Job löscht Abonnements auf Objekte aus der Datenbank. Man kann verschiedene Kombinationen der Parameter verwenden. Wenn z. B. der Parameter Station benutzt wird, dann werden nur die Einträge für diese Station gelöscht.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Dokuments, dessen Abonnement entfernt werden soll

ObjectType (INT): Typ des Objekts

[Benutzer] (STRING): Name des Benutzers, der benachrichtigt werden soll

[UserID] (STRING): Benutzer-GUID des Abo-Admins, wenn System-Abonnements angelegt werden

[GroupID] (STRING): ID der Gruppe, für die das Abo definiert wurde

[Product] (STRING): zum Verfeinern der Suche. Zeichenkette, die die Programm-Instanz des zu entfernenden Abonnements beschreibt.

[Station] (STRING): zum Verfeinern der Suche. Zeichenkette, die die Benutzer-Station des zu entfernenden Abonnements beschreibt.

[Alias] (STRING): Alias des Dokuments, dessen Abonnement entfernt werden soll (max. 255 Zeichen)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.RemoveAbolent

**Beschreibung:**

Dieser Job löscht ein Abonnement-Eintrag aus der Tabelle 'osabonnement' unter der Angabe der Abo-ID oder der AboGrp-ID.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

[AboID] (INT): ID des Abonnements

[AboGrpID] (STRING): ID der Abonnement-Gruppe

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.UpdateReqAboGrp

**Beschreibung:**

Dieser Job ändert für eine Abo-Gruppe das SQL-Statement und den Objekttyp für Anfrage-Abonnements.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

AboGrpID (STRING): Abo-Gruppe

AboRequest (STRING): SQL-Statement für Anfrage-Abonnements

Die Anfrage muss aus Kleinbuchstaben bestehen und mit 'select count(distinct d.id)' oder mit 'select distinct d.id' beginnen, sonst wird sie zurückgewiesen.

Je nach Haupttyp des angefragten Objektes muss 'o.id' für Ordner, 'r.id' für Register oder 'd.id' für Dokumente als Bezeichner verwendet werden.

ObjectType (INT): Typ des Objekts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### [abn.RemoveAllObjAboNotifyFromUser](#)

**Beschreibung:**

Dieser Job entfernt alle, dem Benutzer zugewiesenen Abo-Benachrichtigungen

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### [abn.RemoveObjAboNotifyFromUser](#)

**Beschreibung:**

Dieser Job entfernt eine bestimmte, dem Benutzer zugewiesene Abo-Benachrichtigung

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers

AboSetTime (INT): Zeitpunkt der Benachrichtigung

SetUserID (INT): ID des Benutzers, welcher die Benachrichtigung ausgelöst hat

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### [abn.ConfirmAboRead](#)

**Beschreibung:**

Dieser Job setzt eine Abo-Benachrichtigung, für welche eine Bestätigung angefordert wurde, auf 'bestätigt'

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers, welcher die Benachrichtigung erhalten soll

AboSetTime (INT): Zeitpunkt, an dem der Benutzer Benachrichtigung erhalten hat

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## [abn.RemoveObjRevisitNotifyFromUser](#)

**Beschreibung:**

Dieser Job entfernt eine bestimmte, dem Benutzer zugewiesene Wiedevorlagen-Benachrichtigungen

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers

RevisitTime (INT): Zeitpunkt der Benachrichtigung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## [abn.SetObjRevisitClosed](#)

**Beschreibung:**

Dieser Job setzt eine bestimmte, dem Benutzer zugewiesene Wiedevorlagen-Benachrichtigungen auf 'bearbeitet'

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers

RevisitTime (INT): Zeitpunkt der Anlage der Benachrichtigung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## [abn.SetObjRevisitOpen](#)

**Beschreibung:**

Dieser Job entfernt den Status 'bearbeitet' für eine bestimmte, dem Benutzer zugewiesene Wiedevorlagen-Benachrichtigungen

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers

RevisitTime (INT): Zeitpunkt der Anlage der Benachrichtigung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.ChangeRevisitUser

**Beschreibung:**

Dieser Job weist allen Wiedervorlagen-Benachrichtigungen eines Benutzers einen anderen Benutzer zu.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

NewUserID (INT): ID des neuen Benutzers

OldUserID (INT): ID des bisherigen Benutzers

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.AddRevisit

**Beschreibung:**

Dieser Job setzt ein Objekt für einen bestimmten Benutzer auf 'Wiedervorlage'

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers, welcher die Benachrichtigung erhalten soll

RevisitTime (INT): Zeitpunkt, an dem der Benutzer die Benachrichtigung erhalten soll

InfoText (string): Infotext (max. 225 Zeichen)

RevisitGUID (string): GUID zur Identifizierung der zusammengehörenden Wiedervorlagen (32 Zeichen)

Email (string): Optional anzugebene Email-Adressen getrennt durch ein Semikolon (max. 255 Zeichen)

Confirm (INT): Optional anzugebene Anforderung für eine Passwort gesicherte Bestätigung

Ist „Confirm“ nicht oder mit 0 angegeben, soll keine Passwortabfrage passieren.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.UpdateRevisit

### Beschreibung:

Dieser Job setzt einen neuen Infotext und einen neuen Wiedervorlagezeitpunkt für eine bestehende Wiedervorlage. Optional kann eine neue Email-Adresse angegeben werden.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers, welcher die vorhandene Benachrichtigung erhält

SetUserID (INT): ID des Benutzers, welcher die vorhandene Benachrichtigung erstellt hat

OldRevisitTime (INT): Zeitpunkt, an dem der Benutzer die Benachrichtigung erhalten sollte

SetRevisitTime (INT): Zeitpunkt, an dem die vorhandene Benachrichtigung angelegt wurde

NewRevisitTime (INT): Zeitpunkt, an dem der Benutzer die Benachrichtigung erhalten soll

NewInfoText (string): Neuer Infotext (max. 225 Zeichen)

Email (string): Optional anzugebene Email-Adressen getrennt durch ein Semikolon (max. 255 Zeichen)

Confirm (INT): Optional anzugebene Anforderung für eine Passwort gesicherte Bestätigung

Ist 'Confirm' nicht angegeben, ändert sich diese Einstellung nicht.

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.GetUnreadRevisitCount

### Beschreibung:

Dieser Job ermittelt die Anzahl der gelesenen und ungelesenen Wiedervorlagen eines Benutzers.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

UserID (INT): Benutzer-ID des Benutzers, dessen Wiedervorlagen ermittelt werden sollen.

StartTime (INT): Legt fest, für welchen Zeitraum die Wiedervorlagen ermittelt werden (optional):

§ 0: Alle Wiedervorlagen bis zum aktuellen Zeitpunkt (Default).

§ >0: Angabe des Zeitpunkts in Sekunden ab 01.01.1970 01:00:00 (MEZ), bis zu dem Wiedervorlagen ermittelt werden sollen.

### Hinweis:

Bei der Angabe eines Zeitpunkts werden zum angegebenen Zeitpunkt stets 86399 Sekunden (fast 24 Stunden) hinzugezählt. Das wird gemacht, um die Angabe eines Zeitpunkts zu vereinfachen. Möchte man die Wiedervorlagen ab dem 01.01.2018 ermitteln, dann berechnet man die Sekunden zum Zeitpunkt 01.01.2018 00:00:00 und bekommt dann alle Wiedervorlagen, die bis zum 01.01.2018 23:59:59 fällig werden.

**Rückgabe:**

Read (INT): Anzahl der gelesenen Wiedervorlagen

Unread (INT): Anzahl der ungelesenen Wiedervorlagen

## abn.GetSubscriptions

**Beschreibung:**

Dieser Job liefert die Abonnements für den angemeldeten Benutzer im [DMS Content](#) Format.

**Parameter:**

Flags (INT): Optionen für diesen Job (zur Zeit keine verfügbar)

[XML (BASE64)]: Anfrage im XML-Format (s. [Ausführliche Beschreibung](#)). Hiermit kann innerhalb der abonnierten Objekte eine Auswahl getroffen werden. Die Anfragemöglichkeiten sind dabei auf die Möglichkeiten linearer Anfragen beschränkt.

Es können zusätzlich folgende Parameter zur Formatierung des zurück gelieferten XML Dokumentes gesetzt werden: RequestType, OutputFormat, OutputLanguage, Baseparams, Offset, Pagesize, MaxHits, Rights, DateFormat, Variants, FileInfo, Baseparams. Die Beschreibung dieser Parameter findet sich in der Beschreibung des Jobs [dms.GetResultList](#)

**Rückgabewerte:**

[Count] (INT): Anzahl der Abonnements

[XML] (BASE64): Abonnements im DMSContent XML-Format

Folgende Felder werden dabei zurückgegeben:

Objekt Statusfelder:

Anzahl der Verknüpfungen

Haupttyp (nur für Dokumente)

Anzahl der Seiten (nur für Dokumente)

Archivierungsstatus (nur für Dokumente)

Auscheckstatus (nur für Dokumente)

Eigenschaften des Abonnements

Zeitpunkt, an dem die Wiedervorlage angelegt wurde (**dbname="firstvisit"**)

Abo Typ: (**dbname="osabotype"**). Dieses Feld kann folgende Werte annehmen

OBJECT\_CREATED (@value="2")

DATA\_CHANGED (@value="3")

DOCUMENT\_CHANGED (@value="4")

DOCMOVEDFROMTRAY (@value="38") soll wie OBJECT\_CREATED behandelt werden.

CREATEREFERENCE (@value="39")

Bestätigungsstatus (**dbname="osconfirm"**). Werte:

0=muss nicht bestätigt werden

1=muss bestätigt werden



2=wurde bestätigt

3=muss mit Passwortabfrage bestätigt werden

ID des Benutzer, der die Aktion ausgelöst hat, welche die Abo-Benachrichtigung für den Benutzer gesetzt hat (**dbname**="set\_user\_id")

GUID des Benutzer, der das Abo einrichtet hat (**dbname**="set\_user\_id"). Dieser Wert ist nur dann gesetzt, wenn ein anderer Benutzer das Abo einrichtet hat

Bemerkung zum Abonnement (**dbname**="infotext")

Indexdaten des Objekts

**Beispiel:**

```
<Rowset>
<Columns>
<Column object="Fotos" type="DOCUMENT" name="links" system="1" datatype="INTEGER"
dbname="links" ostyle="9" size="10">OBJECT_LINKS</Column>
<Column object="Fotos" type="DOCUMENT" name="anzahl" system="1" datatype="INTEGER"
dbname="anzahl" ostyle="9" size="10">OBJECT_COUNT</Column>
<Column object="Fotos" type="DOCUMENT" name="flags" system="1" datatype="INTEGER"
dbname="flags" ostyle="9" size="10">OBJECT_FLAGS</Column>
<Column object="Fotos" type="DOCUMENT" name="lockuser" system="1"
datatype="INTEGER" dbname="lockuser" ostyle="9" size="10">OBJECT_LOCKUSER</Column>
<Column object="Fotos" type="DOCUMENT" name="haupttyp" system="1"
datatype="INTEGER" dbname="haupttyp" ostyle="9" size="10">OBJECT_MAIN</Column>
<Column object="Revisit" type="REVISIT" name="firstvisit" system="1" datatype="DATETIME"
dbname="firstvisit" ostyle="9" size="10">REV_FIRSTVISIT</Column>
<Column object="Revisit" type="REVISIT" name="osabotype" system="1" datatype="INTEGER"
dbname="osabotype" ostyle="9" size="10">REV_OSABOTYPE</Column>
<Column object="Revisit" type="REVISIT" name="osconfirm" system="1" datatype="INTEGER"
dbname="osconfirm" ostyle="9" size="10">REV_OSCONFIRM</Column>
<Column object="Revisit" type="REVISIT" name="set_user_id" system="1" datatype="INTEGER"
dbname="set_user_id" ostyle="9" size="10">REV_SET_USER_ID</Column>
<Column object="Revisit" type="REVISIT" name="osuserid" system="1" datatype="TEXT"
dbname="osuserid" ostyle="X" size="32">REV_OSUSERID</Column>
<Column object="Revisit" type="REVISIT" name="infotext" system="1" datatype="TEXT"
dbname="infotext" ostyle="X" size="225">REV_INFOTEXT</Column>
<Column object="Fotos" type="DOCUMENT" name="Bemerkungen" datatype="TEXT"
dbname="feld1" ostyle="X" size="248">Bemerkungen</Column>
</Columns>
<Rows>
<Row id="73543">
<Value>0</Value>
<Value>1</Value>
<Value value="2">NOT_ARCHIVABLE</Value>
<Value value="0">UNLOCKED</Value>
<Value value="3">COLOR</Value>
<Value value="1089804789">14.07.2004 13:33:09</Value>
<Value value="2">OBJECT_CREATED</Value>
<Value>1</Value>
<Value value="53">LIEBE</Value>
<Value value="" />
<Value>Neue Fotos</Value>
<Value>Vorderansicht</Value>
</Row>
</Rows>
</Rowset>
```

## abn.GetRevisits

### Beschreibung:

Dieser Job liefert die Wiedervorlagen für den angemeldeten Benutzer im [DMS Content](#) Format.

Hinweis:

Zur Zeit werden keine Mappen berücksichtigt

### Parameter:

Flags (INT): Optionen für diesen Job

§ 4096 = das XML-Dokument wird UTF-8 kodiert, ansonsten UTF-16

StartTime (STRING): optionaler Zeitstempel für den Zeitpunkt, bis zu dem die Wiedervorlagen zurückgegeben werden sollen. Format: DD.MM.YYYY HH:MM.SS, wobei die Zeitangabe entfallen kann.

Spezialwert 0: (default) es werden alle aktuell vorliegenden, nicht abgehakten Wiedervorlagen geliefert

Zusätzlich können folgende Parameter zur Formatierung des zurück gelieferten XML Dokumentes gesetzt werden: RequestType, OutputFormat, Baseparams, Offset, Pagesize, MaxHits, Rights, DateFormat, Variants, FileInfo, Baseparams. Die Beschreibung dieser Parameter findet sich in der Beschreibung des Jobs [dms.GetResultList](#)

### Rückgabewerte:

[Count] (INT): Anzahl der Wiedervorlagen

[XML] (BASE64): Wiedervorlagen im DMSContent XML-Format

Folgende Felder werden dabei zurückgegeben:

§ Objekt Statusfelder:

- § Anzahl der Verknüpfungen
- § Haupttyp (nur für Dokumente)
- § Anzahl der Seiten (nur für Dokumente)
- § Archivierungsstatus (nur für Dokumente)
- § Auscheckstatus (nur für Dokumente)

§ Eigenschaften der Wiedervorlage:

- § Zeitpunkt, an dem die Wiedervorlage angelegt wurde (dbname="set\_time")
- § Zeitpunkt, ab dem die Wiedervorlage dem Benutzer vorgelegt wird (dbname="firstvisit")
- § Zeitpunkt, an dem der Benutzer die Wiedervorlage zur Kenntnis genommen hat (dbname="lastvisit"). Siehe auch 'abn.SetObjRevisitClosed' oder 'abn.SetObjRevisitOpen'
- § Bemerkung zur Wiedervorlage (dbname="infotext")
- § Bestätigung der Wiedervorlage (dbname="osconfirm")
- § (0=keine Bestätigung durch ein Passwort erwartet, 1 = Bestätigung nur durch Eingabe des Passwortes möglich)
- § Benutzer, der die Wiedervorlage eingerichtet hat (dbname="set\_user\_id")

§ Indexdaten des Objekts

**Beispiel:**

```

<Rowset>
<Columns>
<Column object="Graustufenbild" type="DOCUMENT" name="links" system="1"
datatype="INTEGER" dbname="links" ostyle="9" size="10">OBJECT_LINKS</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="anzahl" system="1"
datatype="INTEGER" dbname="anzahl" ostyle="9" size="10">OBJECT_COUNT</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="flags" system="1"
datatype="INTEGER" dbname="flags" ostyle="9" size="10">OBJECT_FLAGS</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="lockuser" system="1"
datatype="INTEGER" dbname="lockuser" ostyle="9" size="10">OBJECT_LOCKUSER</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="haupttyp" system="1"
datatype="INTEGER" dbname="haupttyp" ostyle="9" size="10">OBJECT_MAIN</Column>
<Column object="Revisit" type="REVISIT" name="set_time" system="1" datatype="DATETIME"
dbname="set_time" ostyle="9" size="10">REV_SET_TIME</Column>
<Column object="Revisit" type="REVISIT" name="lastvisit" system="1" datatype="DATETIME"
dbname="lastvisit" ostyle="9" size="10">REV_LASTVISIT</Column>
<Column object="Revisit" type="REVISIT" name="firstvisit" system="1" datatype="DATETIME"
dbname="firstvisit" ostyle="9" size="10">REV_FIRSTVISIT</Column>
<Column object="Revisit" type="REVISIT" name="osconfirm" system="1" datatype="INTEGER"
dbname="osconfirm" ostyle="9" size="10">REV_OSCONFIRM</Column>
<Column object="Revisit" type="REVISIT" name="set_user_id" system="1" datatype="INTEGER"
dbname="set_user_id" ostyle="9" size="10">REV_SET_USER_ID</Column>
<Column object="Revisit" type="REVISIT" name="infotext" system="1" datatype="TEXT"
dbname="infotext" ostyle="X" size="225">REV_INFOTEXT</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="Dokumentart" datatype="TEXT"
dbname="feld1" ostyle="X" size="30">Dokumentart</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="Datum" datatype="DATE"
dbname="datum1" ostyle="D" size="10">Datum</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="Autor" datatype="TEXT"
dbname="feld2" ostyle="X" size="50">Autor</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="Quelle" datatype="TEXT"
dbname="feld3" ostyle="X" size="150">Quelle</Column>
<Column object="Graustufenbild" type="DOCUMENT" name="Inhalt" datatype="TEXT"
dbname="feld4" ostyle="X" size="150">Inhalt</Column>
</Columns>
<Rows>
<Row id="415">
<Value>0</Value>
<Value>1</Value>
<Value value="2">NOT_ARCHIVABLE</Value>
<Value value="">UNLOCKED</Value>
<Value value="1">GRAYSCALE</Value>
<Value value="1089723507">13.07.2004 14:58:27</Value>
<Value value="0" />
<Value value="1089723600">13.07.2004 15:00:00</Value>
<Value />
<Value value="53">LIEBE</Value>
<Value>Bitte anschauen</Value>
<Value>Zeichnung</Value>
<Value>04.09.2002</Value>
<Value>Liebe</Value>
<Value>Höhle</Value>
<Value>Rind</Value>
</Row>
<Row id="416">
<Value>0</Value>
<Value>0</Value>
<Value value="8">NO_PAGES</Value>

```

```

<Value value="">UNLOCKED</Value>
<Value value="1">GRAYSCALE</Value>
<Value value="1089727995">13.07.2004 16:13:15</Value>
<Value value="0" />
<Value value="1089728100">13.07.2004 16:15:00</Value>
<Value />
<Value value="53">LIEBE</Value>
<Value>Bild nachliefern</Value>
<Value>Ärgernisse</Value>
<Value>05.09.2002</Value>
<Value>Admin</Value>
<Value>unbekannt</Value>
<Value>Fledermaus</Value>
</Row>
</Rows>
</Rowset>

```

## abn.SetOsInformed

### Beschreibung:

Dieser Job setzt ein Objekt in der Abo-Liste als 'gelesen'

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers, welcher die vorhandene Benachrichtigung erhält

AboSetTime (INT): Zeitstempel, wann das Abo angelegt wurde

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.ResetOsInformed

### Beschreibung:

Dieser Job setzt ein Objekt in der Abo-Liste als 'ungelesen'

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Objekts

ObjectType (INT): Typ des Objekts

UserID (INT): ID des Benutzers, welcher die vorhandene Benachrichtigung erhält

AboSetTime (INT): Zeitstempel, wann das Abo angelegt wurde

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## abn.GetRecentObjects

### Beschreibung:

Dieser Job ermittelt die zuletzt von angemeldetem Benutzer bearbeiteten DMS-Objekte.

#### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Count (INT): Anzahl der zu ermittelnden DMS-Objekte

HistActIDList (String): Komma-Separierte Liste der HistActID

#### Rückgabe:

Recent (String): Komma und Semikolon-Separierte Liste der ermittelten DMS-Objekte.

Parameter (In):

Count (Erforderlich)	DWORD	Anzahl der zu ermittelnden zuletzt bearbeiteten DMS-Objekte. Dieser Wert ist auf einen Bereich von 5 bis 99 begrenzt. Wird ein Wert ausserhalb dieses Bereich angegeben, wird die näherliegende Bereichsgrenze angenommen.  Bsp.: Count=0 -> Count=5 Count=1000 -> Count=99
HistActIDList (Optional)	String	Eine Komma-Separierte Liste von osHistAct-Aktionen, welche zu Bestimmung der zuletzt bearbeiteten DMS-Objekte herangezogen werden sollen.  Ist dieser Parameter nicht angegeben, sind das die Aktionen „2,3,4“ 2 - Objekt angelegt: Das angegebene Objekt wurde durch Funktionen des Clients oder durch einen Import erzeugt. 3 - Indexdaten geändert: Die Indexdaten des Objektes oder dessen Status wurden durch Funktionen des Clients oder durch ein Update mittels des Imports geändert. 4 - Dokument geändert: Das Dokument wurde durch Funktionen des Clients oder durch ein Update mittels des Imports geändert.

Parameter (Out):

Recent	String	Komma und Semikolon-Separierte Liste der ermittelten DMS-Objekte. Form: ObjId1,ObjType1,Aktion1,Zeit1;...; ObjId(n),ObjType(n),Aktion(n),Zeit(n) Bsp.: 12,0,2,12389147391; 13,1,2,12389137474; 14,0,3,12389127897 Die Reihenfolge ist nach Zeit absteigend sortiert. Also jüngste zuerst.
--------	--------	--

## ADO-Datenbank-Engine (Namespace ado)

Über die ADO-Datenbank-Engine wird die Möglichkeit zur Verfügung gestellt, innerhalb der 3-Tier-Architektur Zugriff auf die Datenbank zu erhalten. Dies geschieht dadurch, dass der Client seine SQL-Anfrage an den Applikationsserver richtet, dieser die Abfrage ausführt und das Ergebnis als ADO-Recordset (Active Data Object) in XML-Repräsentation an den Client sendet.

Diese Funktionalität wird insbesondere für SQL-Anfragen verwendet, um über die normalen Recherchemöglichkeiten des Clients hinaus Anfragen formulieren zu können.

Über diese Schnittstelle können nur dann auch Datenmanipulationen (INSERT, UPDATE, DELETE) durchgeführt werden, wenn dies in der Registry des Applikationsservers freigeschaltet wird.

## ado.ExecuteSQL

### Beschreibung:

Dieser Job führt ein SQL-Statement auf der Datenbank aus. Als Resultat wird eine XML-Datei mit der Antwort auf die Abfrage erzeugt und ins ostemp-Verzeichnis geschrieben.

### Parameter:

Flags (LONG): wird z. Z. nicht unterstützt -> 0 übergeben

CursorType (int): -1 = Cursortyp wie in der Registry angegeben (default); 0,1,2,3 = entsprechend der ADO Konstanten für Cursortypen. Andere Werte führen zu Fehlermeldungen.

Command (STRING): SQL-Befehl zur Ausführung

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### Rückgabewerte:

Dateiliste: Pfad und Name der Ergebnisdatei im XML-Format

### Beispiel:

zurückgelieferte XML-Datei für 'SELECT \* FROM osorganisations'

```
<xml xmlns:s="uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882"
xmlns:dt="uuid:C2F41010-65B3-11d1-A29F-00AA00C14882"
xmlns:rs="urn:schemas-microsoft-com:rowset" xmlns:z="#RowsetSchema">
<s:Schema id="RowsetSchema">
<s:ElementType name="row" content="eltOnly" rs:CommandTimeout="30"
rs:updatable="true">
<s:AttributeType name="id" rs:number="1" rs:nullable="true"
rs:writeunknown="true" rs:basecatalog="as_test"
rs:basetable="osorganisations" rs:basecolumn="id">
<s:datatype dt:type="string" rs:dbtype="str" dt:maxLength="32"/>
</s:AttributeType>
<s:AttributeType name="name" rs:number="2" rs:nullable="true"
rs:writeunknown="true" rs:basecatalog="as_test"
rs:basetable="osorganisations" rs:basecolumn="name">
<s:datatype dt:type="string" rs:dbtype="str" dt:maxLength="255"/>
</s:AttributeType>
<s:AttributeType name="layout" rs:number="3" rs:nullable="true"
rs:maydefer="true" rs:writeunknown="true"
rs:basecatalog="as_test" rs:basetable="osorganisations"
rs:basecolumn="layout">
<s:datatype dt:type="bin.hex" dt:maxLength="2147483647"
rs:long="true"/>
</s:AttributeType>
<s:AttributeType name="active" rs:number="4" rs:nullable="true"
rs:writeunknown="true" rs:basecatalog="as_test"
rs:basetable="osorganisations" rs:basecolumn="active">
<s:datatype dt:type="int" dt:maxLength="4" rs:precision="10"
rs:fixedlength="true"/>
</s:AttributeType>
<s:extends type="rs:rowbase"/>
</s:ElementType>
```

```

</s:Schema>
<rs:data>
<z:row id="45808CE977334AB88C5A8EFF467689A8" name="Test" active="1"/>
</rs:data>
</xml>

```

## Convert-Engine (Namespace cnv)

Diese Engine stellt Jobs für die Konvertierung von Bilddateien zur Verfügung.

§ cnv.ConvertDocument

§ cnv.CreateSlide

§ cnv.AddAnnotations

§ cnv.GetIcons

§ cnv.GetExifData

§ cnv.GetPageCount

§ cnv.GetPictureInfos

§ cnv.GetRendition

### cnv.ConvertDocument

#### Beschreibung:

Dieser Job konvertiert eine oder mehrere Dokumentendateien des angegebenen Formats in eine oder mehrere PDF- oder TIFF- Dateien. Entsprechend der jeweiligen Konfigurationen sind darüber hinaus auch beliebige Ein- und Ausgabeformate möglich.

Dabei sind z. B. folgende Klassen von Transformationen möglich:

Bitmapformat (JPG, TIF) -> PDF

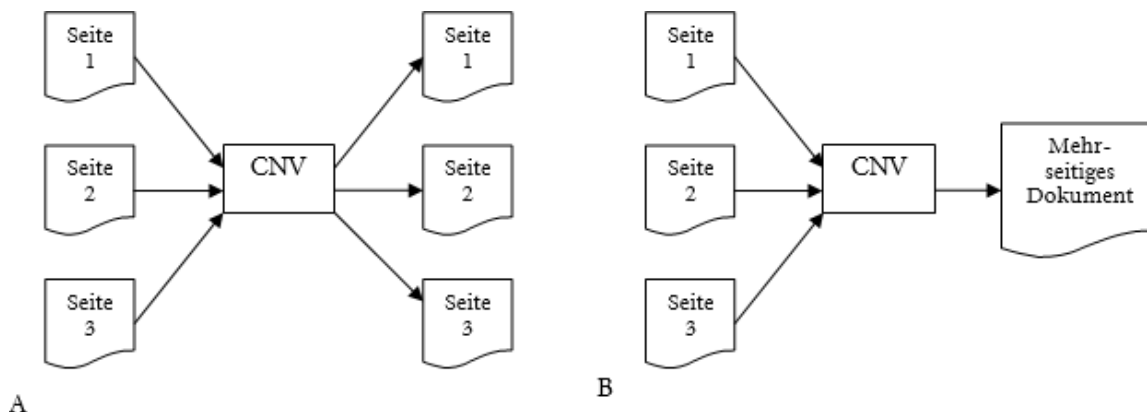
Singlepage TIFF -> Multipage TIFF

ASCII-COLD (asc) -> TIFF / PDF

XSL:FO (.fo, .xml) -> PDF

Office Dokumente (ps, doc, xls, ppt, txt, rtf, pdf, ima) -> PDF

Für Bitmapformate und ASCII COLD Dokumente lässt sich dabei über ein Flag steuern, ob für jede Eingabedatei eine eigene Ausgabedatei erzeugt werden soll (A) oder ob die einzelnen Seiten in einem Dokument zusammengefasst werden (B):





**Parameter:**

Flags (INT): Optionen für den Job

- § 0 = es wird für jede Eingabe-Datei eine Ausgabedatei erstellt
- § 1 = alle Eingabe-Dateien werden in eine Ausgabedatei geschrieben
- § 2 = es wird für jede Eingabe-Datei eine Ausgabedatei erstellt, die vom Server erstellten Temporären Dateien werden nicht gelöscht
- § 3 = alle Eingabe-Dateien werden in eine Ausgabedatei geschrieben, die vom Server erstellten Temporären Dateien werden nicht gelöscht

SourceFormat (STRING): Format der Ausgangsdatei

DestinationFormat (STRING): Format, in das konvertiert werden soll (s.o.).

- Anm.: Die Formatangaben werden in Kleinschreibweise erwartet ('pdf' statt 'PDF').

Timeout (INT): (Optional) Maximale Zeit in Millisekunden, die eine Konvertierung über externe Programme dauern darf. Wird berücksichtigt bei XSL:FO und Office Format Konvertierungen.

AddAnnotations (INT): 1 = Öffentliche Folien werden eingebrannt.

ConvertEqualFormat (INT): (Optional, Default ist 0) Wenn auf 1 gesetzt, dann wird versucht eine Konvertierung durchzuführen, selbst wenn das Quellformat und Zielformat identisch ist, z. B. um ein PDF in ein PDF/A zu konvertieren.

Watermark (INT): (Optional, Default ist 0) Wenn auf 1 gesetzt, dann wird ein erzeugtes PDF Dokument mit Kopf- und Fußzeilen versehen. Aussehen und Inhalt wird über den Administrator konfiguriert. Siehe enaio® Administratorhandbuch -> Registerkarte Druckkennzeichnung

Eine zusätzliche Möglichkeit besteht darin, die Funktionalität der erweiterten Wasserzeichen zu verwenden. Fordern Sie dazu die Dokumentation der Bibliothek oxsvrspt.dll an.

dwObjectID (INT): (Optional) Dokument Objekt ID für die Benutzung in Wasserzeichen. Nur relevant, wenn die Option Watermark auf 1 gesetzt ist und die Druck Wasserzeichen so konfiguriert sind, dass die Dokumenten ID mit ausgegeben werden soll.

ProtectPDF (INT): (Optional, Default ist 0) Ein erzeugtes PDF Dokument wird geschützt. Es dann nicht mehr möglich, das ein Betrachter das Dokument ausdruckt oder Textpassagen kopiert.

ObjectID (INT): (Pflichtparameter) ID des Dokuments

ObjectType (INT): (Pflichtparameter) Typ des Objekts

oder

Dateiliste: Pfad und Name der Dateien, die konvertiert werden sollen.

Werden ObjectID und ObjectType übergeben, erfolgt die Konvertierung über OS RenditionPlus (\_\_\_ren.bat).

Digest (STRING): (Optional) Hashwert des Dokuments

Optional kann dann als Zielformat 'SLIDE' zur Vorschaugenerierung übergeben werden. Mit 'Height' und 'Width' kann die Größe der zu generierenden Vorschau angegeben werden.

Statt 'SLIDE' ist dann auch für eine Texterkennung als Zielformat 'TXT' zulässig.

**Rückgabewerte:**

Dateiliste: Pfad und Name der konvertierten Datei(en)



**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

**Siehe auch:**

[cnv.CreateSlide](#)

**Verwendung von XSL:FO**

- § Für die Konvertierung von FO (Formatting Objects) Dateien muss der Apache FOP Prozessor installiert sein. Dieser benötigt wiederum eine Java Laufzeitumgebung 1.6 oder höher. Der Pfad zur FOP Batch-Datei (Windows) bzw. zum FOP Shellskript (Linux) muss z. B. über den Enterprisemanager in der Registry hinterlegt werden. Desweiteren kann eine Zeitspanne (Timeout) definiert werden, nach welcher der Konvertierungsprozess abgebrochen wird.
- § Wird dem FOP ein XML Dokument übergeben, muss zusätzlich ein XSLT Dokument als **2. Datei** in der Eingabedateiliste übergeben werden, mit dem das XML in ein XSL:FO Dokument überführt werden kann.
- § Sollen Bilder in das resultierende PDF eingebunden werden, so müssen diese in der XSL:FO Datei über eine für den Applikationsserver zugängliche URL referenziert werden.
- § Es kann nur ein FO bzw. XML Dokument pro Job verarbeitet werden.

Beispiel für einen cnv.ConvertDocument Aufruf:

SourceFormat: XML

DestinationFormat: PDF

Flags=0

**1. Eingabedatei: XML Datei mit beliebigen Eingabeinformationen.**

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="example.fo"?>
<persons>
<person age="42">Max Mustermann</person>
<person age="37">Denise Musterfrau</person>
</persons>
```

**2. Eingabedatei: XSL Datei zur Konvertierung in eine XML:FO Datei.**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format" version="1.0">
<xsl:template match="persons">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
<fo:layout-master-set>
<fo:simple-page-master page-master-name="one" page-height="29cm" page-width="21cm"
margin-left="2cm" margin-right="2cm">
<fo:region-body margin-top="50pt" margin-bottom="50pt" />
</fo:simple-page-master>
</fo:layout-master-set>
<fo:block text-align="center" font-size="24pt" font-weight="bold" line-
height="28pt" space-after="10mm">
<xsl:apply-templates/>
</fo:block>
</fo:root>
</xsl:template>
<xsl:template match="person">
...
</xsl:template>
</xsl:stylesheet>
```

## cnv.CreateSlide

### Beschreibung:

Dieser Job konvertiert Dateien und Dokumente der Formate JPG oder TIF in eine DIA-Datei mit dem gleichen Format der Eingabedatei. DIA-Dateien sind hoch komprimiert und dienen als Vorschau auf ein Dokument.

### Parameter:

Flags (INT): Optionen für den Job

§ 0 = es wird eine DIA-Ausgabedatei erstellt

§ 2 = es wird eine DIA-Ausgabedatei erstellt, die vom Server erstellten temporären Dateien werden nicht gelöscht

(Pflichtparameter)

ObjectID (INT): ID des Dokuments

ObjectType (INT): Typ des Objekts

oder

Dateiliste: Pfad und Name der Datei im Format JPG oder TIF, die konvertiert werden sollen.

Page (STRING): (Optional) Seitennummer des Dokuments (Dateiliste), für welche die Daten ermittelt werden sollen. Default '1', 'All' für alle Seiten.

### Rückgabewerte:

Dateiliste: Pfad und Name der konvertierten DIA-Datei(en) mit Format entsprechend der Eingabedatei(en)

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

### Siehe auch:

[cnv.ConvertDocument](#)

## cnv.AddAnnotations

### Beschreibung:

Dieser Job brennt Anmerkungen (Folien) in eine oder mehrere Bilddateien ein.

Hinweis:

Der Job berücksichtigt, die Zugriffsrechte auf die Annotations. D. h. es wird z. B. nur die persönliche Folie des angemeldeten Benutzers angezeigt.

### Parameter:

Flags (INT):

§ 0 = Die Eingabedateien werden gelöscht. Von unveränderten Eingabedateien wird eine temporäre Kopie erstellt.

§ 2 = Eingabedateien werden nicht gelöscht. Unveränderte Eingabedateien werden in die Ausgabeliste zurückgeschrieben.

AnnotationID (INT): (Optional) Bestimmte ID einer Annotation, die in die Eingabedatei(en) eingebrannt werden soll.

dwObjectID (INT): (Optional) ID eines Objekts, die in die Eingabedatei(en) eingebrannt werden soll. AnnotationID kann, dwObjectID muss hierfür übergeben werden.

Dateiliste: Pfade und Namen der Dateien, die mit Anmerkungen versehen werden sollen.

**Rückgabewerte:**

nProcessed: Anzahl der Bilddateien, die mit einer Anmerkung versehen wurden.

sAnnotationIDs: Kommaseparierte Liste der AnnotationIDs, die eingebrannt wurden.

Dateiliste: Pfad und Name der Bilddatei, die mit Anmerkungen versehen sind, wenn für diese Eingabedatei Anmerkungen vorhanden waren. Ansonsten wird die Eingabedatei unverändert zurückgegeben (siehe Flags). Das Ausgabeformat entspricht dem der jeweiligen Eingabedatei.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## cnv.GetIcons

**Beschreibung:**

Dieser Job liefert Icons im GIF Format zurück. Damit ist es möglich benutzerdefinierte Icons, wie im Archivbereich und in den Trefferlisten verwendet, auszulesen.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

sIconIds (String): Kommaseparierte Liste der Icon-IDs.

**Rückgabewerte:**

sIconIds (String): Kommaseparierte Liste der Icon-IDs in der gleichen Reihenfolge wie die Ausgabedateien.

Dateiliste: Pfad und Name der Icons im GIF Format.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## cnv.GetExifData

**Beschreibung:**

Dieser Job ermittelt die EXIF-Daten aus den Bilddateien eines Dokuments.

Informationen (EXIF-, Dicom-, allgemeine Daten) werden nur für EXIF-, JPEG-, TIF- und Dicom-Dateien ermittelt.

**Parameter:**

ObjectID (INT): ID des Dokuments

ObjectType (INT): Typ des Objekts

oder

Dateiliste: Pfade und Namen der Dateien, deren Daten ermittelt werden sollen.

Flags (INT): (Pflichtparameter) 1 = Übergebene Dateiliste nicht löschen, 0 = löschen für Aufrufe von Clientseite.

Page (STRING): (Optional) Seitennummer des Dokuments (Dateiliste), für welche die Daten ermittelt werden sollen. Default '1', 'All' für alle Seiten.

**Rückgabewerte:**

Enthaltene Informationen.

Die Informationen werden in der Form 'Leadtool-Name(Seitennummer) Wert' zurückgegeben.

z. B.:

CMNT\_SZMAKE(1) "Panasonic"

CMNT\_SZMODEL(1) "DMC-TZ10"

...

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## cnv.GetPageCount

**Beschreibung:**

Dieser Job ermittelt die Gesamtseitenzahl aus den Bilddateien eines Dokuments.

Eine Ermittlung erfolgt nur für Bild- und PDF-Dokumente. Wenn aus allen Dateien aus der Liste oder des Dokuments die Gesamtseitenzahl bestimmt werden konnte, werden 'FileCount' und 'PageCount' zurückgegeben.

**Parameter:**

(Pflichtparameter)

ObjectID (INT): ID des Dokuments

ObjectType (INT): Typ des Objekts

oder

Dateiliste: Pfade und Namen der Dateien, deren Gesamtseitenzahl ermittelt werden soll.

Flags (INT): (Pflichtparameter) 1 = Übergebene Dateiliste nicht löschen, 0 = Löschen für Aufrufe von Clientseite.

**Rückgabewerte:**

FileCount (INT): Anzahl der Dateien des Dokuments (Dateiliste)

PageCount (INT): Anzahl der Dateien des Dokuments (Dateiliste)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## cnv.GetPictureInfos

**Beschreibung:**

Dieser Job ermittelt Bildinformationen aus den Dateien eines Dokuments.

Eine Ermittlung erfolgt nur für Bild- und PDF-Dokumente.

**Parameter:**

(Pflichtparameter)

ObjectID (INT): ID des Dokuments

ObjectType (INT): Typ des Objekts

oder

Dateiliste: Pfade und Namen der Dateien, deren Gesamtseitenzahl ermittelt werden soll.

Flags (INT): (Pflichtparameter) 1 = Übergebene Dateiliste nicht löschen, 0 = Löschen für Aufrufe von Clientseite.

Page (STRING): (Optional) Seitennummer des Dokuments (Dateiliste), für welche die Daten ermittelt werden sollen. Default '1', 'All' für alle Seiten.

### **Rückgabewerte:**

Bildinformationen, die Informationen werden in der Form 'Infoname(Seitennummer) Wert' zurückgegeben.

Folgende Informationen (INT) werden für Bilddokumente zurückgegeben:

"InfoBits(1)"

"InfoCompress(1)"

"InfoFormat(1)"

"InfoIFD(1)"

"InfoLayers(1)"

"InfoSizeDisk(1)"

"InfoSizeMem(1)"

"InfoViewPerspective(1)"

"InfoHeight(1)"

"InfoWidth(1)"

"InfoFileType(1)"

"InfoFileSuffix(1)"

Mit den entsprechenden Werten in Klammern wird die Seitennummer angegeben.

Für PDF-Dateien wird Folgendes zurückgegeben:

"InfoFormat(1)"

"InfoFileType(1)"

"InfoFileSuffix(1)"

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## **cnv.GetRendition**

### **Beschreibung:**

Dieser Job führt mittels eines Aufrufs über die Datei \_\_\_\_ren.bat eine Rendition für ein Dokument durch.

### **Parameter:**

(Pflichtparameter)

ObjectID (INT): ID des Dokuments

ObjectType (INT): Typ des Objekts

Flags (INT): (Pflichtparameter) 1 = Übergebene Dateiliste nicht löschen, 0 = Löschen für Aufrufe von Clientseite.

DestinationFormat (STRING): Format, in das die Datei konvertiert werden soll. Siehe auch `cnv.ConvertDocument`

Optional kann dann als Zielformat 'SLIDE' zur Vorschaugenerierung übergeben werden. Mit 'Height' und 'Width' kann die Größe der zu generierenden Vorschau angegeben werden.

Statt 'SLIDE' ist für eine Texterkennung auch als Zielformat 'TXT' zulässig.

Digest (STRING): (Optional) Hashwert des Dokuments

Watermark (INT): (Optional, Default ist 0) Wenn auf 1 gesetzt, dann wird ein erzeugtes PDF-Dokument mit Kopf- und Fußzeilen versehen. Aussehen und Inhalt wird über den Administrator konfiguriert. Siehe enaio® Administratorhandbuch, Kapitel: Registerkarte 'Druckkennzeichnung'

AddAnnotations (INT): 1 = Öffentliche Folien werden eingebrannt.

ProtectPDF (INT): (Optional, Default ist 0) Ein erzeugtes PDF-Dokument wird geschützt. Das Dokument kann dann weder ausgedruckt werden, noch können Textpassagen kopiert werden.

SynchData (INT): (Optional, Default ist 0)

### **Rückgabewerte:**

Dateiliste: Pfad und Name der konvertierten Datei(en)

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten Fehlercode

## DMS-Engine (Namespace dms)

Der DMS Executor umfasst Jobs zum Anfragen und Bearbeiten von Indexdaten, DMS Objekten, Relationen und Mappen unter Berücksichtigung des Sicherheitssystems. Weiterhin existieren Jobs um das Sicherheitssystem auf Objektebene zu verwalten.

### Bereiche

§ [XML Import](#)

§ [XML Export \(Recherche\)](#)

§ [Sicherheitssystem](#)

§ [Relationen und Relationstexte](#)

§ [Mappen \(Portfolios\)](#)

§ [Benutzerbezogene Daten](#)

§ [Sonstige Jobs](#)

### XML Import

§ [DMS.CheckInDocument](#)

- § [DMS.CheckOutDocument](#)
- § [DMS.GetXMLJobOptions](#)
- § [DMS.UndoCheckOutDocument](#)
- § [DMS.XMLDelete](#)
- § [DMS.XMLInsert](#)
- § [DMS.XMLMove](#)
- § [DMS.XMLCopy](#)
- § [DMS.XMLUpdate](#)
- § [DMS.XMLImport](#)
- § [DMS.XMLUnknownToKnown](#)

### Ausführliche Beschreibung

Alle XML-Importjobs entsprechen alle demselben Schema und können daher zum großen Teil allgemein behandelt werden. Sie haben alle dieselben Eingabeparameter und im Wesentlichen dieselben Rückgabeparameter. So gibt es beispielsweise den Job 'XMLInsert', der die Übergabeparameter 'Flags', 'Options' und 'XML' besitzt.

Durch die ['Flags'](#) wird das allgemeine Verhalten des Jobs (z. B. Fehlerlistengenerierung oder XML-Validierung) gesteuert. Über den Parameter ['Options'](#) können bestimmte Überprüfungen (z. B. Schlüsselwortprüfung) an- bzw. ausgeschaltet werden. Der Parameter ['XML'](#) enthält die Daten, die das einzufügende Objekt beschreiben. Der Parameter ['JobUserGUID'](#) ermöglicht es, den Benutzerkontext für diesen Job zu ändern.

#### Beispiel:

Das Beispiel zeigt ein XML, mit dem ein Dokument vom Typ 'Dokumenttyp-Name', der im Schrank 'Schrang-Name' liegt, mit dem Job 'XMLInsert' eingefügt wird. Dabei wird in den Indexdaten des neuen Dokumentes das Feld mit dem Namen 'Feld-Name' der Wert 'Mein Wert' eingefügt.

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive name="Schrang-Name">
<ObjectType name="Dokumenttyp-Name" type="DOCUMENT">
<Object>
<Fields>
<Field name="Feld-Name">Mein Wert</Field>
</Fields>
</Object>
</ObjectType>
</Archive>
</DMSData>
```

Als Rückgabeparameter gibt es einerseits die Objekt-ID des neu eingefügten Objektes, des Weiteren gibt es optional (siehe [Flags](#)) einen [XML-Text](#), der die aufgetretenen Fehler beschreibt. Jeder Job hat über die Parameter hinaus einen [Rückgabewert](#). Dieser ist im Erfolgsfall '0' oder qualifiziert einen eventuell aufgetretenen Fehler grob.

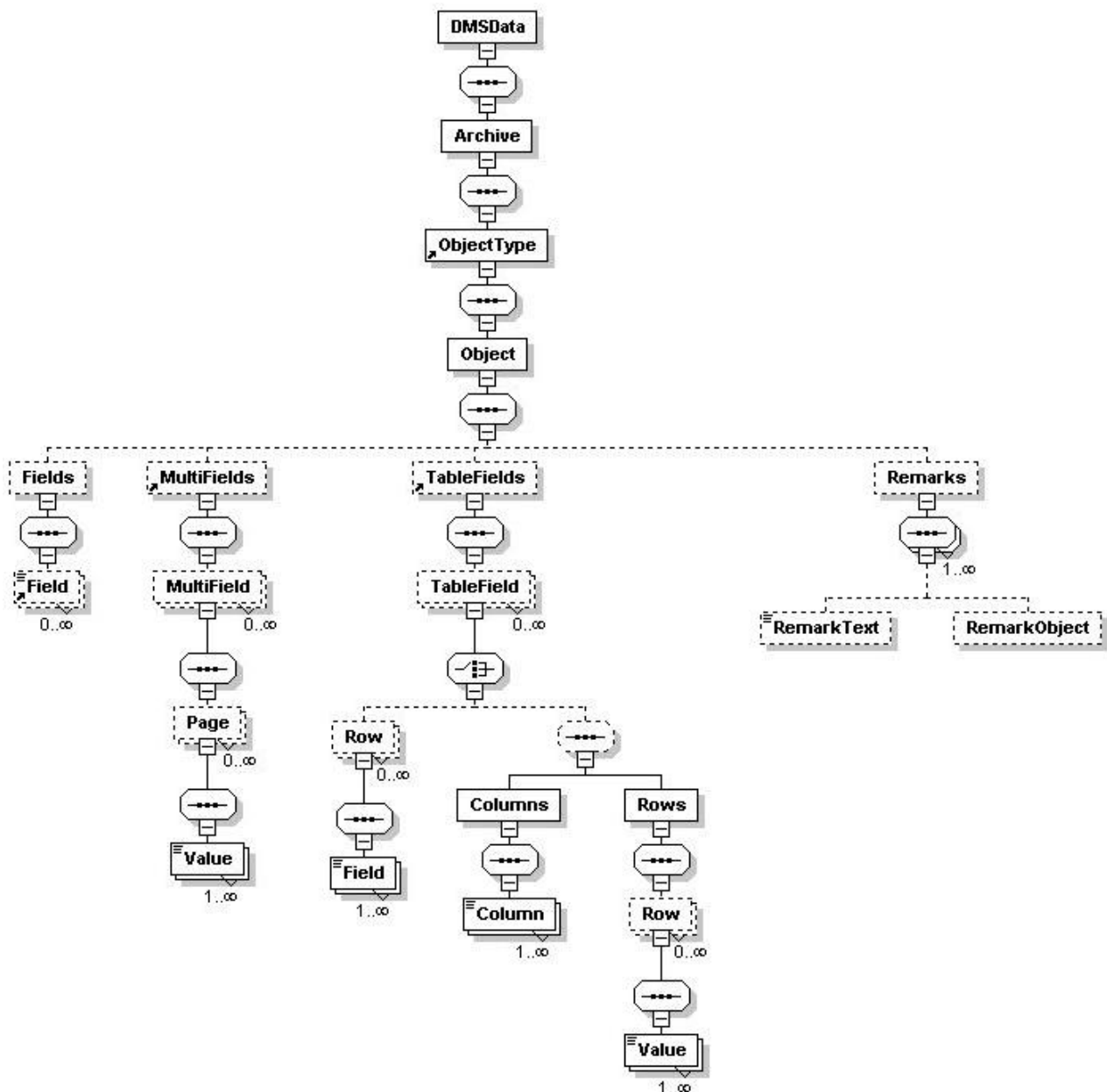
#### Der Parameter 'XML'

Über den Parameter 'XML' wird die Objektbeschreibung im XML-Format übergeben. Das XML muss Base64 kodiert werden, da der Server ansonsten Probleme hat, Sonderzeichen zu übertragen. Auch UTF-16/UCS-4-Formate könnten ansonsten nicht übertragen werden. Zu der XML-Struktur existiert ein Schema, gegen das das XML validiert werden kann. Um dieses Schema einsehen zu können,

existiert der Job '[GetXMLSchema](#)'. Diesem übergibt man im Parameter Schema den Wert 'DMSData' und erhält in der Rückgabe-Dateiliste das Import-Schema. Werden Attribute gesetzt, sollen sie jedoch nicht vom Job ausgewertet werden, so sind Text-Attribute mit dem Leer String '' zu setzen, numerische Attribute sind mit dem Wert '-1' vorzubelegen.

### Das Import-Schema

Der hierarchische Aufbau der XML-Struktur entspricht der folgenden Grafik.



### Beschreibung des XML-Schemas

**DMSData:** Dieser Tag stellt das Root Tag des XML dar. Über das optionale Attribut 'query\_language' lässt sich die Sprache für die DMS Bezeichner angeben. Als Wert wird der entsprechende [Sprachcode](#) erwartet, z. B. '7' für Deutsch oder '9' für Englisch. Wird diese Attribut nicht angegeben oder auf '0' gesetzt, wird als Anfragesprache die Defaultsprache angenommen.



**Archive:** Über die Attribute dieser Tags wird der Schrank identifiziert, in dem der zu behandelnde Objekttyp liegt oder eingefügt werden soll. Werden keine möglichen Attribute angegeben, wird versucht, anhand des zu importierenden Objekttypen den entsprechenden Schrank zu ermitteln. Dieses geht jedoch zu Lasten der Performance und sollte daher nach Möglichkeit vermieden werden. Der Schrank kann anhand seiner Attribute `id` (ID des Objekttyps), `name` (Name des Schranks), `internal_name` (interner Schrankname) oder `osguid` (GUID des Schranks) identifiziert werden.

**ObjectType:** Innerhalb dieser Tags wird der Objekttyp des zu behandelnden Objektes festgelegt. In den Attributen muss über den Typ ('type') festgelegt werden, ob es sich um einen Ordner, Register oder Dokument handelt (Mögliche Werte sind 'FOLDER', 'REGISTER' und 'DOCUMENT'). Der Objekttyp selbst kann anhand seiner Attribute: `name` (Name), `internal_name` (interner Name), `id` (Objekttyp ID), `table` (Tabellenname), `osguid` (GUID) oder einer gültigen Kombination aus `maintype` und `cotype` (Haupt- und Untertyp) ermittelt werden.

**Object:** Dieser Tag umfasst Informationen zum konkret zu behandelnden Objekt. Muss durch den Job ein bereits bestehendes Objekt auf dem Server festgelegt werden (z. B. bei XMLUpdate), so muss hier die `object_id` (ID des Ordners, Registers oder Dokuments) als Attribut hinterlegt werden. Soll ein Standort festgelegt werden (z. B. bei XMLInsert()), so wird dieser hier als Kombination von `register_id`, `register_type` (Registerid, Registertyp) und/oder `folder_id` (Ordnerid) beschrieben. Zusätzlich ist es möglich, im Fall eines Dokumentes den `maintype` (Haupttypen) festzulegen. Varianten von W-Dokumenten können durch Füllen des Attributes '`variantparent_id`' (ID des Dokuments, dem die Variante hinzugefügt werden soll) angegeben werden (siehe auch [Options](#): `Variantsamelevel`, `Variantsetactive`). Die Unterelemente dieser Tags müssen unbedingt in der vorgegebenen Reihenfolge im XML angegeben werden. Es ist möglich, durch die Angabe des Attributes '`currency_timestamp`' im Zusammenhang mit dem Zeitstempel des Feldes 'MODIFYTIME', das vorab angefragt werden muss, zu überprüfen, ob dieses Objekt seit seiner Abfrage von einem anderen Benutzer verändert worden ist. In diesem Falle wird eine Aktualisierung verweigert. Über das Attribut `sourceparent_id` kann beim Verschieben und Löschen eines Dokumentes, das mehrere Standorte besitzt, bestimmt werden, welcher Standort verschoben werden soll.

**Warnung:** Wird der Haupttyp eines Dokumentes geändert, wenn gleichzeitig die Dokumentenhistorie geführt wird, dann wird bei der Wiederherstellung einer älteren Dokumentversion lediglich das Dokument wiederhergestellt, nicht jedoch der geänderte Haupttyp. Dieses kann bei der Bearbeitung des Dokumentes zu Inkonsistenzen führen!

**Fields:** Dieser Tag darf für jedes Objekt lediglich einmal vorhanden sein und stellt eine Liste aller Felder dar, die behandelt werden sollen.

**Field:** Dieser Tag beschreibt das Feld, das eingefügt oder verändert werden soll. Das Feld kann anhand der Attribute: `name` (Name des Feldes auf dem Datenblatt), `internal_name` (interner Feldname), `dbname` (Feldname in Datenbank), `sortpos` (Tabposition auf dem Datenblatt) oder der `osguid` (GUID) identifiziert werden. Der Text des Elementes enthält den Wert, der dem Feld zugeordnet werden soll. Dabei ist es auch möglich einen Wert auf 'leer' zu setzen, indem man das Attribut '`field_function`' (s.u.) angibt. Ist das Feld ein Systemfeld, so ist dieses dadurch zu kennzeichnen, dass das System-Attribut (`system`) den Wert '1' erhält. Mögliche Systemfelder sind:

Datenbankname	Interner Name	Bedeutung
OSOWNER	OBJECT_USERGUID	Besitzer des Objektes
FOREIGNID	OBJECT_FOREIGNID	Fremd ID eines Verweises
SYSTEMID	OBJECT_SYSTEMID	System ID eines Objektes

**Warnung:** Wird über die System ID (=0) und eine entsprechende Foreign ID auf ein Dokument verwiesen, das wiederum ein Variantenverwaltung besitzt, so kann es zu inkonsistenten Zuständen kommen, wenn Varianten mit Haupttypen aktiviert werden, die nicht mit dem Haupttyp des Verweises übereinstimmen!

Darüber hinaus kann optional ein Attribut 'field\_function' gesetzt werden. Es existieren folgende erlaubte Werte:

Attributwert	Bedeutung
NULL	Der Feldwert wird leer eingefügt (Als DB-Null)
OBJECT_ID	Der Wert des Feldes entspricht der eindeutigen internen ID des Objektes
USER	Der Name des abgemeldeteten Benutzers

**MultiFields:** Dieser Tag darf für jedes Objekt lediglich einmal vorhanden sein und stellt eine Liste aller Mehrfachfelder dar, die behandelt werden sollen. Mehrfachfelder sind nur im Falle von Dokumenten erlaubt.

**Multifield:** Durch dieses Element wird ein bestimmtes Mehrfachfeld eines Dokumentes identifiziert. Das Multifield kann anhand der Attribute: name (Name des Mehrfachfelds), internal\_name (interner Name), dbname (Name des Feldes in der Datenbank) oder der osguid (GUID) identifiziert werden.

**Page:** Dieses Element bezeichnet eine bestimmte Seite, dessen Seitennummer in dem Attribut id angegeben werden muss.

**Value:** Hier stehen die entsprechenden Werte, die den angegebenen Seiten entsprechen. Der Wert steht im Text des Elementes.

**TableFields:** Dieser Tag darf für jedes Objekt lediglich einmal vorhanden sein und stellt eine Liste aller zu behandelnden Tabellenfelder des Objektes dar.

**TableField:** Durch dieses Element wird ein bestimmtes Tabellenfeld (Table control) eines Objektes identifiziert. Ein Tabellenfeld kann anhand der Attribute: name (Name), internal\_name (interner Name), dbname (Name in der Datenbank) oder der osguid (GUID) identifiziert werden.

**Row:** Dieser Tag beschreibt eine Zeile eines Tabellenfeldes. Es ist damit die Zusammenfassung aller Spalten einer bestimmten Zeile. (Alternativ zu dieser Angabe von Tabellenfeldeinträgen kann man diese auch durch eine (LOL) Angabe von Rows und Columns bestimmen. Siehe unten).

**Field:** Dieser Tag ist der Wert einer konkreten Spalte einer Zeile des Tabellenfeldes. Es kann lediglich über die Attribute name (Name), internal\_name (internen Namen) oder dbname (Name in der Datenbank) identifiziert werden. Der Elementtext beinhaltet den Wert, der hinzugefügt werden soll.

Darüber hinaus kann optional ein Attribut 'field\_function' gesetzt werden (zu Details: siehe oben)

## ODER

**Columns:** Dieser Tag ist eine Sammlung von Column-Tags.

**Column:** Dieser Tag beschreibt ein Feld innerhalb eines Tabellencontrols. Das Feld kann anhand seiner Attribute name (Name), internal\_name (internen Namen) oder dbname (Name in der Datenbank) werden. In den folgenden Tags unter Rows stehen die Feldwerte in der Reihenfolge, in der hier die Spalten aufgeführt worden sind.

**Rows:** Dieser Tag beschreibt eine Liste von Row-Tags und damit eine Sammlung von Zeilen des Tabellen Controls.

**Row:** Hier wird eine Zeile beschrieben, die einer Zeile in dem Tabellen Control entspricht.

**Value:** Dieser Tag beschreibt einen Feldwert einer Zeile. Das zugeordnete Feld muss unter dem Tag 'Column' an der entsprechenden Stelle ebenfalls angegeben worden sein. Der Text des Elementes entspricht dem Wert des Feldes.

Darüber hinaus kann optional ein Attribut 'field\_function' gesetzt werden (zu Details: siehe oben)

**Remarks:** Dieser Tag beschreibt eine Liste von Notizen, die dem Objekt zugeordnet werden sollen.

**RemarkText:** Dieser Tag beschreibt einen Notiztext für das angegebene Objekt. Der Elementtext entspricht dem Text der Notiz.

**RemarkObject:** Dieser Tag beschreibt ein Objekt, das über die Notizen mit dem angegebenen Objekt verknüpft werden soll. Das zu verknüpfende Objekt ist über die Attribute object\_id (ID des Objekts) und object\_type (Typ des Objekts) zu bestimmen.

### Der Parameter 'Flags'

Jeder Job besitzt einen Parameter Flags. Durch die Flags können das allgemeine Verhalten des Jobs gesteuert werden. Die Flags können auch kombiniert werden. Soll zum Beispiel eine Fehlerliste in Dateiform (Flags = 1) und UTF-16 codiert (Flags = 16) zurückgeliefert werden, muss im Parameter Flags der Wert '17' (1+16) übergeben werden. Es existieren folgende Flags:

Flag	Beschreibung
1	Die Fehlerliste wird in Dateiform zurückgeliefert.
2	Der Job liefert keine Fehlerliste zurück.
4	Durch den Job wird keine Validierung mit der XSD-Datei durchgeführt.
8	Die mitgelieferten Dateien werden am Server nicht gelöscht.
16	Die zurückgelieferte Fehlerliste wird in UTF-16 codiert. (Defaultwert UTF-8)

### Der Parameter 'Options'

Durch den Parameter 'Options' können bestimmte Überprüfungen an- bzw. ausgeschaltet werden. Dies kann teilweise zu erheblicher Performancesteigerung führen. Der Parameterwert ist eine Semikolon-separierte Liste mit dem Format OPTION1=1;OPTION2=0;....

Option	Beschreibung	Default	nutzbar im Job
APPENDFILESTOFRONT	gibt an für Update und varinatenerzeugung ob anzuhängende Files vor (1) oder nach dem Bestehenden angehängt werden (0)	(0) Option nicht aktiv	XMLInsert XMLUpdate

ARCHIVABLE	gibt an, ob das Dokument den Status archivierbar (1) oder nicht archivierbar erhält (0)	XMLInsert: (0) Option nicht aktiv XMLUpdate: Option nicht definiert	XMLInsert XMLUpdate XMLCopy
ARCHIVEIMMEDIATELY	gibt an, ob das Dokument sofort archiviert wird (1) oder nicht (0)  Hinweis: Kann das Dokument nicht archiviert werden, wird kein Dokument ohne Seiten angelegt.	(0) Option nicht aktiv	XMLInsert
ARCHIVEIMMEDIATELYOBJDEF	gibt an, ob zu einem Dokument, das sofort archiviert wird (ARCHIVEIMMEDIATELY), die Objektdefinition ebenfalls archiviert wird (1) oder nicht (0).	(0) Option nicht aktiv	XMLInsert
CHECKACCESS	für das Objekt werden die Benutzerrechte geprüft (1) oder nicht geprüft (0)	(1) Option aktiv	XMLInsert XMLUpdate XMLMove XMLDelete XMLCopy
CHECKCATALOGUE	es wird geprüft, ob alle übergebenen Katalogeinträge auch im Original-Katalog enthalten sind (1) oder nicht geprüft (0)	(1) Option aktiv	XMLInsert XMLUpdate
CHECKEXISTENCE	es wird geprüft, ob das angegebene Objekt an der angegebenen Position existiert (1) oder nicht geprüft (0)	(1) Option aktiv	XMLInsert XMLUpdate XMLMove XMLDelete XMLCopy
CHECKKEYFIELDS	es wird geprüft, ob Schlüsselfelder eindeutig sind (1) oder nicht geprüft (0)	(1) Option aktiv	XMLInsert XMLUpdate
CHECKOBLIGATION	es wird geprüft, ob alle Pflichtfelder gesetzt wurden (1) oder nicht geprüft (0)	(1) Option aktiv	XMLInsert XMLUpdate
CHECKPOSITION	es wird geprüft, ob angegebene (Ziel-) Objekte (z. B. Ordner) existieren (1) oder nicht geprüft (0)	(1) Option aktiv	XMLMove XMLCopy

CHECKREADONLY	es wird geprüft, ob keine Felder verändert wurden, für die keine Schreibrechte bestehen (1) oder nicht geprüft (0)	(1) Option aktiv	XMLUpdate
DELETECASCADING	gibt an, ob Objekte kaskadierend gelöscht werden sollen, auch wenn sie Unterobjekte enthalten (1) oder nicht (0)	(0) Option nicht aktiv	XMLDelete
COPYCASCADING	gibt an, ob Objekte kaskadierend kopiert werden sollen, wenn sie Unterobjekte enthalten (1) oder nicht (0)	(0) Option nicht aktiv	XMLCopy
FULLTEXTFILEATTACHED	Gibt an, ob die letzte übermittelte Datei die Volltextdaten für das Dokument enthalten soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLInsert XMLUpdate
HARDDELETE	gibt an, ob das Objekt endgültig gelöscht werden soll, also nicht in den Papierkorb verschoben wird (1) oder nicht (0)	(0) Option nicht aktiv	XMLDelete
INITFIELDS	gibt an, ob alle nicht gesetzten Felder mit den Defaultwerten belegt werden sollen (1) oder nicht (0)	(1) Option aktiv	XMLInsert XMLUpdate
INUSERTRAY	gibt an, ob das Objekt in die Benutzer-Ablage eingefügt werden soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLMove XMLInsert
INWFTRAY	gibt an, ob das Objekt in die Workflow-Ablage eingefügt werden soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLMove XMLInsert
REPLACEFILES	gibt an, ob bereits gespeicherten Dateien durch die übergebenen Dateien ersetzt (1) oder angehängt werden (0)	XMLInsert (Varianten): (1) Option aktiv, XMLUpdate: (0) Option nicht aktiv	XMLUpdate XMLInsert
REPLACEMULTIFIELDS	gibt an, ob die übergebenen Multifelder die Originalen ersetzen sollen (1) oder angehängt werden (0)	(0) Option nicht aktiv	XMLUpdate

REPLACEREMARKS	gibt an, ob die übergebenen Notizen die Originalen ersetzen sollen (1) oder angehängt werden (0)	(0) Option nicht aktiv	XMLUpdate
REPLACETABLEFIELDS	gibt an, ob die übergebenen Tabellenfelder die Originalen ersetzen sollen (1) oder angehängt werden (0)	(0) Option nicht aktiv	XMLUpdate
TRUNCATEVALUES	gibt an, ob übergebene Strings abgeschnitten werden sollen, wenn sie länger sind, als definiert (1) oder nicht (0)	(0) Option nicht aktiv	XMLInsert XMLUpdate
TYPELESS	gibt an, ob das Objekt als typenlos in die Ablage eingefügt werden soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLInsert
UPDATEALLFIELDS	gibt an, ob nicht angegebene Felder auf leer gesetzt werden sollen (1) oder nicht (0)	(0) Option nicht aktiv	XMLInsert XMLUpdate
VARIANTSAMELEVEL	gibt an, ob die Variante auf der gleichen Ebene (1) oder als 'Untervariante' (0) eingefügt werden soll	(0) Option nicht aktiv	XMLInsert
VARIANTSETACTIVE	gibt an, ob die neue Variante im selben Schritt als 'Aktive' (1) gesetzt wird oder nicht (0)	(0) Option nicht aktiv	XMLInsert
VARIANTTRANSFERRETENTION	gibt an, ob die neue Variante die geplante Retention Zeit vom Original Dokument übernehmen soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLInsert
LINKDOCUMENT	gibt an, ob ein Dokument nur einen neuen Standort erhalten soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLCopy
WFTOUSERTRAY	Gibt an, ob beim Verschieben eines Dokumentes dieses von der Workflowablage in die Benutzerablage verschoben werden soll (1) oder nicht (0)	(0) Option nicht aktiv	XMLMove
KEEPLINKWHENEXISTS	Gibt an, ob ein bereits existierender Link als Fehler gewertet werden soll (1) oder	(0) Option nicht aktiv	XMLCopy XMLMove

	nicht (0) (Gilt für XMLCopy nur in Zusammenhang mit der Option LINKDOCUMENT)		
DELETEVARIANTMODE	Gibt an, ob das Löschen einer gegebenen inaktiven Variante zum Löschen des gesamten Variantenbaums führt (1) oder nicht (0)	(0) Option nicht aktiv	XMLDelete
COPYINDEXONLY	Gibt an, ob nur die Indexdaten kopiert werden (1) oder nicht (0)	(0) Option nicht aktiv	XMLCopy
COPYCREATEHISTORY	Gibt an, ob Informationen zum Kopieren in die Historie eingetragen werden (1) oder nicht (0)	(1) Option aktiv	XMLCopy

### Der Parameter 'JobUserGUID'

Durch den Parameter 'JobUserGUID' kann der Benutzer Kontext des XML Jobs geändert werden. Dies gilt im Zusammenhang mit allen XMLImport-Jobs, also allen Jobs 'XML...', die Modifikationen an Objekten im DMS vornehmen. Wird hier die GUID des Benutzers angegeben, so werden alle Checks (z. B. Zugriffsrechte), sowie Ablagen oder ähnliches nur von dem angegebenen Benutzer verwendet.

Achtung: Diese Option ist lediglich verfügbar, wenn die entsprechenden Jobs von anderen Jobs aus dem Server heraus aufgerufen werden. Einem Client wird das Setzen dieser Option jedoch stets verweigert!

### Der Parameter 'File\_N'

Jedem Job, der Übergabe Dateien auswertet, kann stattdessen eine Anzahl Parameter übergeben werden. Die Parameter lauten aufsteigend File\_0, File\_1, usw. Dieser Parameter ist vom Typ String und beinhaltet einen Datei-Pfad, die Datei muss somit nicht mehr in der Dateiliste übertragen werden. Dieses ist für den Fall gedacht, dass der Aufrufer sich selber auf dem Rechner des Servers befindet, in dem Fall fällt der Übertragungsaufwand der Datei über den Netzwerkadapter weg. Es ist in diesem Falle aber unbedingt sicherzustellen, dass dieser Pfad (ob lokal oder UNC-Notation) vom Server mit lesenden bzw. schreibenden Rechten erreichbar ist. Es kann entweder nur eine Liste von Dateien übergeben werden oder eben eine Menge File-Parameter, eine Mischung von beidem ist nicht zulässig.

### Der Rückgabewert

Jeder Job generiert neben jobspezifischen Rückgabewerten eine Rückgabenummer. Diese ist im Erfolgsfall stets '0'. Ist ein Fehler aufgetreten, so kann er über die Rückgabenummer grob qualifiziert werden. Für eine detaillierte Analyse sind jedoch die genauen Fehlermeldungen unerlässlich. Es existieren folgende Rückgabenummern:

Fehlernummer	Beschreibung
0	Der Job ist erfolgreich ausgeführt worden.



-1	Es ist ein allgemeiner Fehler aufgetreten. (In diesem Falle kann der Fehler nicht näher spezifiziert werden)
-2	Es ist kein Schrank angegeben worden.
-3	Der angegebene Schrank ist unbekannt.
-4	Es ist kein Registertyp angegeben worden.
-5	Der angegebene Registertyp ist unbekannt.
-6	Es wurde kein Dokumenttyp angegeben.
-7	Der angegebene Dokumenttyp ist unbekannt.
-8	Das angegebene Register befindet sich nicht im angegebenen Ordner.
-9	Der angegebene Dokumenttyp ist in dem angegebenen Schrank nicht zugelassen.
-10	Die benötigte Identifizierung des Ordners fehlt.
-11	Die benötigte Identifizierung des Dokumentes fehlt.
-12	Die benötigte Identifizierung des Registers fehlt.
-13	Die benötigte Identifizierung des Registers ist unbekannt.
-14	Die benötigte Identifizierung des Ordners ist unbekannt.
-15	Die benötigte Identifizierung des Dokumentes ist unbekannt.
-16	Die Aktualisierung des Ordners ist fehlgeschlagen.
-17	Die Aktualisierung des Dokumentes ist fehlgeschlagen.
-18	Die Aktualisierung des Registers ist fehlgeschlagen.
-21	Das Dokument wurde bereits archiviert.
-22	Das angegebene/benötigte Objekt ist unbekannt.
-23	Die ID des angegebenen Registers existiert nicht auf dem Archivserver.
-24	Der Feldname konnte nicht aufgelöst werden.
-28	Der Wert für ein gegebenes Feld ist nicht erlaubt.
-29	Die angegebene ObjektId ist ungültig.
-30	Die Pflichtfelder sind nicht gefüllt worden.
-31	Der angegebene Wert stimmt nicht mit dem Typ auf dem Archivserver überein.
-32	Die angegebene/benötigte Datei existiert nicht.
-40	Ein Übergabeparameter ist fehlerhaft oder fehlt vollständig.
-47	Der Benutzer besitzt die entsprechenden Rechte auf dem Archivserver nicht.
-51	Das Dokument besitzt keine Seiten.
-65	Es konnte kein Index vom Server bezogen werden.
-68	Das Verschieben von Ordnern ist nicht erlaubt.
-89	Die Beziehung zwischen Dokument und Register ist nicht erlaubt.



-90	Verweisdokumente können nicht ohne Angabe eines Standortes verschoben werden
-94	Es sind keine Dokument-Seiten erlaubt.
-1001	Der angegebene Wert konnte im zugehörigen Katalog nicht gefunden werden.
-1002	Das Schlüsselfeld ist nicht eindeutig.
-1003	Es wurde versucht, ein schreibgeschütztes Feld zu setzen.
-1004	Es wurde keine Dokumentliste angegeben.
-1005	Das angegebene Objekt liegt ist mit einem Workflow-Prozess verbunden.
-1006	Die angefragte Funktionalität ist in der vorliegenden Version nicht implementiert.
-1007	Beim Lesen der Objektdefinitionen vom ObjDefReader ist ein Fehler aufgetreten.
-1008	Es konnte auf ein angefordertes File nicht zugegriffen werden.
-1009	Das Objekt liegt im Papierkorb.
-1010	Das Dokument liegt in einer Mappe.
-1011	Die Rekurrierungstiefe ist zu groß, die Aktion wurde abgebrochen.
-1012	Das Zielregister ist ein Kind des zu verschiebenden Registers.
-1013	Es wurde eine System ID, jedoch keine Foreign ID angegeben.
-1014	Einem Verweisdokument können keine Seiten hinzugefügt werden.
-1015	Das Dokument kann nicht auf ein anderes Dokument verweisen, da es Seiten besitzt.
-1016	Beim Parsen des XML Textes ist ein Fehler aufgetreten.
-1017	Bei der Validierung des XML Textes ist ein Fehler aufgetreten.
-1018	Der XML Text ist unvollständig.
-1019	Der angegebene Besitzer konnte nicht ermittelt werden.
-1020	Der Objekttyp ist bei dieser Operation ungültig.
-1021	Es darf nicht kaskadierend gelöscht werden, das Objekt besitzt jedoch noch Unterobjekte.
-1022	Der Verweis wurde nicht gefunden.
-1023	Das angegebene Systemfeld darf nicht vom Benutzer geändert werden.
-1024	Ein Jobparameter fehlt.
-1025	Der angegebene Parameterwert ist ungültig.
-1026	Es ist für den Benutzer nicht möglich, die Rechteprüfung abzustellen.
-1027	Für ein Tabellenfeld wurden mehr Werte angegeben als Spalten vorhanden sind.
-1028	Beim Einfügen der Notiz-Objekte/-Texte ist ein Fehler aufgetreten.
-1029	Ein XML Element ist unbekannt.

-1030	Ein Objekttyp ist auf dem DMS Server nicht gefunden worden.
-1031	Ein Objekt Feld ist auf dem DMS Server nicht gefunden worden.
-1032	Der Job wurde auf Benutzeranfrage hin abgebrochen.
-1033	Die angegebene Bedingung ist ungültig.
-1034	Ein XML Attribut ist fehlerhaft.
-1035	Ein notwendiges XML Attribut fehlt.
-1036	Es dürfen nur Dokumente ausgecheckt werden.
-1037	Das Dokument ist eingechekt worden.
-1038	Das Dokument ist ausgecheckt worden.
-1039	Das Dokument wurde an einen anderen Benutzer ausgecheckt.
-1040	Das Dokument wurde an eine andere Station ausgecheckt.
-1041	Das Dokument kann nicht ausgecheckt worden, da es keine Seiten besitzt.
-1042	Das Dokument befindet sich nicht in der Workflow-Ablage.
-1043	Das Dokument befindet sich in der Workflow-Ablage.
-1044	Die Parent-Variante des Dokumentes konnte nicht ermittelt werden.
-1045	Es dürfen nur für W-Dokumente neue Varianten angelegt werden.
-1046	Die Dokument-Variante konnte nicht ermittelt werden.
-1047	Die Angaben für die Benutzerdaten sind nicht eindeutig
-1048	Ein konkurrierende Aktualisierung eines Objektes ist fehlgeschlagen
-1049	Die Anfrage wurde nicht gefunden
-1050	Die Kopie des Objektes existiert bereits an dem angegebenen Standort
-1051	Das Anfrageformat wird nicht unterstützt
-1052	Der Schrankname konnte nicht ausgelesen werden
-1053	Der Dokumentenname konnte nicht ausgelesen werden
-1054	Der Registernamen konnte nicht ausgelesen werden
-1055	Ein Ausdruck besitzt ein ungültiges Format
-1056	Der Sektionsname ist ungültig
-1057	Allgemeiner Fehler beim Auslesen der Anfrage
-1058	Unbekannter Anfragetyp
-1059	Die Neue Variante ist angelegt worden, konnte jedoch nicht auf 'Aktiv' gesetzt werden
-1060	Die Anfrage konnte nicht bearbeitet werden.
-1061	Das Verschieben bzw. Verlinken von Objekten ist systemweit gesperrt
-1062	Der Standort des Objektes konnte nicht ermittelt werden

-1063	Das Dokument existiert bereits an dem Standort
-1064	Das Kopieren von Objekten mit Schlüsselfeldern ist nicht erlaubt
-1065	Das angegebene Passwort ist falsch
-1066	Das neue und das alte Passwort unterscheiden sich nicht
-1067	Die Volltextanfrage enthält nur zu ignorierende Wörter
-1068	Es steht nicht genügend Speicher zur Verfügung
-1069	Der Zugriff auf einen Systemressource wurde verweigert.
-1070	Im Job XMLImport ist als Aktion bzgl. der Anzahl von Treffern 'Fehler' angegeben worden

## Jobübergreifende Restriktionen

Die folgenden Beschreibungen sind allgemeingültig, gelten also für alle XML Import Jobs.

### Datums- und Zeitformate

Datumsfelder können in den unten aufgeführten Formaten importiert werden. Dabei entspricht TT dem immer zweistelligen Tag ('09' statt '9'), MM dem Monat und JJ dem zweistelligen bzw. JJJJ dem vierstelligen Jahr.

§ TTMMJJ

§ TTMMJJJJ

§ TT.MM.JJJJ

Zeitfelder können im Format HH:MM:SS, wobei HH der zweistelligen Stunden, MM den Minuten und SS der Sekunden entspricht, importiert werden. Zeitstempel werden grundsätzlich im Format TT.MM.JJJJ HH:MM:SS importiert, wobei das Datum und die Zeit durch ein Leerzeichen getrennt sind.

### Besondere Feldtypen

§ Ein Kontrollkästchen darf lediglich die Werte 0 oder 1 enthalten.

§ Eine Optionsschaltfläche kann als Feld mit dem Namen der ersten (nach der Tabulator Reihenfolge) Schaltfläche identifiziert werden. Sind diese darüber hinaus über ein Gruppenfeld vorschriftsmäßig gruppiert, so kann das Feld über den Namen des Gruppenfeldes identifiziert werden. Erlaubte Werte sind die mit 0 beginnende Reihenfolge der entsprechenden Schaltflächen.

§ Die Feldtypen 'Patientenart', 'Seite', 'Geschlecht' und 'Frage' können sowohl den ersten Buchstaben, als auch den vollständig ausgeschriebenen Wert besitzen.

§ Dezimalwerte dürfen mit einem Plus- oder Minuszeichen beginnen. Die Vorkommastellen und Nachkommastellen dürfen sowohl durch einen Punkt als auch durch ein Komma getrennt werden.

§ In mehrzeiligen Textfelder können die einzelnen Zeilen der Werte durch ein 'Carriage Return Line Feed' getrennt werden. In XML bedeutet dieses die Kombination: &#13;&#10.

### Zusätzliche Feldwerte

Alle Felder können den Wert über das Attribut 'field\_function' auch den Wert NULL annehmen. In diesem Fall bedeutet dies, dass das Feld als leer erwünscht wird. Ein numerisches Feld oder auch

Textfeld kann den Wert der internen ID des Objektes über dasselbe Attribut annehmen. In diesem Falle wird der Wert des Feldes durch die neue oder bestehende ID des entsprechenden Objektes ersetzt.

### Die XML-Fehlerliste

Sollte ein Fehler aufgetreten sein, so kann er zwar anhand der Fehlernummer im Rückgabewert bestimmt werden, jedoch ist der exakt aufgetretene Fehler oftmals nicht hinreichend genau beschrieben. Aus diesem Grund erzeugen alle Jobs eine Fehlerliste. Diese kann im Job nach der Ausführung mit `GetErrorList()` ermittelt werden. In dieser Fehlerliste stehen die genauen Beschreibungen der zuletzt aufgetretenen Fehler. Die zuerst beschriebenen Fehler sind die aussagekräftigsten. Diese Fehlerliste wird auch standardmäßig im Rückgabeparameter 'DMSResult' als XML geliefert. Optional kann dieses XML auch als Rückgabe-Datei zurückgeliefert werden.

#### Beispiel:

##### XML-Fehlerliste

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DMSData>
<Messages>
<Message Sourcename="oxjobdms">Das Feld &gt;Tabelle - Spalte2&lt; besitzt
einen Katalog; der Wert &gt;7&lt; konnte jedoch nicht im Katalog
gefunden werden</Message>
<Message Sourcename="oxjobdms">SAX Fehler: Common exception</Message>
<Message Sourcename="oxjobdms">Beim XML Import ist ein Fehler
aufgetreten</Message>
</Messages>
</DMSData>
```

Ansonsten können alle auf dem Server aufgetretenen Fehler im Rahmen der üblichen Serverprotokollierung eingesehen werden.

## DMS.CheckInDocument

#### Beschreibung:

Dieser Job checkt das angegebene Dokument ein. Das Dokument wird zum Server übertragen und am Standort (im Parameter Dateiliste angegeben) gelöscht.

#### Parameter:

Flags (INT): 1 = das übergebene Dokument darf von einer anderen Station eingecheckt werden, ansonsten 0

[ArchiveType] (INT): Typ des Schrankes

ObjectType (INT): Typ des Objekts

ObjectID (INT): ID des Dokuments

Dateiliste: Pfad und Name der einzucheckenden Datei

#### Rückgabewerte:

Info (STRING): wurde das Dokument von einem anderen Benutzer/Station ausgecheckt, wird der Name zurückgeben

#### Siehe auch:

[DMS.CheckOutDocument](#)

## DMS.CheckOutDocument

### Beschreibung:

Dieser Job checkt das angegebene Dokument aus. Das Dokument wird in der Datenbank als ausgecheckt markiert. Das Dokument selbst muss dann mit Hilfe des Jobs [std.StoreInCache](#) vom Server geholt werden.

### Parameter:

Flags (INT): 1 = das Dokument wird nach 'extern' ausgecheckt werden, ansonsten 0

ArchiveType (INT): Typ des Schrankes

ObjectType (INT): Typ des Objekts

ObjectID (INT): ID des Dokuments

### Rückgabewerte:

[Info] (STRING): ist das Dokument bereits ausgecheckt, wird der Name des Auscheckenden zurückgegeben

### Siehe auch:

[DMS.CheckInDocument](#) , [DMS.UndoCheckOutDocument](#)

## DMS.UndoCheckOutDocument

### Beschreibung:

Dieser Job macht das Auschecken des angegebenen Dokuments rückgängig.

### Parameter:

Flags (INT): 1 = das Auschecken darf von einer anderen Station rückgängig gemacht werden, ansonsten 0

ArchiveType (INT): Typ des Schrankes

ObjectID (INT): ID des Dokumentes

ObjectType (INT): Typ des Objekts

### Rückgabewerte:

[Info] (STRING): Wenn ein anderer Benutzer versucht das Auschecken rückgängig zu machen, wird der Name des Auscheckenden zurückgegeben (gleiches gilt für die Station)

## DMS.GetXMLJobOptions

### Beschreibung:

Dieser Job liefert alle Optionen der Jobs XMLInsert, XMLUpdate, XMLMove und XMLDelete.

### Rückgabewerte:

JobOptions (BASE64): Liste aller Joboptionen im XML-Format

### Beispiel:

#### Aufbau von JobOptions

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DMSOptions>
<DMSOption optionstring="" description="" defaultvalue="" />
```

```
<DMSOption optionstring="" description="" defaultvalue="" />
.
.
.
</DMSOptions>
```

**Hinweis:**

Genauere Beschreibung von JobOptions

§ optionstring: Name der Option

§ description: Beschreibung der Option

§ defaultvalue: voreingestellter Wert

§ SET: die Option ist standardmäßig aktiviert

§ NOT\_SET: die Option ist standardmäßig nicht aktiviert

§ UNDEFINED: die Option ist standardmäßig nicht definiert

**Siehe auch:**

[Der Parameter 'Options'](#)

## DMS.RestoreIndexdataVersion

**Beschreibung:**

Dieser Job stellt eine angegebene Version aus der Indexdatenhistorie wieder her.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

ObjectID (INT): ID des Dokumentes, dessen Indexdaten wiederhergestellt werden sollen

Guid (STRING): GUID der Version des Objektes, das wiederhergestellt werden soll

**Rückgabewerte:** keine jobspezifischen

## DMS.XMLDelete

**Beschreibung:**

Dieser Job löscht das angegebene Objekt. Es können Ordner, Register oder Dokumente gelöscht werden. Grundsätzlich wird zwischen zwei Arten zu löschen durch die Option 'HARDDDELETE' unterschieden:

§ 'Physikalisch löschen': Das Objekt wird vollständig gelöscht, also nicht den Papierkorb verschoben. Im Falle, dass das Objekt ein Dokument ist, werden auch alle seine Dateien vom Server entfernt.

§ 'In den Papierkorb löschen': Das Objekt wird nicht physikalisch entfernt, sondern befindet sich anschließend im Papierkorb und kann von dort aus wiederhergestellt werden.

Ein Objekt kann Unterobjekte besitzen (z.B ein Ordner enthält Dokumente). Dieses Objekt kann nicht gelöscht werden, wenn nicht alle seine Unterobjekte gelöscht werden. Dieses wird durch die Option 'DELETECASCADING' gesteuert.

Das Löschen von Objekten, die einem Workflowprozess zugeordnet sind, ist nicht möglich. Für den Fall, dass ein Dokument mehrere Standorte besitzt, wird der Job im Allgemeinen das Dokument von allen seinen Standorten löschen. Es ist jedoch auch möglich, das Dokument von genau einem solchen Standort zu entfernen. Dazu sind im Objekt-Element die Eltern-Attribute (Registerid, Register-Typ oder Ordnerid) zu setzen. Dieses stellt nun den Standort des zu entfernenden Dokuments dar.

**Parameter:**

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen (z. B. HARDDELETE=1 ;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#))

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

**Rückgabewerte:** keine jobspezifischen

**Hinweis:**

Die folgenden XML-Beispiele enthalten immer alle Tags und Tag-Attribute, die für die jeweilige Aktion verwendet werden können. Nicht benötigte Tags bzw. Attribute können natürlich weggelassen werden.

**Beispiel:****XML für das Löschen eines Ordners**

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType name="Pressearchiv" type="FOLDER" internal_name="" osguid=""
table="" id="-1">
<Object object_id="214"/>
</ObjectType>
</Archive>
</DMSData>
```

**Beispiel:****XML für das Löschen eines Registers**

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType type="REGISTER" name="Register" internal_name="" osguid=""
table="" id="-1">
<Object object_id="229"/>
</ObjectType>
</Archive>
</DMSData>
```

**Beispiel:****XML für das Löschen eines Dokuments**

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive name="Pressearchiv" id="-1" internal_name="" osguid="">
<ObjectType name="Word-Texte" maintype="4" cotype="0" type="DOCUMENT"
id="-1" internal_name="" osguid="" table="">
<Object object_id="214" variantparent_id="-1"/>
</ObjectType>
</Archive>
</DMSData>
```

**DMS.XMLInsert****Beschreibung:**

Dieser Job fügt ein Objekt in enaio® ein. Es kann ein Ordner, Register oder Dokument eingefügt werden. Der Rückgabeparameter 'ObjectID' entspricht der ID des hinzugefügten Objektes oder ist -1, wenn der Job fehlgeschlagen ist. Ist das einzufügende Objekt ein Dokument und soll dieses Dateien besitzen, die auf den Archivserver übertragen werden sollen, so ist die Eingabe Dateiliste mit den entsprechenden Datei-Pfadangaben zu füllen. Dabei können ebenfalls Dias übertragen werden. Sollten mehr als ein Dia übertragen werden, so wird lediglich das erste Dia aus der Liste am Archivserver eingefügt. Der Standort des neuen Objektes muss im Falle eines Dokumentes oder Registers in das XML Objekt Element eingetragen werden. Ist hier nur die Register ID eingetragen, so wird Registertyp und Schrank automatisch ermittelt. Aus Performancegründen sollten diese Angaben jedoch immer getätigt werden.

**Parameter:**

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen

(z. B. ARCHIVABLE=1;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#))

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

[Dateiliste]: Pfad und Name der einzufügenden Dokumente

File\_N: (STRING) N-ter Dateipfad als Alternative zur Dateliste

**Rückgabewerte:**

ObjectID (INT): neue Objektid , wenn Job erfolgreich, ansonsten -1

ObjectType (INT): Typ des Objekts, ansonsten -1

[Dateiliste]: Pfad und Name der XML-Datei mit der Fehlern (siehe [Flags](#))

**Hinweis:**

Die folgenden XML-Beispiele enthalten immer alle Tags und Tag-Attribute, die für die jeweilige Aktion verwendet werden können. Nicht benötigte Tags bzw. Attribute können natürlich weggelassen werden.

**Wichtig:**

Wenn in den Tags wichtige Attribute wie z. B.: maintype, register\_id, register\_type oder system nicht gebraucht werden, sollte man sie entweder ganz weglassen, auf '0' oder auf '-1' setzen, je nach Funktionalität.

**Wichtig:**

Das folgende Beispiel mit den '<TableFields/>' Tags funktionieren nur, wenn sich eine Tabelle in der Verschlagwortungsmaske für den Schrank befindet.

**Beispiel:**

XML für das Einfügen eines Ordners in einen Schrank

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" name="Pressearchiv" internal_name=""
osguid="" type="FOLDER">
<Object>
<Fields>
<Field dbname="" name="Fachgebiet" internal_name=""
```



```

osguid="">Softwareentwicklung</Field>

<Field dbname="feld2" system="" name="" internal_name=""
osguid="">Testbenutzer</Field>
</Fields>
<TableFields>
<TableField dbname="Tabelle" system="" name="" internal_name="">
<Row>
<Field name="Bezeichnung" internal_name=""
dbname="">Dokumentation</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="WHITE">Hier kann eine Notiz hinterlegt
werden.</RemarkText>
<RemarkObject action="INSERT" object_id="123"
object_type="196616">Verknüpfungsnotiz</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>

```

**Wichtig:**

Das folgende Beispiel mit den '<TableFields/>' Tags funktionieren nur, wenn sich eine Tabelle in der Vorschlagwortungsmaske für das Register befindet.

**Beispiel:****XML für das Einfügen eines Registers**

```

<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" name="Jahr2004" internal_name="" osguid=""
type="REGISTER">
<Object folder_id="228">
<Fields>
<Field dbname="" system="" name="Kategorie" internal_name=""
osguid="">Neuentwicklung</Field>

<Field dbname="" system="" sortpos="" name="" internal_name=""
osguid="BDED8A3C99E64AD2A4ECBFDB586">öffentlich</Field>

</Fields>
<TableFields>
<TableField dbname="" name="Tabelle" internal_name="">
<Row>
<Field name="Thema" internal_name=""
dbname="">Dokumentenmanagement</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="WHITE">Eine Notiz zum Register.</RemarkText>
<RemarkObject action="INSERT" object_id="234"
object_type="196616">Verknüpfungsnotiz</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>

```

**Wichtig:**

Das folgende Beispiel mit den '<TableFields/>' Tags funktionieren nur, wenn sich eine Tabelle in der Verschlagwortungsmaske für das Dokument befindet.

**Beispiel:****XML für das Einfügen eines Dokuments**

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" maintype="4" cotype="0" name="Word-Texte"
internal_name="" osguid="" type="DOCUMENT">
<Object object_id="-1" folder_id="" register_id="78" register_type="0"
variantparent_id="-1" maintype="0">
<Fields>
<Field dbname="" system="0" name="Autor" internal_name=""
osguid="">Testuser</Field>
</Fields>
<MultiFields>
<MultiField dbname="" system="0" name="" internal_name=""
osguid="2AED8A3399EE778DS4ECBFDB582">
<Page id="1">
<Value>345</Value>
</Page>
<Page id="2">
<Value>123</Value>
</Page>
</MultiField>
</MultiFields>
<TableFields>
<TableField dbname="" name="Tabelle" internal_name="" osguid="">
<Row>
<Field name="Team" internal_name=""
dbname="">Entwicklung</Field>
<Field name="" internal_name="feld2" dbname="">
Status: freigegeben</Field>
</Row>
</TableField>
<TableField dbname="" name="" internal_name=""
osguid="AAED8A3C99EED78DS4ECBFDB586">
<Row>
<Field name="" internal_name="" dbname="fd1">Jahr 2004</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="BLUE">Eine Notiz zum Dokument.</RemarkText>
<RemarkObject action="INSERT" object_id="432"
object_type="196616">Verknüpfungsnotiz</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>
```

**DMS.XMLMove****Beschreibung:**

Dieser Job verschiebt ein Objekt. Es kann ein Register oder Dokument verschoben werden. Die Eltern-Attribute (Registerid, Registertyp und Ordnerid) sind die Attribute, die den neuen Standort

bezeichnen. Dabei sollte die Ordnerid auch in dem Fall angegeben werden, dass der neue Standort des Objektes ein Register ist.

Wird ein Register verschoben, so wird es mitsamt seinen Unterobjekten zum neuen Standort verschoben. Darüber hinaus müssen für Verweisdokumente der Standort als Attribut 'sourceparent\_id' im Object Tag angegeben werden.

Dokumente können in die Benutzerablage verschoben werden. Dazu brauchen keine Standortbestimmungen im Objekt-Element des XML angegeben werden, aber die Option 'WFTOUSERTRAY' muss dazu aktiviert werden. Das zu verschiebende Dokument muss zuvor in der Workflowablage gelegen haben, Dokumente aus Ordnern und Registern dürfen nicht in die Benutzerablage verschoben werden.

Dokumente können aus der Benutzerablage oder der Workflowablage verschoben werden. Diese Dokumente können dann in ein Register oder einen Ordner verschoben werden. Der Job erkennt dabei automatisch, ob sich das Objekt in einem Register, Ordner, der Workflowablage oder der Benutzerablage befindet.

**Parameter:**

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen

(z. B. ARCHIVABLE=1;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#))

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

**Rückgabewerte:** keine jobspezifischen

**Hinweis:**

Die folgenden XML-Beispiele enthalten immer alle Tags und Tag-Attribute, die für die jeweilige Aktion verwendet werden können. Nicht benötigte Tags bzw. Attribute können natürlich weggelassen werden.

**Wichtig:**

Wenn in den Tags wichtige Attribute wie z. B.: id nicht gebraucht werden, sollte man sie entweder ganz weglassen, auf '0' oder auf '-1' setzen, je nach Funktionalität.

**Beispiel:**

XML für das Verschieben eines Registers

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType type="REGISTER" name="Register" internal_name="" osguid=""
table="" id="-1">
<Object object_id="28" folder_id="58" register_id="-1"
register_type="-1"/>
</ObjectType>
</Archive>
</DMSData>
```

**Beispiel:**

XML für das Verschieben eines Dokuments

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
```

```
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType type="DOCUMENT" name="Word-Texte" maintype="4" cotype="0"
internal_name="" osguid="" table=""
id="-1">
<Object object_id="248" register_id="228" register_type="6488064"
folder_id="58"/>
</ObjectType>
</Archive>
</DMSData>
```

## DMS.XMLCopy

### Beschreibung:

Dieser Job kopiert Objekte, es können Ordner, Register und Dokumente kopiert werden. Werden Register oder Ordner kopiert, so kann mit der Option 'COPYCASCADING' bestimmt werden, ob alle enthaltenen Objekte kaskadierend mit kopiert werden sollen.

Wird ein Dokument kopiert, so kann es als neues Dokument kopiert werden oder am neuen Standort kann ein Verweis des ursprünglichen Dokumentes entstehen. Das Dokument besitzt dann zwei Standorte, es existiert dann jedoch lediglich ein Indexdatensatz. Um das Dokument so zu verlinken, ist die Joboption 'LINKDOCUMENT' anzugeben. Für eine solche Verweiskopie müssen sich der Quell- und der Zielstandort unterscheiden. Diese Option gilt auch für alle Dokumente, die durch die Angabe der Option 'COPYCASCADING' kopiert werden.

Die Eltern-Attribute (Registerid, Registertyp und Ordnerid) sind die Attribute, die den neuen Standort bezeichnen. Dabei sollte die Ordnerid auch in dem Fall angegeben werden, dass der neue Standort des Objektes ein Register ist. Der Registertyp kann vom Executor ermittelt werden, aus Performancegründen sollte dieses jedoch immer angegeben werden.

### Parameter:

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen

(z. B. LINKDOCUMENT=1;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#))

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

File\_N: (STRING) N-ter Dateipfad als Alternative zur Dateiliste

**Rückgabewerte:** keine jobspezifischen

### Hinweis:

Die folgenden XML-Beispiele enthalten immer alle Tags und Tag-Attribute, die für die jeweilige Aktion verwendet werden können. Nicht benötigte Tags bzw. Attribute können natürlich weggelassen werden.

### Beispiel:

XML für das Erstellen eines neuen Standortes für ein Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType type="DOCUMENT" name="Word-Texte" internal_name="" osguid=""
table="" id="-1">
<Object object_id="28" folder_id="58" register_id="-1"
register_type="-1"/>
</ObjectType>
</Archive>
```

```
</ObjectType>
</Archive>
</DMSData>
```

## DMS.XMLUpdate

### Beschreibung:

Dieser Job ändert das angegebene Objekt. Es kann ein Ordner, Register oder Dokument aktualisiert werden. Ist das einzufügende Objekt ein Dokument, können der Datei-Liste Pfade auf Dateien übergeben werden. Ob diese Dateien die bestehenden ersetzen oder an diese angehängt werden sollen wird durch die Option 'REPLACEFILES' bestimmt.

Dokumente können zu Verweisdokumenten gemacht werden, solange sie keine Dateien besitzen. Dieses geschieht durch das Setzen des Systemfeldes 'Fremd-ID' (und evtl. System-ID). Für Verweisdokumente gilt ähnliches: Sie können wieder zu Dokumenten gemacht werden, indem die Fremd-ID (und evtl. System-ID) geleert werden. Dieses geschieht durch Setzen der Systemfelder mithilfe des Attributes 'field\_function' und dem Wert 'NULL'.

Durch Angabe des Systemfelds 'Besitzers' kann der Besitzer geändert werden. Dieser darf jedoch nicht mittels der GUID angegeben werden, sondern muss als Wert dem Benutzernamen entsprechen.

Wird im Object Tag das Attribut 'concurrency\_timestamp' mit einem Wert gefüllt, so wird überprüft, ob sich das der Datensatz seit dem letzten Abruf des Datensatzes des Aufrufers geändert hat, und liefert einen Fehler zurück. So kann verhindert werden, dass Änderungen, die zwischen dem Abruf und der gewünschten Änderung, vorgenommen werden, verloren gehen.

Standardmäßig werden Zeilen von Tabellencontrols an die bestehenden Daten angehängt. Wird im <Row>- Element das Attribut 'line' mit einer Zeilennummer gesetzt, wird die betreffende Zeile aktualisiert. Die Nummerierung der Zeilen ist dabei 1-basiert. Wird die Job Option REPLACETABLEFIELDS gesetzt, werden alle bestehenden Zeilen vollständig durch die neuen Daten ersetzt.

Die Job Option REPLACEREMARKS legt fest, ob beim Anlegen von Notizen oder Verknüpfungen alle bestehende Notizen und ebenfalls alle bestehenden Verknüpfungen gelöscht werden oder die Notizen und Verknüpfungen hinzugefügt werden (Default). 'RemarkText' und 'RemarkObject' haben das Attribut 'action' mit folgenden Werten: INSERT (Default), DELETE, UPDATE, UPDATE\_TEXT, UPDATE\_COLOR. Bestehende Notizen werden über die ID (remark\_id), Verknüpfungen über Objekt-ID und Objekttyp-ID des verknüpften Objekts angegeben.

### Parameter:

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen

(z. B. ARCHIVABLE=1;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#))

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

[Datei-Liste]: Pfad und Name des geänderten Dokuments

File\_N: (STRING) N-ter Dateipfad als Alternative zur Dateliste

**Rückgabewerte:** keine jobspezifischen

### Hinweis:

Die folgenden XML-Beispiele enthalten immer alle Tags und Tag-Attribute, die für die jeweilige Aktion verwendet werden können. Nicht benötigte Tags bzw. Attribute können natürlich weggelassen werden.

### Beispiel:

#### XML für das Ändern eines Ordners

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" name="Pressearchiv" internal_name=""
osguid="" type="FOLDER">
<Object object_id="54">
<Fields>
<Field dbname="" system="0" name="Fachgebiet"
internal_name="" osguid="">Softwareentwicklung</Field>
<Field dbname="feld2" system="0" internal_name=""
osguid="">Testbenutzer</Field>
</Fields>
<TableFields>
<TableField dbname="Tabelle" system="0" name="" internal_name="">
<Row>
<Field name="Bezeichnung" internal_name=""
dbname="">Dokumentation</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="WHITE">Eine Notiz zum Ordner.</RemarkText>
<RemarkObject action="INSERT" object_id="123"
object_type="196616">Verknüpfungstext</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>
```

### Beispiel:

#### XML für das Ändern eines Registers

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" name="Jahr2004" internal_name="" osguid=""
type="REGISTER">
<Object object_id="312" folder_id="228" register_id="-1"
register_type="-1">
<Fields>
<Field dbname="" system="0" name="Kategorie"
internal_name="" osguid="">Neuentwicklung</Field>
<Field dbname="" system="0" name="" internal_name=""
osguid="BDED8A3C99E64AD2A4ECBFD586">öffentlich</Field>
</Fields>
<TableFields>
<TableField dbname="" name="Tabelle" internal_name="">
<Row>
<Field name="Thema" internal_name=""
dbname="">Dokumentenmanagement</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="WHITE">Eine Notiz zum Register.</RemarkText>
```

```
<RemarkObject action="INSERT" object_id="234"
object_type="196616">Verknüpfungstext</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>
```

**Beispiel:****XML für das Ändern eines Dokuments**

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" maintype="4" cotype="0" name="Word-Texte"
internal_name="" osguid="" type="DOCUMENT">

<Object object_id="221" folder_id="" register_id="-1" register_type=""
variantparent_id="-1" maintype="0">
<Fields>
<Field dbname="" system="0" name="Autor" internal_name=""
osguid="">Testuser</Field>
</Fields>
<MultiFields>
<MultiField dbname="" system="0" name="" internal_name=""
osguid="2AED8A3399EE778DS4ECBFDB582">
<Page id="1">
<Value>345</Value>
</Page>
<Page id="2">
<Value>123</Value>
</Page>
</MultiField>
</MultiFields>
<TableFields>
<TableField dbname="" name="Tabelle" internal_name="" osguid="">
<Row>
<Field name="Team" internal_name=""
dbname="">Entwicklung</Field>
<Field name="" internal_name="feld2"
dbname="">Status: freigegeben</Field>
</Row>
</TableField>
<TableField dbname="" name="" internal_name=""
osguid="AAED8A3C99EED78DS4ECBFDB586">
<Row>
<Field name="" internal_name="" dbname="fd1">Jahr 2004</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="BLUE">Eine Notiz zum Dokument.</RemarkText>
<RemarkObject action="INSERT" object_id="432"
object_type="196616">Verknüpfungstext</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>
```

**DMS.XMLImport****Beschreibung:**

Dieser Job ermöglicht das Einfügen oder Aktualisieren eines Objektes in Abhängigkeit des Ergebnisses einer vorhergehenden Suche.

#### Parameter:

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen

(z. B. ARCHIVABLE=1;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#)) mit der Erweiterung um das Tag <SearchFields> für die Suchfelder. S. Diagramm unten.

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

[Action0]: Durchzuführende Aktion bei keinem Treffer (s. Aktionstabelle)

[Action1]: Durchzuführende Aktion bei einem Treffer (s. Aktionstabelle)

[ActionM]: Durchzuführende Aktion bei mehreren Treffern (s. Aktionstabelle)

[Dateiliste]: Pfad und Name der einzufügenden Dokumente

#### Rückgabewerte:

ObjectID (INT): neue Objektid ,wenn Job erfolgreich, ansonsten -1

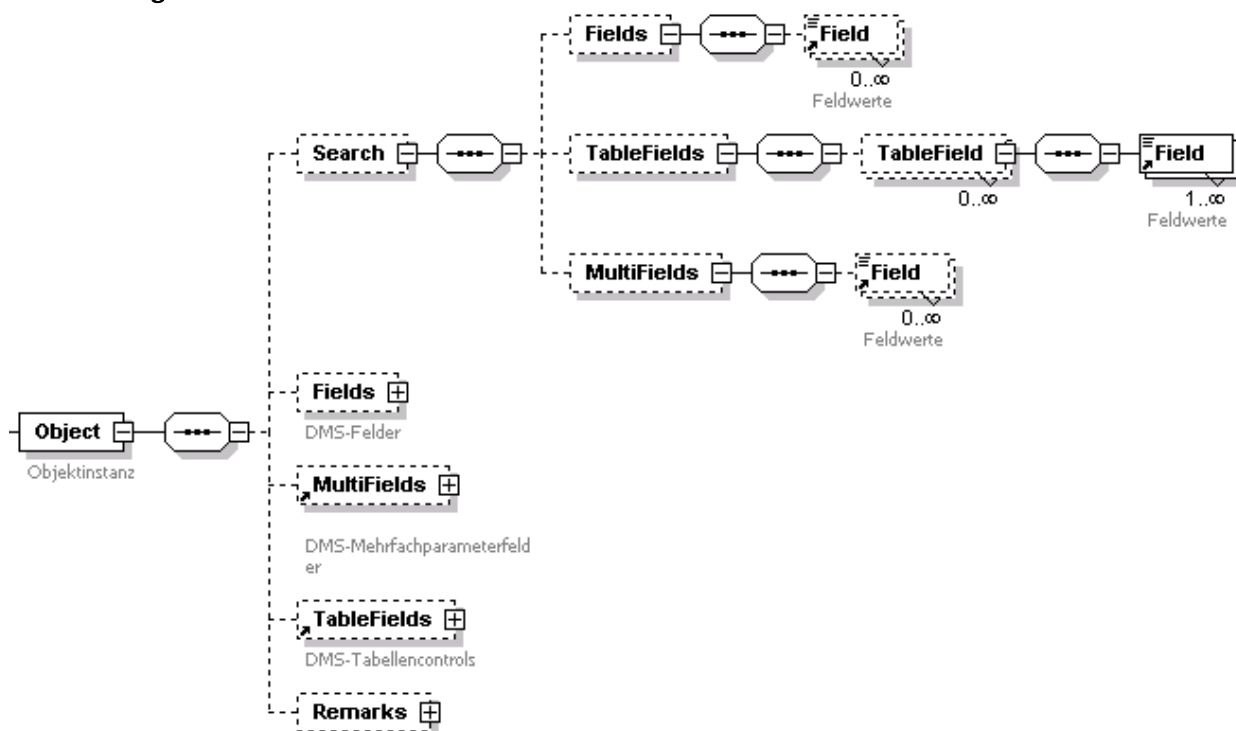
ObjectType (INT): Typ des Objekts, ansonsten -1

Hits (INT): Anzahl der Treffer der Suche

Action: Durchgeführte Aktion. Mögliche Werte sind UPDATE, INSERT, NONE, ERROR

[Dateiliste]: Pfad und Name der XML-Datei mit den Fehlern (siehe Flags)

#### Erweiterung des XML Formates:





**Aktionstabelle**

Suchergebnis [Trefferzahl]	Parametername	Möglicher Parameterwert	Erläuterung
0	Action0	INSERT [Default] NONE ERROR	Einfügen. Siehe <a href="#">DMS.XMLInsert</a> Keine Aktion durchführen Fehlermeldung erzeugen
1	Action1	NONE UPDATE [Default] INSERT	Keine Aktion durchführen Objekt aktualisieren [Default]. Siehe <a href="#">DMS.XMLUpdate</a> Einfügen am Standort des gefundenen Objektes. Siehe <a href="#">DMS.XMLInsert</a>
>1	ActionM	NONE UPDATE INSERT ERROR [DEFAULT]	Keine Aktion durchführen Nur das erste Objekt aktualisieren. Siehe <a href="#">DMS.XMLUpdate</a> Einfügen am Standort des ersten gefundenen Objektes. Siehe <a href="#">DMS.XMLInsert</a> Fehlermeldung erzeugen

Wird der Standort des Objektes vorgegeben oder eingeschränkt, wird dieser Standort sowohl für die Suche als auch zum Einfügen verwendet. Die Suche kann dabei auch zur Bestimmung des Standortes verwendet werden. Wird kein Standort vorgegeben und kein Objekt gefunden, so ist das Einfügen eines Registers oder Dokumentes nicht möglich. In diesem Fall wird eine Fehlermeldung generiert.

Werden keine Suchfelder angegeben, wird das Objekt eingefügt.

**Beispiel:**

Im folgenden Beispiel soll in einer Adresse die Telefonnummer eines Ansprechpartners geändert werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
  <Archive name="Adressen">
    <ObjectType name="Adressen">
      <Object>
        <Search>
          <Fields>
            <Field name="Ansprechperson:">Schaumer</Field>
            <Field name="Vorname:">Harald</Field>
          </Fields>
        </Search>
        <Fields>
          <Field name="Telefon:">0815-12345</Field>
        </Fields>
      </Object>
    </ObjectType>
  </Archive>
</DMSData>
```

**DMS.XMLUnknownToKnown****Beschreibung:**

Dieser Job ändert ein typenloses in ein Dokument mit Typ. Typenlose Dokumente befinden sich entweder in der Benutzerablage oder in der Workflowablage. Dazu sollte in dem Job entweder die Joboption 'INWFTRAY' oder 'INUSERTRAY' angeschaltet sein, zusätzlich sollte die Option 'ISTYPELESS' aktiviert sein. Der Typ zu dem das Dokument gemacht werden soll, wird in dem Tag 'ObjectType' festgelegt und die ID des typenlosen Dokumentes steht in dem 'Object' Tag als Attribut 'object\_id'.

#### Parameter:

Flags (INT): allgemeine Optionen für den Job (siehe [Flags](#))

Options (STRING): Semikolon-separierte Job-Optionen

(z. B. INWFTRAY=1;CHECKACCESS=0) (siehe [Options](#))

XML (BASE64): enthält Objektbeschreibung im XML-Format (siehe [Der Parameter XML](#))

JobUserGUID (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))

**Rückgabewerte:** keine jobspezifischen

#### Hinweis:

Die folgenden XML-Beispiele enthalten immer alle Tags und Tag-Attribute, die für die jeweilige Aktion verwendet werden können. Nicht benötigte Tags bzw. Attribute können natürlich weggelassen werden.

#### Beispiel:

XML für das Typisieren eines typenlosen Dokumentes

```
<?xml version="1.0" encoding="UTF-8"?>
<DMSData>
<Archive id="-1" name="Pressearchiv" internal_name="" osguid="">
<ObjectType table="" id="-1" name="Pressearchiv" internal_name=""
osguid="" type="FOLDER">
<Object object_id="54">
<Fields>
<Field dbname="" system="" name="Fachgebiet"
internal_name="" osguid="">Softwareentwicklung</Field>
<Field dbname="feld2" system="0" internal_name=""
osguid="">Testbenutzer</Field>
</Fields>
<TableFields>
<TableField dbname="Tabelle" system="0" name="" internal_name="">
<Row>
<Field name="Bezeichnung" internal_name=""
dbname="">Dokumentation</Field>
</Row>
</TableField>
</TableFields>
<Remarks>
<RemarkText action="INSERT" color="WHITE">Eine Notiz zum Ordner.</RemarkText>
<RemarkObject action="INSERT" object_id="123"
object_type="196616">Verknüpfungsnotiz</RemarkObject>
</Remarks>
</Object>
</ObjectType>
</Archive>
</DMSData>
```

## XML Export (Recherche)

Mit Hilfe der Exportfunktionalität des DMS Executors lassen sich Indexdaten und DMS Objektinformationen recherchieren.

- § [DMS.GetObjDef](#)
- § [DMS.GetObjectTypeByID](#)
- § [DMS.GetResultList](#)
- § [DMS.GetObjectDetails](#)
- § [DMS.GetDeletedObjects](#)
- § [DMS.GetLinkedObjects](#)
- § [DMS.SelectDistinctFieldValues](#)
- § [DMS.GetUserTrayObjects](#)
- § [DMS.ExecuteStoredQuery](#)
- § [DMS.GetStoredQuery](#)
- § [DMS.AddStoredQuery](#)
- § [DMS.UpdateStoredQuery](#)
- § [DMS.RemoveStoredQuery](#)
- § [DMS.ConvertQuery](#)
- § [DMS.GetObjectHistory](#)
- § [DMS.GetShadowData](#)
- § [DMS.GetObjectsByDigest](#)

## DMS.GetObjDef

### Beschreibung:

Dieser Job liefert alle Objektdefinitionen im XML-Format.

### Parameter:

Flags (INT): Optionen für diesen Job

- § 1 =Objektdefinition wird über Parameter sRet zurückgegeben, ansonsten als XML-Datei
- § 2 =Es werden die sichtbaren Schränke und Objekte ausgegeben, aber keine Feldinformationen. Die Sichtbarkeit wird dabei unabhängig vom Objektflag 'Sicherheitssystem ignorieren' überprüft.
- § 4 =Die Bilder der Objektdefinition werden BASE64 kodiert ausgegeben in der XML Rückgabe.
- § 4096 = das XML-Dokument wird UTF-8 kodiert, ansonsten UTF-16

### Rückgabewerte:

[FileCount] (INT): 1 = Ausgabe-Datei wurde erfolgreich erstellt, ansonsten 0

[sRet] (BASE64): Objektdefinition im XML-Format

[Dateiliste]: Name und Pfad der XML-Datei, die die Objektdefinition enthält

### Beispiel:

Objektdefinitionen im XML-Format

```

<?xml version="1.0" encoding="UTF-8"?>
<asobjdef created="" version="4.00">
<OS4x/>
<languages>
<language lang_id="" active="" name=""/>
</languages>
<cabinet cotype="" name="" internal="">
<object compressionflags="" historyflags="" maintype="" cotype=""
iconid="" name="" internal="" os_guid="" extablename="" tablename=""
no_dias="0" reference="0" fulltext="0" apply_security="0"
multidoc="0">
<names>
<name lang_id="" tooltip=""/>
</names>
<fields>
<field classstring="" name="" fieldname="" os_guid="" init=""
taborder="" tooltip="" prnalias="" internal="">
<names>
<name lang_id="" tooltip=""/>
</names>
<ids/>
<flags align="left" dt="A" flags="" flags1="" flags2=""
input_length="0" readonly="supervisor" multifield="0"
zeropad="0" control_type="edit" datatype="text"
constraints="none" required="0" crosscheck="0" color=""
case="0"/>
<field_pos bottom="0" left="0" right="0" top="0"/>
<input_pos bottom="0" left="0" right="0" top="0"/>
<init func="0" init_type="const"/>
<list addon32="" multiselection="0" order="0">
<rawdata/>
<extra/>
<row/>
</list>
<page/>
</field>
</fields>
<ids oid="" pid="" vid=""/>
<frame bottom="0" left="0" right="0" top="0"/>
<multiframe height="" width=""/>
</object>
</cabinet>
</asobjdef>

```

## DMS.GetObjectTypeByID

### Beschreibung:

Mit Hilfe dieses Jobs lässt sich der Typ einer Objektinstanz bestimmen.

### Parameter:

Flags (INT): reserviert. Muss 0 sein.

ObjectID (INT): ID der Objektinstanz

### Rückgabewerte:

ObjectType (INT): Typ des Objekts

Der Objekttyp setzt sich generell aus einem Haupt- (Highword) und einem Untertyp (Lowword) zusammen. Die folgende Tabelle gibt eine Übersicht über die unterstützten Objekttypen:

Objekttyp	Haupttyp	Untertyp
-----------	----------	----------

Ordner	0	>=0
Register	99	>=0
Dokument	1 = Graustufen 2 = S/W Modul 3 = Farbe 4 = Windows 5 = Multimedia 6 = e-Mail 7 = XML 8 = Container	>=0
Typenloses Dokument in der Benutzerablage	200	1-7 (s. Dokumentenhaupttyp)
Typenloses Dokument in der Workflowablage	300	1-7 (s. Dokumentenhaupttyp)
Mappe	203	0
Notiz	32767	1-4

## DMS.GetResultList

### Beschreibung:

Dieser Job sucht nach DMS-Objekten, die der Anfrage entsprechen. Die Festlegung der DMS-Zielobjekte und Felder geschieht über ein XML-Anfragedokument. In dem XML-Anfragedokument können OS-Namen, interne Namen, GUIDs und Datenbanknamen verwendet werden.

Über Attribute lassen sich allgemeine Eigenschaften der Anfrage setzen. Diese Attribute lassen sich sowohl als Jobparameter setzen, als auch als XML Attribute im Rootelement der XML Anfrage definieren, wobei die XML Attribute vorrangig behandelt werden.

### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

XML (BASE64): Anfrage im XML-Format (s. [Ausführliche Beschreibung](#))

[Encoding] (STRING): Kodierung des Ergebnisdokumentes. Zulässige Werte: UTF-8 und UTF-16. Default ist UTF-16.

[Offset] (INT): Offset für ersten zurückgegebenen Datensatz. Default=0 (s. [Blättern durch Trefferlisten](#))

[PageSize] (INT): Anzahl der Datensätze die zurückzugebenden werden sollen; -1 = Alle (Default) (s. [Blättern durch Trefferlisten](#))

[MaxHits] (INT): Maximal zu bestimmende Trefferzahl pro Objekttyp; -1 = Alle (Default), 0 = nicht ermitteln (Ein hoher Wert für MaxHits kann die Performance beeinträchtigen.) (s. [Blättern durch Trefferlisten](#))

[Sql] (INT): 1 = Sql-Statements werden in das Ergebnisdokument geschrieben (nur LOL); Default=0

- [Rights] (INT): 1 = Zusätzliche Zugriffsrechte (Öffnen/Löschen/Schreiben) werden für jede Objektinstanz ermittelt (HOL) Default=0 (s. [Rechte](#))
- [ObjectInserts] (INT): 1 = Anzahl der einfügbaren Objekte für jeden Objekttyp. Default=0 (s. [Rechte](#))
- [DateFormat] (STRING): gibt an, wie zurückgelieferte Datumsangaben formatiert werden sollen (s. [Datumsformate](#))
- [RegisterContext] (INT): 0 = Bei Angabe von Registerklauseln werden nur Objekte in Registern angefragt (HOL) 1=Default (s. [Allgemeines Anfrageverhalten](#))
- [ItemDelimiter] (STRING): Trennzeichen zwischen den Werten für lineare Liste als einfache Textdatei (s. [Ausgabeformate](#))
- [RequestType] (STRING): 'LOL', 'INF' oder 'HOL' möglich
- [OutputFormat] (STRING): In Abhängigkeit vom Anfragetyp (Requesttype) 'LOL', 'INF', 'TXT' oder 'HOL' möglich. (s. [Ausgabeformate](#))
- [Baseparams] (INT): 1 = Basisparameter werden zurückgeliefert, bei HOL als eigene XML Struktur (s. [Basisparameter](#))
- [FileInfo] (INT): 1 = Dateigröße in Bytes, Dateiendung und MIME Type von Dokumentendateien werden ermittelt (s. [Dateinformationen](#))
- [FollowDocLink] (INT): 0 = Wenn aktiviert, dann werden die Dateiinformatoren des verlinkten Dokuments (grüne Pfeil-Verweise) zurückgegeben. Diese Option wird nur ausgewertet, wenn [FileInfo=1] und kann die Performance beeinflussen.
- [Variants] (INT): 1 = Bei einem Dokument wird der Variantenbaum zurückgegeben wenn mehrere Varianten dieses Dokuments existieren (s. [Dokumentenvarianten](#))
- [Status] (INT): 1 = Objektstatus wird geliefert (dazu zählen Verknüpfungen und speziell für Dokumente: Modultyp, Auscheck- und Archivierungsstatus) (s. [Objektstatus](#))
- [CheckParams] (INT): 0 = Wurde für einen Anfrageparameter kein Wert, werden Bedingungen, welche diesen Parameter referenzieren, ignoriert. Dies ist der Defaultwert. 1 = Es wird eine Fehlermeldung erzeugt, wenn ein referenzierter Parameter nicht definiert wurde.
- [FieldSchema] (STRING): 'MIN' (nur Objekt ID), 'ALL' (alle Indexfelder) oder 'DEF' (benutzerdefiniert) möglich. Diese Einstellung kann innerhalb des XML Anfragedokumentes durch das 'field\_schema' Attribut im <Fields>, <ParentObjects> oder <ChildObjects> Element überschrieben werden.
- [AutoStar] (INT): Gibt an, ob bei Bedingungen auf Textfelder automatisch ein \* angehängt bzw. vorangestellt werden soll. 1=Stern vorne, 2=Stern hinten, 3=Stern hinten und vorne.
- [QueryLanguage] (INT): [Code](#) der Sprache, welcher für die DMS Namen in der Anfrage verwendet wird. Defaultsprache=0
- [OutputLanguage] (INT): [Code](#) der Sprache, welcher für die DMS Namen des Ergebnisdokumentes verwendet werden soll. Defaultsprache=0
- [DisableSearchGroups] (INT): Wird dieser Parameter auf 1 gesetzt, werden Suchgruppen nicht berücksichtigt, d.h. Suchbedingungen werden nur für das definierte Feld gesetzt. Wird der Parameter auf 0 gesetzt (Default), beziehen sich Suchbedingungen auf alle Felder einer zugehörigen Suchgruppe.
- [JobUserGUID] (STRING): bestimmt den Benutzerkontext (siehe [Der Parameter JobUserGUID](#))
- [GarbageMode] (INT): 1=Nur Objekte aus dem Papierkorb berücksichtigen. 0=Objekte aus dem Papierkorb nicht berücksichtigen.

**Rückgabewerte:**

Count (INT): Anzahl der Datensätze die zurückgegeben werden

TotalHits (INT): Anzahl der vorhandenen Treffer

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[XML] (BASE64): Trefferliste im XML-Format

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

**Ausführliche Beschreibung**

Eine Anfrage besteht im Wesentlichen aus der Festlegung des angefragten DMS-Objekts, der Auswahl der zurückzugebenden Felder und dem Setzen von Suchbedingungen. Darüber hinaus besteht z. B. die Möglichkeit, Standortinformationen anzufragen, Volltext- oder Basisparameterrecherchen durchzuführen, Verknüpfungen zu Objekten aus anderen Schränken zu erstellen oder die Anfragen zu parametrisieren.

§ [DMS – Anfragetypen](#)

§ [DMS-Ergebnisformate](#)

§ [Allgemeines Anfrageverhalten](#)

§ [Erstellen von Anfragen](#)

§ [Blättern durch Trefferlisten](#)

§ [DMS Referenz](#)

**DMS-Anfragetypen**

Bei Suchanfragen wird prinzipiell zwischen linearen und hierarchischen Anfragen unterschieden. Einen Sonderfall stellen die Detailanfragen dar.

**Lineare Anfragen / Lineare Objektlisten (LOL)**

Über lineare Anfragen können die einfachen Indexdaten von Objekten eines bestimmten Typs angefordert werden. Tabellensteuerelemente und Mehrfachparameterfelder sind davon ausgeschlossen. Werden Dokumente oder Register angefragt, können auch Felder des direkten Elternregisters und des Ordners angefragt werden.

**Hierarchische Anfragen / Hierarchische Objektlisten (HOL)**

Hierarchische Anfragen können nicht nur den Inhalt von Tabellensteuerelementen und Mehrfachparameterfeldern zurückgeben, sondern auch Kindelemente, also z. B. die Register und Dokumente innerhalb des angefragten Ordners. Zusätzlich können Objekteigenschaften wie z. B. Dokumentenvarianten, Zugriffsrechte und Notizen angefordert werden.

Neben den funktionellen Unterschieden zwischen linearen und hierarchischen Anfragen gibt es einen bedeutenden technischen Unterschied. Die Trefferlisten für lineare Anfragen können mit einem Satz von Datenbankabfragen erhalten werden. Bei hierarchischen Anfragen muss zusätzlich zu jedem Treffer, d.h. für jede Objektinstanz, mindestens eine weitere Datenbankabfrage durchgeführt werden. Jede angefragte Objekteigenschaft erhöht diese Zahl der Datenbankabfragen pro Objektinstanz.

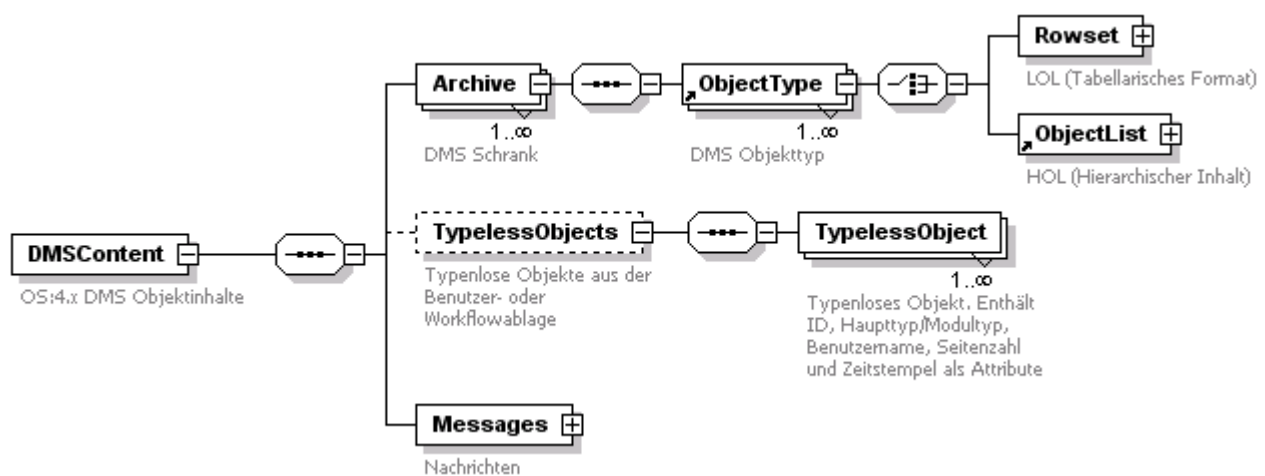
Kann auf Tabellensteuerelemente, Mehrfachparameter, Kindelemente und Zusatzeigenschaften (Dokumentenvarianten, Zugriffsrechte, Notizen) verzichtet werden, empfiehlt sich aus Performancegründen, immer eine lineare Anfrage zu stellen.

## Gemischte Anfragen (MIX)

Aufgrund der oben beschriebenen erheblichen Performanceunterschiede gibt es noch einen dritten Anfragetyp, der als gemischte Anfrage bezeichnet wird. Dabei handelt sich um eine Mischung zwischen linearer und hierarchischer Anfrage, in der komplexe Informationen zu Ordern oder Registern und einfache Informationen zu deren direkten Kindobjekten angefordert werden können. Zu den Kindobjekten wird pro Objekttyp eine Datenbankabfrage gestellt. Das Einsatzszenario entspricht z. B. dem Öffnen eines Ordners, um dessen vollständigen Indexdaten und Objekteigenschaften zu erhalten sowie die Übersicht aller Register und Dokumente, die sich in diesem Ordner befinden.

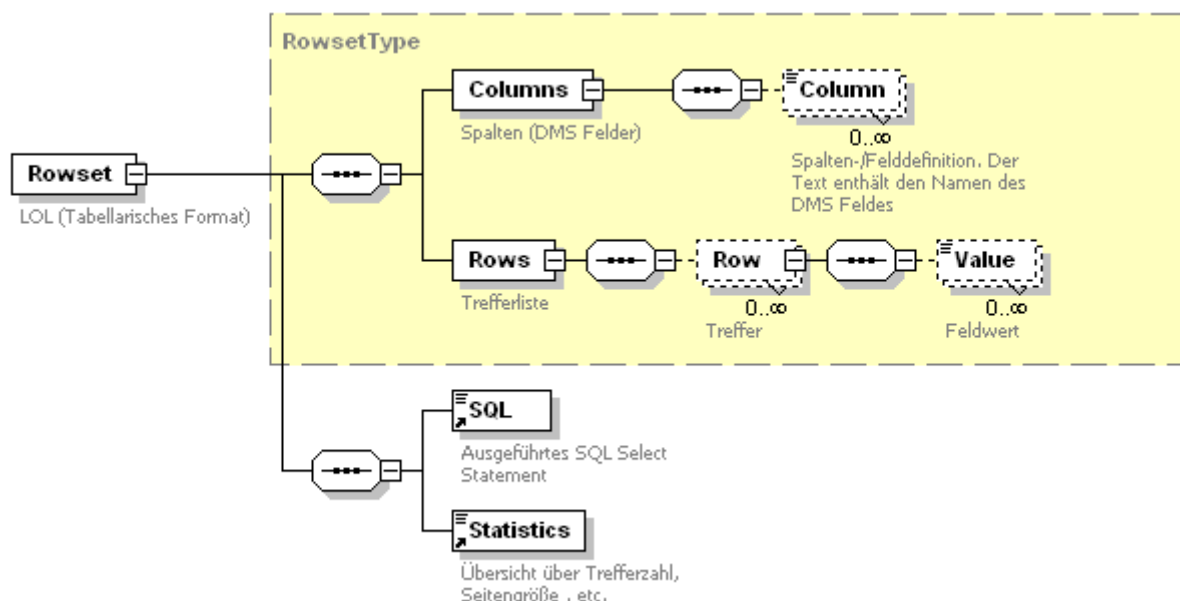
## DMS – Ergebnisformate

Das XML Format für die Anfrageergebnisse beginnt immer mit dem <DMSContent> Element. Dieses enthält ein Attribut 'format', welches die Werte LOL für lineare Objektliste, HOL für hierarchische Objektliste und 'MIXED' für gemischte Trefferliste annehmen kann.



## Lineare Objektliste (LOL)

Lineare Objektlisten werden durch das <Rowset> Element eingeleitet. Sie besitzen einen tabellarischen Aufbau, d.h. es wird zuerst der Tabellenkopf beschrieben (<Columns>) und darunter alle Trefferzeilen (<Rows>) aufgelistet.



## Beispiel:



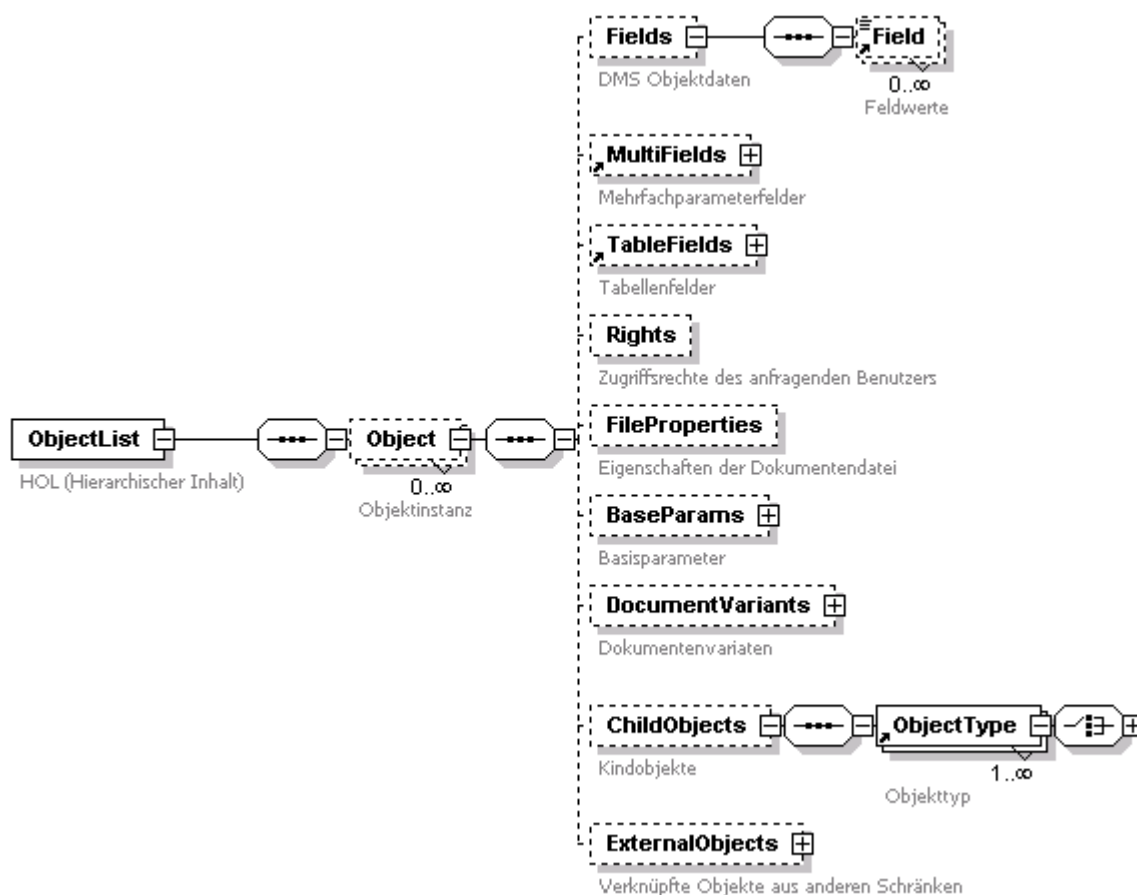
```

<?xml version="1.0" encoding="UTF-16" standalone="yes" ?>
<DMSContent format="LOL" version="4.50.582.4137" timestamp="2004-05-
24T15:18:19" user="ROOT" station="OSEPA">
<Archive name="Patient" id="5" osguid="04C7E64C981C456A9D1F5B12D188A752">
<ObjectType name="Patient" id="5" osguid="04C7E64C981C456A9D1F5B12D188A752"
type="FOLDER" table="stamm6">
<Rowset>
<Columns>
<Column object="Patient" type="FOLDER" name="Name" datatype="TEXT"
dbname="feld2" ostype="X" size="50">Name</Column>
<Column object="Patient" type="FOLDER" name="Vorname" datatype="TEXT"
dbname="feld3" ostype="X" size="50">Vorname</Column>
<Column object="Patient" type="FOLDER" name="Ort" datatype="TEXT"
dbname="feld10" ostype="X" size="50">Ort</Column>
</Columns>
<Rows>
D:\Testdaten\VirtPC\Queries\HOL\tmp0002.xml - #      <Row id="38503">
<Value>Abold</Value>
<Value>Beate</Value>
<Value>Berlin</Value>
</Row>
<Row id="38003">
<Value>Abold</Value>
<Value>Christina</Value>
<Value>Berlin</Value>
</Row>
</Rows>
</Rowset>
<Statistics startpos="0" pagesize="2" total_hits="530" />
</ObjectType>
</Archive>
<Messages/>
</DMSContent>

```

### Hierarchische Objektliste (HOL)

Hierarchische Trefferlisten werden durch das <ObjectList> Element eingeleitet. Für jedes Trefferobjekt werden Felddefinition und Feldwert geschrieben. Hierarchische Trefferlisten können über das <ChildObjects> Element beliebige Hierarchietiefen abbilden.



### Beispiel:

#### Hierarchische Objektliste

```

<DMSContent format="HOL" version="4.50.582.4137" timestamp="2004-05-
24T15:29:03" user="ROOT" station="OSEPA">
  <Archive name="Patient" id="5" osguid="04C7E64C981C456A1F5B12D188A752">
    <!--Ordner -->
    <ObjectType name="Patient" id="5"
    osguid="04C7E64C981C456A9D1F5B12D188A752" type="FOLDER" table="stamm6">
      <!--Ordnerliste -->
      <ObjectList>
        <Object id="37751">
          <Fields>
            <Field name="PatientenID" datatype="TEXT" dbname="feld1" ostype="X"
            size="20">777777</Field>
            <Field name="Name" datatype="TEXT" dbname="feld2" ostype="X"
            size="50">Sandmann</Field>
            <Field name="Vorname" datatype="TEXT" dbname="feld3" ostype="X"
            size="50">Sandor</Field>
          </Fields>
          <!--Kindobjekte des Ordners à
          <ChildObjects>
            <!--Register -->
            <ObjectType name="Aufenthalt" id="6488064"
            osguid="A90F043941488DB43B69CBDA" type="REGISTER" table="register1">
              <ObjectList>
                <Object id="3226">
                  <Fields>
                    <Field name="Fallnummer" datatype="TEXT" dbname="feld62" ostype="X"
                    size="20">987654</Field>
  
```

```

<Field name="Beginn" datatype="DATE" dbname="datum1" ostyle="D"
size="10">01.10.2003</Field>
<Field name="Ende" datatype="DATE" dbname="datum2" ostyle="D"
size="10" />
</Fields>
<!-- Kindobjekte des Registers -->
<ChildObjects>
<!--Dokumenttyp 'Arztbrief' -->
<ObjectType name="Arztbrief" id="262273" osguid="14799BCD39A920AF3"
type="DOCUMENT" modul="WINDOWS" table="object162">
<ObjectList>
<Object id="37744">
<Fields>
<Field name="Datum" datatype="DATE" dbname="datum1" ostyle="D"
size="10">12.11.2003</Field>
<Field name="erstell. Arzt" datatype="TEXT" dbname="feld1"
ostype="X" size="50">DEMO</Field>
<Field name="Typ" datatype="TEXT" dbname="feld3" ostyle="X"
size="20">Arztbrief operativ</Field>
<Field name="Status" datatype="INTEGER" dbname="zahl2"
ostype="9" size="1" />
</Fields>
</Object>
</ObjectList>
<Statistics startpos="0" pagesize="5" total_hits="1" />
</ObjectType>
</ChildObjects>
</Object>
</ObjectList>
<Statistics startpos="0" pagesize="5" total_hits="1" />
</ObjectType>
</ChildObjects>
</Object>
</ObjectList>
<Statistics startpos="0" pagesize="5" total_hits="1" />
</ObjectType>
</Archive>
<Messages />
</DMSContent>

```

## Darstellung der Feldwerte

Im Allgemeinen werden Feldwerte als XML Elementtext ausgegeben. Für einige Feldtypen gibt es aber Unterschiede zwischen dem Wert in der Datenbank und dem angezeigten Wert, dazu zählen:

- § Zeit
- § Zeitstempel
- § Optionsschaltflächen (Radiobuttons)
- § Systemfeld Besitzer
- § Bäume
- § ggf. Listen
- § Spezialwerte wie: Richtung , Frage, ...

Für diese Feldtypen wird der Datenbankwert als Wert eines 'value' Attributes zurückgegeben, während der Anzeigetext im XML Elementtext enthalten ist.

Beispiel für LOL:

```
<Value value="M">männlich</Value>
```

Beispiel für HOL:

```
<Field value="M" name="Geschlecht" datatype="TEXT" dbname="feld7"
ostype="X" size="1">männlich</Field>
```

### Das <Statistics> Element

Am Ende jeder Trefferliste findet sich das <Statistics> Element, welches Angaben über den Ergebnisumfang enthält. Diese Informationen sind insbesondere zum [Blättern durch Trefferlisten](#) wichtig.

```
<Statistics startpos="20" pagesize="20" total_hits="50" />
```

### Das <Messages> Element

Informationen über fehlerhafte Anfragen finden sich in den <Message> Elementen.

**Beispiel:**

```
<Messages>
<Message faultcode="-2116351928" sourcecode="475">Die Volltextanfrage konnte nicht
durchgeführt werden, da kein Suchtext angegeben wurde.
</Message>
</Messages>
```

### Kombination von Anfragetypen und Ausgabeformaten

Das Ausgabeformat für lineare Anfragen ist standardmäßig das oben beschriebene LOL-XML-Format. Es ist aber auch möglich, lineare Trefferlisten in einem einfachen Textformat oder im HOL-XML-Format anzufordern. Dies geschieht über den Parameter 'OutputFormat'.

Im Textformat werden die Werte durch ein über den Parameter 'ItemDelimiter' definierbares Trennzeichen voneinander getrennt.

		Ausgabeformat			
		TXT	LOL	MIX	HOL
Anfragetyp	LOL	+	+	-	+
	MIX		-	+	+
	HOL		-	-	+

### Allgemeines Anfrageverhalten

In einer Suchanfrage können Suchbedingungen für ein oder mehrere Objekttypen formuliert werden. Es können z. B. bei einer Dokumentenanfrage sowohl Bedingungen für ein Register und Ordner festgelegt werden.

Für den enaio® client wurde ein besonderes Anfrageverhalten definiert. Enthalten Anfragen Registerbedingungen – egal ob Anwender oder aus dem Rechtesystem – ist das Anfrageverhalten laut 'Anforderungen für Anfragen in enaio® client' wie folgt definiert:

Für Dokumente gilt die Regel:

Es sind alle Dokumente zu liefern, die sich in einem Register befinden, das den Suchkriterien entspricht und zusätzlich alle Dokumente, die sich in keinem Register befinden.

Für Ordner gilt die Regel:

Es sind alle Ordner zu liefern, in denen sich ein Register befindet, das den Suchkriterien entspricht und alle zusätzlich alle Ordner, die kein Register enthalten.

Im Job GetResultList lässt sich dieses Verhalten über den Jobparameter 'RegisterContext' bzw. das XML Attribut 'registercontext' steuern. Ist der Parameter nicht angegeben oder er hat den Wert 1 ist dieses Suchverhalten aktiv, hat der Parameter den Wert 0 ist diese Funktion abgeschaltet.

### Erstellen von Anfragen

Zur Erstellung einer Anfrage müssen zunächst die anzufragenden DMS Objekte und Felder definiert werden. Die DMS Objekte und Felder können identifiziert werden über:

- § ihren sprechenden Namen (Attribut 'name')
- § den internen Namen (Attribut 'internal\_name')
- § ihre OSGUID (Attribut 'osguid')
- § ihre Datenbanktabellen- oder Spaltennamen (Attribut 'table' bzw. 'dbname')

### Beispiel für eine einfache Anfrage

Mit folgender Anfrage werden alle Ordner des Schrankes 'Patient' angefordert. Da Objektnamen innerhalb eines Schrankes nicht eindeutig sein müssen – z. B. kann ein Dokumenttyp so heißen wie der Ordnerartyp – kann man den Objekttyp über das Attribut 'type' festlegen. Gültige Werte sind 'FOLDER', 'REGISTER' und 'DOCUMENT'.

Über die Zuweisung des Wertes 'ALL' zum <Fields> Attribut 'field\_schema' wird der DMS Executor angewiesen, alle DMS Felder des angefragten Objektes – also hier des Patientenordners – zurückzugeben.

```
<?xml version="1.0" encoding="UTF-8" ?>
<DMSQuery>
  <Archive name="Patient">
    <ObjectType name="Patient" type="FOLDER" >
      <Fields field_schema="ALL"/>
    </ObjectType>
  </Archive>
</DMSQuery>
```

### Felder definieren

Werden nicht alle Indexdaten benötigt, lassen sich die gewünschten Felder auch explizit festlegen. Dazu setzt man für das Feldschema Attribut 'field\_schema' den Wert 'DEF'.

```
<?xml version="1.0" encoding="UTF-8" ?>
<DMSQuery>
  <Archive name="Patient">
    <ObjectType name="Patient" type="FOLDER" >
      <Fields field_schema="DEF">
        <Field name="Name"/>
        <Field name="Vorname"/>
        <Field name="PLZ / Wohnort"/>
        <Field name="Ort"/>
      </Fields>
    </ObjectType>
  </Archive>
</DMSQuery>
```

## Optionsschaltflächen

Eine Sonderrolle nehmen Optionsschaltflächen ein, da es sich hierbei um eine Gruppe von Feldern handelt. Besitzt eine solche Gruppe ein statisches Gruppenfeld, lässt sich dieses als Recherchefeld verwenden. Ansonsten kann jedes einzelne Auswahlfeld als Stellvertreter der Gruppe ausgewählt werden.

## Systemfelder

Über das <Field> Element lassen sich auch Systemfelder anfragen. Dazu ist das Attribut 'system' auf '1' zu setzen. Systemfelder lassen sich über einen internen Namen, eine GUID oder über ihren Datenbankfeldnamen angeben. Eine Übersicht über die zulässigen Systemfelder findet sich in der [DMS Referenz](#). Im folgenden Beispiel wird zusätzlich zu allen Indexdaten des Objektes 'Bilder' die Anzahl der Dokumentdateien angefragt.

```
<?xml version="1.0" encoding="UTF-8" ?>
<DMSQuery>
  <Archive name="Patient">
    <ObjectType name="Bilder" type="DOCUMENT" >
      <Fields field_schema="ALL">
        <Field internal_name="OBJECT_COUNT" system="1" />
      </Fields>
    </ObjectType>
  </Archive>
</DMSQuery>
```

## Sortierung

Um die Trefferlisten einer LOL Anfrage nach bestimmten Feldern zu sortieren, muss das Attribut 'sortpos' mit einem Wert größer 0 angegeben werden. Zusätzlich kann über das Attribut 'sortorder' mit dem Wert ASC eine aufsteigende bzw. mit dem Wert DESC eine absteigende Sortierung gewählt werden. Wird das Attribut 'sortorder' nicht angegeben, wird aufsteigend sortiert, es sei denn, ein Feld mit höherer Sortierpriorität hat das Attribut sortorder explizit gesetzt. In diesem Fall wird der Wert für die folgenden Felder ohne das Attribut übernommen.

Besitzen mehrere Felder das Attribut 'sortpos', so haben Felder mit niedrigerem 'sortpos'-Wert eine höhere Priorität.

Mit der folgenden Anfrage werden alle Register vom Typ 'Aufnahme' aus dem Schrank 'Patient' angefordert. Dabei werden alle Indexdaten sowie die Dokumentendateianzahl angefordert. Sortiert werden die Treffer aufsteigend nach dem Feld 'Autor' und anschließend absteigend nach 'Datum'.

```
<?xml version="1.0" encoding="UTF-8" ?>
<DMSQuery>
  <Archive name="Patient">
    <ObjectType name="Aufnahme" type="REGISTER" >
      <Fields field_schema="ALL">
        <Field name="Autor" sortpos="1" sortorder="ASC" />
        <Field name="Datum" sortpos="2" sortorder="DESC" />
      </Fields>
    </ObjectType>
  </Archive>
</DMSQuery>
```

Für hierarchische Anfragen ist eine Sortierung der Ausgabe nicht möglich

## Suchbedingungen

Bedingungen können nicht nur für das angefragte Objekt formuliert werden, sondern auch auf Eltern- und Kindobjekte. Daher muss zunächst das Klauselobjekt (Attribut ConditionObject), für das die

Suchbedingen gelten sollen, festgelegt werden. Es können in einer Anfrage Suchbedingen für mehrere Objekte gleichzeitig definiert werden.

Im folgenden Beispiel werden alle Patienten angefragt, die am 1.1.2004 auf Station 1 aufgenommen wurden und größer als 180 cm sind:

```
<?xml version="1.0" encoding="UTF-16" ?>
<DMSQuery>
<Archive name="Patient">
<ObjectType name="Patient" type="FOLDER">
<Fields field_schema="DEF">
<Field name="Name" />
<Field name="Vorname" />
<Field name="PLZ / Wohnort" />
<Field name="Ort" />
</Fields>
<Conditions>
<ConditionObject name="Aufenthalt" type="REGISTER">
<FieldCondition name="Station" operator="=">
<Value>1</Value>
</FieldCondition>
<FieldCondition name="Beginn">
<Value>1.1.2004</Value>
</FieldCondition>
</ConditionObject>
<ConditionObject name="Körpermaße" type="DOCUMENT">
<FieldCondition name="Körperhöhe [cm]" operator=">">
<Value>180</Value>
</FieldCondition>
</ConditionObject>
</Conditions>
</ObjectType>
</Archive>
</DMSQuery>
```

Wie in diesem Beispiel zu sehen ist, muss der '>' Operator (Attribut operator='>') kodiert werden, damit das Dokument dem XML Format entspricht. Wird in einer Bedingung kein Operator angegeben, wird der Operator '=' verwendet. Folgende Vergleichsoperatoren sind zulässig:

### Vergleichsoperatoren

Operator	XML konformes Format
<	&lt;
<=	&lt;=
=	=
!=	!=
>	&gt;
>=	&gt;=
BETWEEN	BETWEEN
NOT BETWEEN	NOT BETWEEN
IN	IN
NOT IN	NOT IN

## Platzhalter

Bei Anfragen auf Textfelder können die gleichen Platzhalter wie in enaio® client verwendet werden:

Platzhalter	Bedeutung
*	beliebige Zeichenfolge
?	Einzelnes Zeichen
~	Phonetische Suche (nur MSSQL und Oracle)

Als Escapezeichen dient der Backslash, d. h. mit \? würde nach einem Fragezeichen gesucht werden.

## Null wert

Zur Prüfung eines Werte auf NULL steht das <NULL/> Element zur Verfügung, dass anstelle des <Value> Elementes verwendet werden muss.

## Spezialwerte

Über das Element <SpecialValue> lassen sich Spezialwerte abbilden. Folgende Werte stehen dafür zur Verfügung:

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

## Basisparameter-Eigenschaften von Dokumenten

Basisparameter-Eigenschaften von Dokumenten können als Suchbedingung über 'OBJEKT\_SEARCHFLAGS' angegeben werden.

OBJEKT\_SEARCHFLAGS können über Bitwerte einzeln oder kombiniert angegeben werden.

Für Kombinationen gelten folgende Regeln:

- § Werte von Eigenschaften der gleichen Gruppe werden logisch durch ODER kombiniert.
- § Werte von Eigenschaften unterschiedlicher Gruppen und Werte ohne Gruppe werden logisch durch UND kombiniert.

Eigenschaften, Werte und Gruppen:

archiviert	1	Gruppe 1
------------	---	----------



zur Archivierung freigegeben	2	Gruppe 1
nicht zur Archivierung freigegeben	4	Gruppe 1
ohne Seiten	8	Gruppe 1
von mir ausgecheckt	16	Gruppe 2
von anderen ausgecheckt	32	Gruppe 2
im Register	64	Gruppe 3
in keinem Register	128	Gruppe 3
ausgelagert	256	Gruppe 2
Verweisdokument	512	Gruppe 1
mehrere Standorte	1024	-
mit Varianten	2048	-
signiert in aktueller Version	4096	Gruppe 1
signiert in vorheriger Version	8192	Gruppe 1

Beispiel:

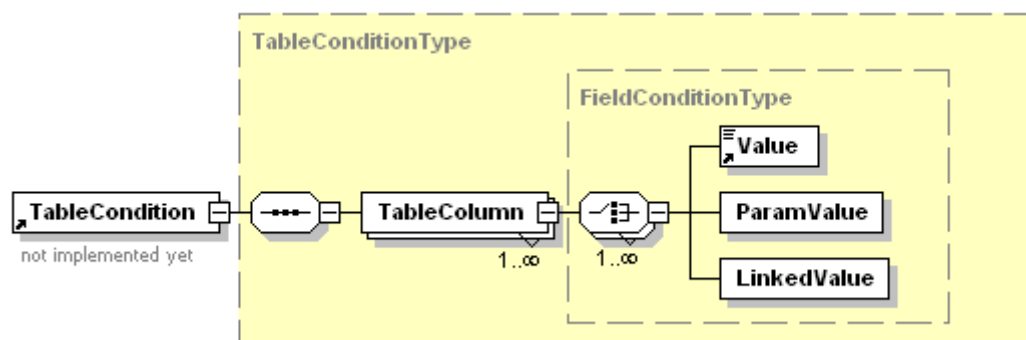
```
<Conditions>
<ConditionObject name="Document1" type="DOCUMENT">
<FieldCondition name="OBJECT_SEARCHFLAGS" system="1">
<Value>68</Value>
</FieldCondition>
</ConditionObject>
</Conditions>
```

Mit dem Wert '68' werden Dokumente mit der Eigenschaft 'im Register' (64) UND der Eigenschaft 'nicht zur Archivierung freigegeben' (4) gesucht.

## Besonderheiten der DMS-Feldtypen

### Suchbedingungen für Tabellenfelder

Dialogelemente vom Typ 'Tabelle' bestehen selbst aus einer oder mehreren Spalten, die einzeln recherchierbar sind. Im XML Anfrageformat ist dies folgendermaßen abgebildet:



Anstelle eines <FieldCondition> Elementes wird ein <TableCondition> Element angegeben, mit dem die Tabelle selbst referenziert wird. Über das <TableColumn> Element wird die Tabellenspalte definiert, für die das Suchkriterium formuliert werden soll.

Beispiel:

```

<DMSQuery>
<Archive name="Testschrank">
<ObjectType name="WinDoc">
<Fields field_schema="ALL" />
<Conditions>
<ConditionObject name="WinDoc">
<TableCondition name="MeineTabelle">
<TableColumn name="1.Spalte" operator="=">
<Value>11</Value>
</TableColumn>
</TableCondition>
</ConditionObject>
</Conditions>
</ObjectType>
</Archive>
</DMSQuery>

```

### Liste mit Mehrfachauswahl

Bei Listen mit Mehrfachauswahl wird grundsätzlich der Platzhalter '\*' an Anfang und Ende des Suchwortes gehängt.

### Datum

Das Datum kann prinzipiell in allen gängigen Formaten angegeben werden. Für zweistellige Jahreszahlen gilt: wird eine Jahreszahl xy kleiner als 50 angegeben, wird daraus die Jahreszahl 20xy erstellt. Ist die Jahreszahl xy kleiner oder gleich 50 wird daraus die Jahreszahl 19xy erstellt.

Beispiel:

04	führt zu	2004
76	führt zu	1976

Bei Verwendung des Gleichheitsoperators für ein unvollständiges Datum, wird über den entsprechenden Zeitbereich gesucht.

Beispiel:

=1999	führt zu	BETWEEN 1.1.1999 AND 31.12.1999
!= 1999	führt zu	NOT BETWEEN 1.1.1999 AND 31.12.1999

Werden zwei Werte (über zwei <Value> Elemente) innerhalb einer Datumsbedingung angegeben, so werden diese automatisch als Bereich behandelt und ein 'BETWEEN' bzw. ein 'NOT BETWEEN' im SQL Statement verwendet.

### Text

Es wird automatisch der LIKE Operator verwendet, wenn Platzhalter verwendet werden.

### Optionsschaltflächen

Soll ein Suchkriterium auf eine Radiobuttongruppe gesetzt werden, so kann sowohl ein evtl. vorhandenes Gruppenfeld als auch ein beliebiges Radiobuttonfeld der Gruppe verwendet werden, um das Suchkriterium festzulegen.

### Boolsche (UND/ODER) Verknüpfungen von Bedingungen

Suchbedingungen werden standardmäßig mit einem logischen UND verknüpft. Es besteht aber auch die Möglichkeit, Suchbedingungen mit ODER bzw. beliebige Kombinationen von UND und ODER zu verknüpfen. Dazu müssen die einzelnen Feldbedingungen über das <FieldGroup> Element gruppiert werden. Feldgruppen können dabei wiederum Feldgruppen enthalten.

```

<ConditionObject name="Pressearchiv">

```

```

<FieldGroup operator="OR">
  <FieldCondition name="Fachgebiet">
    <Value>Technik</Value>
  </FieldCondition>
  <FieldCondition name="Erstellt von:">
    <Value>Schmitz</Value>
  </FieldCondition>
</FieldGroup>
<Created>
  <From>1.1.2002</From>
  <To>31.12.2003</To>
</Created>
</ConditionObject>

```

### Verknüpfung der Bedingungen bei mehreren <ConditionObject> Elementen

Wie bereits erwähnt, lassen sich innerhalb einer Anfrage Bedingungen auf mehrere Objekttypen definieren. Beziehen sich mehrere <ConditionObject> Elemente auf den gleichen Objekttyp so werden die darin jeweils enthaltenen Bedingungsgruppen mit ODER verknüpft.

### Volltextbedingungen

Über das <Fulltext> Element kann eine Volltextanfrage durchgeführt werden. Volltextklauseln können als Text des Elementes <Fulltext> geschrieben werden.

### Standortinformationen

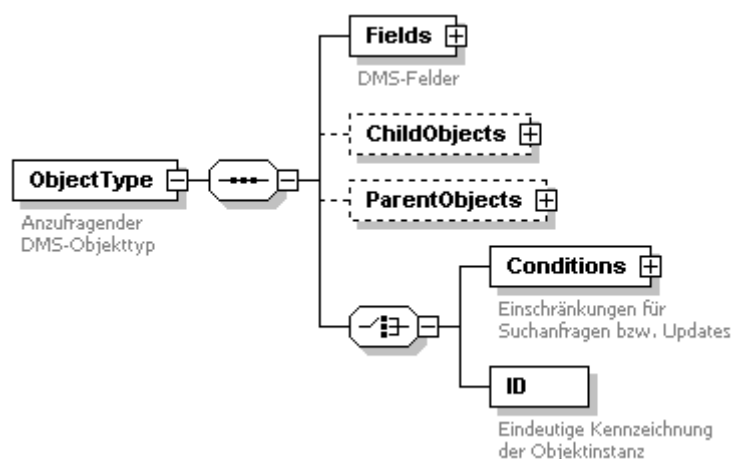
Bei Recherchen nach Dokumenten oder Registern ist es oft wünschenswert, Informationen über den Standort, d.h. Ordner oder das Elternregister zu erhalten. Dazu legt man im <ParentObjects> Element fest, in welchem Umfang Informationen über die Elternobjekte angefragt werden sollen.

Für lineare Anfragen lassen sich maximal ein Register und der Ordner angeben. Ordner-, Register- und Objektfelder erscheinen in einer Trefferzeile.

Für hierarchische Anfragen wird bei Vorhandensein des <ParentsObjects> Element der gesamte Objekttyppfad bestimmt. Über die <SubObjectType> Elemente lassen sich zu den einzelnen Registern und zum Ordner die Felder festlegen.

### Export von hierarchischen Strukturen

Hierarchische Anfragen unterscheiden sich von linearen Anfragen vor allem dadurch, dass ganze Objektstrukturen exportiert werden können. Dazu wird zunächst die Suche wie beschrieben nach dem übergeordneten Objekt formuliert. Als Unterelement erhält die Anfrage nun das Element <ChildObjects>. Dieses besitzt die Attribute 'export\_depth' und 'child\_schema'.



## Angabe der Exporttiefe

Mit dem Attribut 'export\_depth' wird die Exporttiefe festgelegt, d.h. die Anzahl der zu exportierenden Objektebenen. 0 würde keine Kindobjekte exportieren, 1 nur die direkten Kindobjekte, etc.

## Angabe des Objektschemas

Für das Attribut 'child\_schema' stehen folgende Werte zur Verfügung:

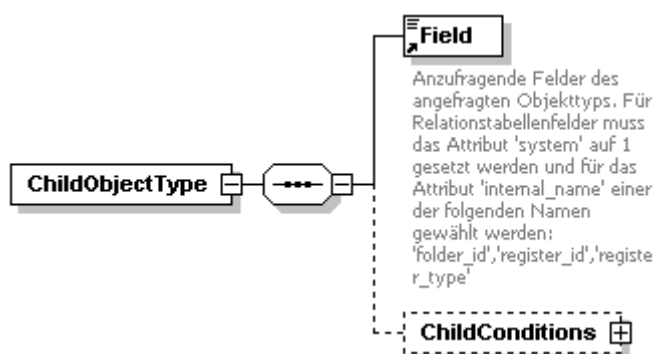
Wert	Bedeutung
DEF	Benutzerdefiniert
REG	Alle Register werden automatisch hinzugefügt
DOC	Alle Dokumente werden automatisch hinzugefügt
ALL	Alle Register und Dokumente werden automatisch hinzugefügt

## Angabe der Kindobjekttypen

Über das <ChildObjectType> Element lassen sich einzelne Kindobjekttypen festlegen. Innerhalb der <ChildObjectType> Elemente können Felder nach den gleichen Regeln wie für das Hauptanfrageobjekt festgelegt werden.

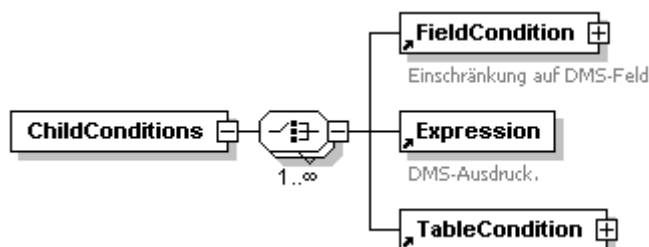
Hinweis:

Ein Kindobjekttyp darf nur einmal in der Kindobjekttypliste auftauchen.



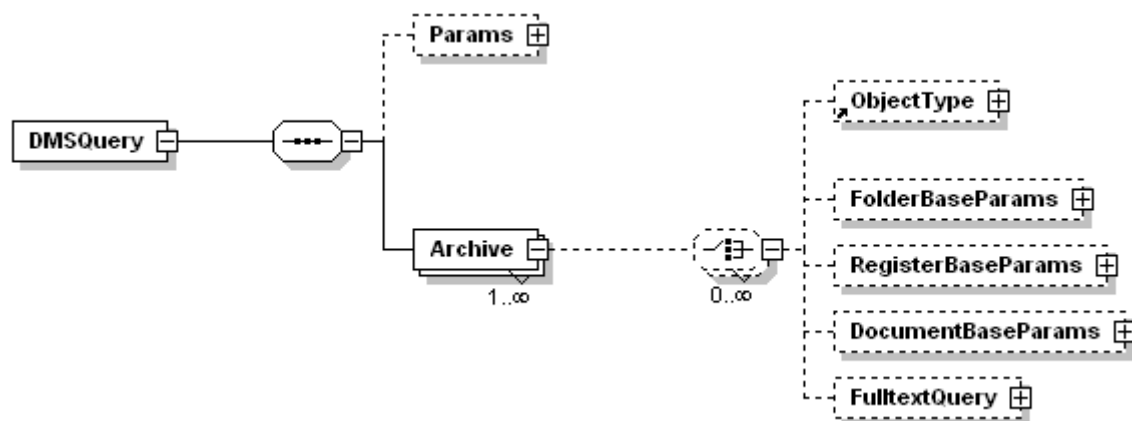
## Einschränkungen für Kindobjekte

Die Auswahl der Kindobjekte lässt sich über zusätzliche Bedingungen weiter einschränken.



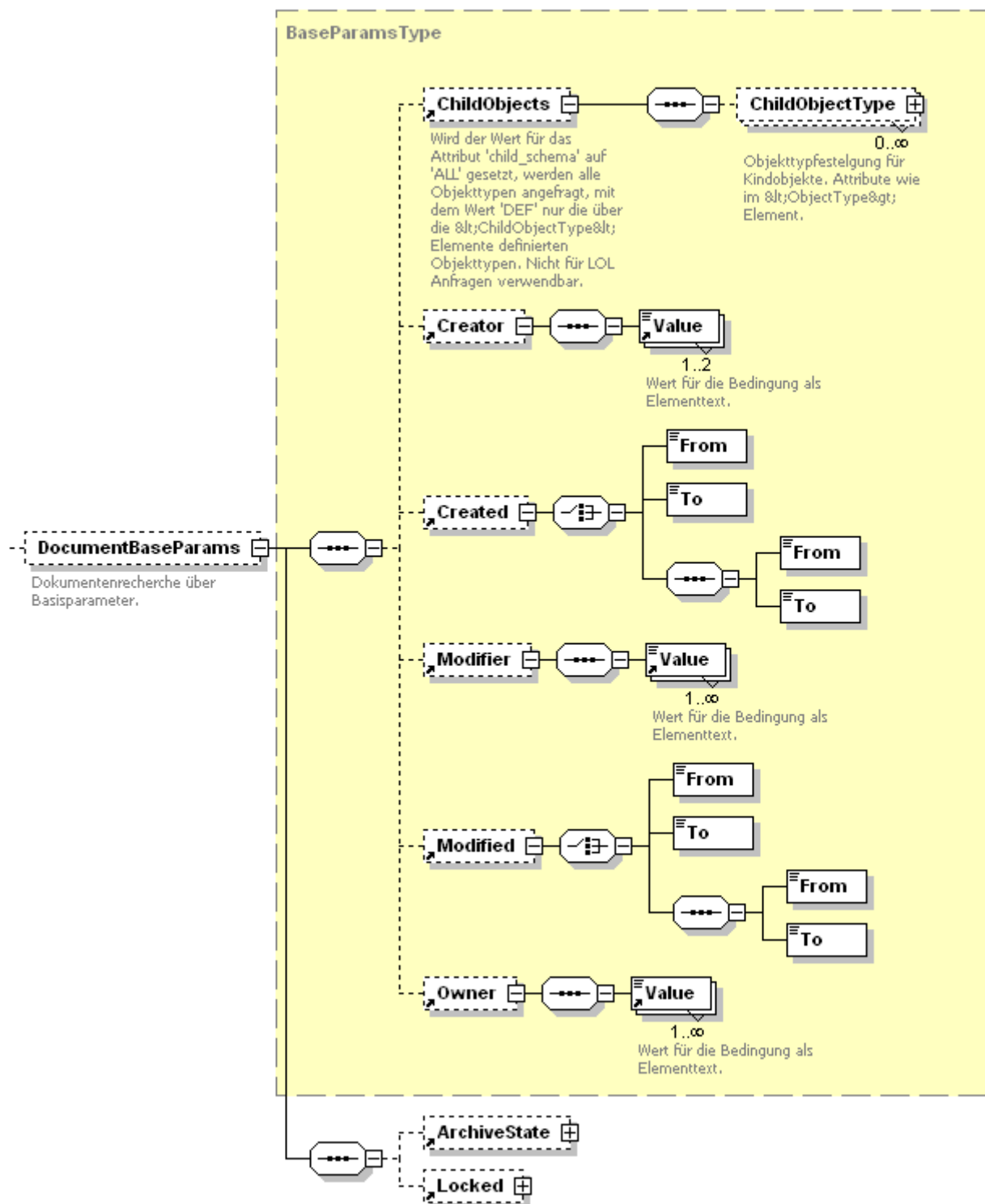
## Basisparameterrecherchen

Es ist möglich, Ordner, Register und Dokumente über die Basisparameter zu suchen. Nach der Festlegung des Schranke über das <Archive> Element, kann man die Details für eine Ordner-, Register-, oder Dokumentenrecherche über Basisparameter festlegen.



Innerhalb des entsprechenden Elementes (<FolderBaseParams>, <RegisterBaseParams> oder <DocumentBaseParams>) lassen sich nun folgende Suchbedingungen für die Basisparameter definieren:

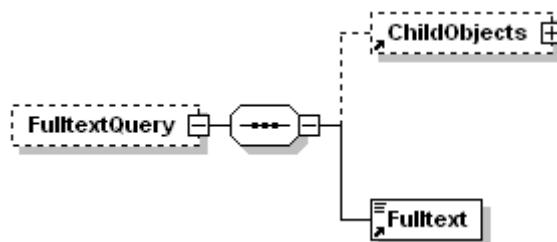
XML Element	Bedeutung
<Creator>	Ersteller
<Created>	Erstellungsdatum
<Modifier>	Benutzer, der das Objekt zuletzt bearbeitet hat
<Modified>	Datum der letzten Änderung. Werden zwei Werte angegeben, so wird über den Zeitraum zwischen diesen Daten gesucht
<Owner>	Benutzername des Besitzers
<ArchiveState><ArchiveStateValue>	Für Dokumente: Archivierungsstatus. Zulässige Werte: ARCHIVED (archiviert), ARCHIVABLE, (archivierbar), NOT_ARCHIVABLE (nicht archivierbar), NO_PAGES (keine Seiten), PAGE_ERROR (Fehlerhafte Seite), REFERENCE (Verweis)
<Locked><LockStateValue>	Für Dokumente: Auscheckstatus. Zulässige Werte: UNLOCKED (nicht ausgecheckt), SELF (vom anfragenden Benutzer selbst ausgecheckt), OTHERS (von einem anderen Benutzer ausgecheckt), EXTERNAL (ausgelagert)



Für Dokumente und Registeranfragen lässt sich außerdem über das **<ChildObjects>** Element die Auswahl der Objekttypen einschränken, oder für Ordner die Feldauswahl und die Sortierreihenfolge in der Trefferliste festlegen.

### Volltextrecherchen

Ist die Volltextindexierung in enaio® eingerichtet, lässt sich über alle in enaio® editor entsprechend konfigurierten Objekttypen eine Volltextsuche durchführen. Dazu wird nach Festlegung des Schranke über das **<Archive>** Elementes, innerhalb des **<FulltextQuery>** Elementes die Volltextklausel über das **<Fulltext>** angegeben.



Über das optionale <ChildObjects> Element lässt sich die Auswahl der Objekttypen einschränken oder die Feldauswahl und die Sortierreihenfolge der Objekte in der Trefferliste festlegen.

### Beispiel:

Volltextsuche nach dem Wort 'Meningitis' in allen Arztbriefen und Diagnosen.

```

<?xml version="1.0" encoding="UTF-8" ?>
<DMSQuery>
  <Archive name="Patient">
    <FulltextQuery>
      <ChildObjects child_schema="DEF">
        <ChildObjectType name="Arztbrief">
          <Fields>
            <Field name="Datum" />
            <Field name="Oberarzt" />
            <Field name="Typ" />
          </Fields>
        </ChildObjectType>
        <ChildObjectType name="Diagnose">
          <Fields field_schema="ALL" />
        </ChildObjectType>
      </ChildObjects>
      <Fulltext>Meningitis</Fulltext>
    </FulltextQuery>
  </Archive>
</DMSQuery>
  
```

### Parametrisierung von Anfragen

Zur Erleichterung der Wiederverwendung von Anfragen besteht die Möglichkeit, Suchbedingungen zu parametrisieren. Dazu werden unterhalb des <DMSQuery> Elementes alle Parameter definiert und initialisiert. In den Bedingungen der Anfrage können diese Parameterwerte über ihren Parameternamen mit Hilfe des 'ref' Attributes im <ParamValue> Element referenziert werden.

### Beispiel:

```

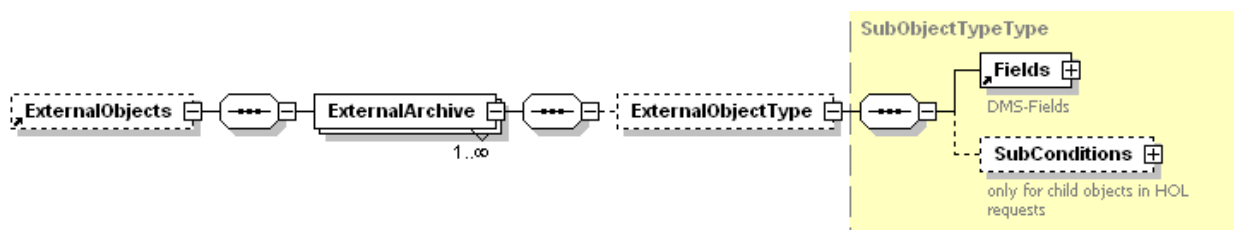
<DMSQuery>
  <Params>
    <Param name="PatID">3987</Param>
  </Params>
  <Archive name="Patient">
    <ObjectType name="Patient">
      <Fields field_schema="ALL" />
      <Conditions>
        <ConditionObject name="Patient">
          <FieldCondition name="PatientenID">
            <ParamValue ref="PatID" />
          </FieldCondition>
        </ConditionObject>
      </Conditions>
    </ObjectType>
  </Archive>
  
```

```
</DMSQuery>
```

Eine Clientanwendung kann auf diese Weise ohne genauere Kenntnis der Anfragestruktur den Wert für die Bedingung über den Parameter ändern.

### Schrankübergreifende Anfragen

Werden in einer Trefferliste zu einem Objekt Informationen benötigt, die in einem Objekt aus einem anderen Schrank enthalten sind, lässt sich dieses in einer HOL Anfrage über das <ExternalObjects> Element formulieren



Das <ExternalObjects> Element wird unterhalb des <ObjectType> Elementes definiert. Die Verknüpfung eines externen Objekts mit dem Ausgangsobjekt geschieht über eine Feldreferenz. Dazu wird bei dem Ausgangsobjektfeld, das den Wert für die Verknüpfung liefern soll, über das Attribut 'link\_name' ein beliebiger Aliasname definiert. In der Felddingung des externen Objektes wird dieser Aliasname über das Attribut 'ref' im Element <LinkedValue> referenziert.

### Beispiel:

Zu jedem S/W Dokument soll die Adresse des Autors mitgeliefert werden:

```
<DMSQuery>
<Archive name="Pressearchiv">
<ObjectType name="S/W-Scans">
<Fields>
<Field name="Dokumentart" />
<Field name="Autor" link_name="Autor" />
<Field name="Datum" />
</Fields>
<ExternalObjects>
<ExternalArchive name="Adressen">
<ExternalObjectType name="Adressen">
<Fields field_schema="ALL" />
<SubConditions>
<FieldCondition name="Name:">
<LinkedValue ref="Autor" />
</FieldCondition>
</SubConditions>
</ExternalObjectType>
</ExternalArchive>
</ExternalObjects>
</ObjectType>
</Archive>
</DMSQuery>
```

### Erweiterte Informationen

#### Basisparameter

Durch Setzen des Basisparameterflags 'baseparams' auf '1' wird erreicht, dass zusätzlich zu den Verschlagwortungsfeldern auch alle Basisparameter angefragt werden. Speziell in der HOL wird dafür eine eigene Elementengruppe erzeugt:



Für alle Objekttypen werden die Werte für folgende Parameter ermittelt:

- § Ersteller
- § Erstellungsdatum
- § Änderungsperson
- § Änderungsdatum
- § Besitzer

Als Wert für den Besitzer wird dessen Name als text ausgegeben, die Benutzer GUID als Attribut 'osguid'.

Für Dokumente werden zusätzlich noch der ein- bzw. Auscheckstatus und der Archivierungsstatus bestimmt, sowie Retentionzeiten.

```
<BaseParams>
<Creator>liebe</Creator>
<Created>12.12.2003</Created>
<Modifier>root</Modifier>
<Modified>14.12.2003</Modified>
<Owner guid="BC1123CDFAAAS">liebe</Owner>
<ArchiveState>ARCHIVABLE</ArchiveState>
<Locked>SELF</Locked>
</BaseParams>
```

## Objektstatus

Durch Setzen des Attributs 'status' auf '1' im <DMSQuery> Element werden zu jedem Objekt folgende Statusinformationen ermittelt:

- § Verknüpfungen

Und speziell für Dokumente:

- § Modultyp
- § Archivierungsstatus
- § Eingecheckt/Ausgecheckt
- § Anzahl der Seiten
- § Anzahl der Realen Dokumenten Seiten (Systemfeld: OBJECT\_DOCPAGECOUNT). Wenn diese bekannt sind.

## Rechte

Durch Setzen des Attributs 'rights' auf '1' im <DMSQuery> Element, werden für den anfragenden Benutzer zu jedem Objekt die Zugriffsrechte ermittelt. Folgende Rechte werden ermittelt:

Attribut	Beschreibung
modify_index	Benutzer darf die Indexdaten des Objekts ändern
delete_object	Benutzer darf das Objekt löschen
export_object	Benutzer darf das Objekt öffnen bzw. Exportieren
modify_object	Benutzer darf Änderungen an den Dokumentendateien vornehmen.

**Beispiel:**

```
<Rights modify_index="1" delete_object="1" export_object="1"
modify_object="1" />
```

**Objekttyprelationen**

Über Objekttyprelationen lässt sich die Anzahl von Objektinstanzen eines gegebenen Typs einschränken. Über den Jobparameter 'ObjectInserts' bzw. über das Anfrageattribut 'object\_inserts' lässt sich die Anzahl der noch einfügbaren Objektinstanzen unter Berücksichtigung der Objekttyprelationen und des Rechtesystems anfragen. Für jeden möglichen Kindobjekttyp, der nach dem Rechtesystem eingefügt werden darf, gibt es dann unterhalb des <Rights> Elementes ein Element <ObjectInserts>, welches als Attribute den Objekttyp (Attribut 'type') und die Anzahl der einfügbaren Instanzen (Attribut 'count') enthält. Ist der Wert für 'count' = -1, gibt es keine Einschränkungen.

Weiterhin befindet sich im <Rights> Element das Attribut 'object\_inserts', welches anzeigt, ob Objekttyprelationen angefragt wurden.

Wurde das 'rights' Attribut nicht auf '1' gesetzt, werden die Attribute für die Zugriffsrechte im <Rights> Element auf -1 gesetzt.

```
<Rights object_inserts="1" modify_index="1" delete_object="1" export_object="1"
modify_object="1">
  <ObjectInserts type="65536" count="-1" />
  <ObjectInserts type="131108" count="-1" />
  <ObjectInserts type="131119" count="0" />
  <ObjectInserts type="196608" count="-1" />
  <ObjectInserts type="196619" count="0" />
  <ObjectInserts type="196622" count="-1" />
  <ObjectInserts type="262144" count="-1" />
  <ObjectInserts type="6488064" count="198" />
</Rights>
```

**Dateiinformatioenen**

Durch Setzen des Attributs 'fileinfo' auf '1' im <DMSQuery> Element, werden zu jedem Dokument die folgenden Dateiinformatioenen extrahiert und als Attribute in ein <FileProperties> Element geschrieben.

Wenn der Job-Parameter [FollowDocLink] auf '1' gesetzt ist (oder im <DMSQuery> Element), werden bei Dokumentverweisen („grüner Pfeil Verweis“) die Dateiinformatioenen des verlinkten Dokuments zurückgeliefert. In diesem Fall werden zusätzlich die Attribute 'linkid' und 'linktypeid' geschrieben.

Folgende Dateieigenschaften können ermittelt werden:

Attribut	Beschreibung
count	Anzahl der Dateien
size	Größe des Dokumentes in Bytes
extension	Dateityp Standarderweiterung
mimetype	Mimetyt
linkid	ID des verlinkten Objektes
linktypeid	Type ID des verlinkten Objektes
documentpagecount	Anzahl der Dokument Seiten (wenn bekannt)

**Beispiel:**

```
<Object id="415">
<FileProperties count="1" size="179489" extension="jpg"
mimetype="image/jpeg" />
</Object>
```

**Notizen**

Wenn das DMSQuery Attribut 'remarks' auf '1' gesetzt wird, werden Notizen und Notizverknüpfungen mit zurückgeliefert für jedes angefragte Dokument, wenn welche vorhanden sind. Es werden Textnotizen und Objektnotizen (Verknüpfungen) ausgegeben. Diese Funktion steht nur bei HOL-Anfragen zur Verfügung.

Attribut/Tag	Beschreibung
id	ID der Notiz
type	Typ der Notiz (1=Weiß,2=Gelb,3=Grün,4=Blau)
relation	Bei Objektrelation 1
medium	Medium ID wenn Notizen im Work Verzeichnis abgelegt werden, 0 wenn der Notiztext in der Datenbank gespeichert ist.
Creator	Anlegernamen mit interner ID des Anlegers als Attribut.
Created	Anlegedatum mit Timestamp als Attribut.
Modifier	Letzter Bearbeiter mit der interner ID des Bearbeiters Attribut.
Modified	Datum der letzten Bearbeitung mit Timestamp als Attribut.
Text	Notiztext mit interner ID, wenn die Notizen in der Datenbank abgelegt sind. Leer wenn es sich um eine Objektnotiz handelt.

**Beispiel:**

```
<Object id="81">
<Remarks>
<Remark id="1413" type="1" relation="0" medium="4">
<Creator id="16">MAIER</Creator>
<Created value="1143560855">28.03.2006 17:47:35</Created>
<Modifier id="18">SCHMIDT</Modifier>
<Modified value="1946463756">30.04.2006 14:50:12</Modified>
<Text id="">Notiztext</Text>
</Remark>
</Remarks>
</Object>
```

**Spracheinstellungen**

Über das Attribut 'lang\_id' oder 'query\_language' kann festgelegt werden, in welcher Sprache die Anfrage erfolgen soll. Die Feldnamen bzw. Objektnamen werden dann sprachabhängig gesucht. Ohne Angabe wird die Defaultsprache verwendet.

**Icons**

Durch setzen des Attributs 'icons' auf '1' in der DMSQuery wird das System angewiesen, die Icon-IDs aller benutzerdefinierten Icons zurückzuliefern. Dabei werden sowohl die die Icon-IDs aus dem Archivbereich, als auch die benutzerdefinierten Icon-IDs aus einer Trefferliste bestimmt. Die

DMSContent Elemente '<Object>' (HOL-Anfrage) und '<Row>' (LOL-Anfrage) erhalten zusätzlich das Attribut 'iconid'.

Um auf Basis der Icon-ID eine Bilddatei zu erhalten, kann der Job 'cnv.GetIcons' verwendet werden.

### Dokumentenvarianten

Für Dokumente vom Modultyp 'W-Dokumente' lassen sich in enaio® Varianten erstellen und verwalten. Durch Setzen des Attributs 'variants' auf '1' im <DMSQuery> Element, werden zu jedem W-Dokument die Varianten ermittelt und entsprechend ihrer hierarchischen Struktur über <DocumentVariant> und <DocumentVariants> Elemente ausgegeben. Dabei werden jedoch keine Indexdaten ermittelt.

Attribut	Bedeutung
is_active	Wenn diese Variante die aktive Variante ist, dann 1, sonst 0.
doc_id	Dokumenten ID dieser Variante
doc_ver	Versionsbezeichnung
doc_parent	ID der Ursprungsvariante, aus welcher diese Variante erzeugt worden ist. Zu beachten ist, dass hier auch IDs bereits gelöschter, nicht mehr im System vorhandener Dokument-IDs enthalten sein können.

### Beispiel:

```
<DocumentVariant is_active="1" doc_id="64758" doc_ver="Original"
doc_parent="0">
<DocumentVariants level="0">
<DocumentVariant is_active="0" doc_id="73405" doc_ver="1.0.0"
doc_parent="64758">
<DocumentVariants level="1">
<DocumentVariant is_active="0" doc_id="73406" doc_ver="1.1.0"
doc_parent="73405"/>
<DocumentVariant is_active="0" doc_id="73407" doc_ver="2.0.0"
doc_parent="64758">
<DocumentVariants level="1" />
</DocumentVariant>
</DocumentVariants>
</DocumentVariant>
```

### Hinweis:

Bei Recherchen nach W-Dokumenten wird immer die aktive Variante zurückgegeben.

### Mehrere Anfragen in einem Anfragedokument definieren

In einem Anfragedokument lassen sich mehrere Anfragen definieren.

### Beispiel:

```
<DMSQuery>
<Archive name="Adressen">
<ObjectType name="Adressen">
...
</ObjectType>
</Archive>
<Archive name="Patient">
```

```

<ObjectType name="Farbbilder" alias="F2">
  ...
</ObjectType>
<ObjectType name="Graustufenbilder" alias="G1">
  ...
</ObjectType>
<ObjectType name="Farbbilder" alias="F1">
  ...
</ObjectType>
</Archive>
</DMSQuery>

```

Um insbesondere auf Ergebnisse von Anfragen des gleichen Objekttyps gezielter zugreifen zu können, ist es möglich, das optionale Attribut 'alias' im <ObjectType> Element zu verwenden. Dieser Aliasname wird dann in den <ObjectType> Elementen des Ergebnisdokuments wiedergegeben.

Bei hierarchischen Anfragen mit Standortermittlung, wird das Attribut zusätzlich in das <ObjectType> Element des Ordners geschrieben.

### Blättern durch Trefferlisten

Um Trefferlisten seitenweise abzurufen, muss man eine Anfrage mehrfach stellen und dabei dem Job über den Parameter 'PageSize' die Seitengröße und über den Parameter 'Offset' den Startpunkt für die Trefferliste mitteilen. Mit 'MaxHits' kann man die Gesamttreffermenge begrenzen.

Im Ergebnisdokument findet sich am Ende einer Trefferliste das <Statistics> Element mit den Attributen startpos, pagesize und total\_hits:

```

<Statistics startpos="20" pagesize="20" total_hits="50" />

```

Daraus lässt sich ermitteln, wie viele Treffer die angeforderte Seite tatsächlich enthält und ob es noch weitere Treffer gibt. Der Wert des Attributs 'total\_hits' kann maximal den Wert des Eingabeparameters 'MaxHits' annehmen.

### Beispiel:

Aus einem Suchergebnis soll immer nur ein Ausschnitt von 20 Treffern angezeigt werden. Es sind maximal die ersten 100 Treffer von Interesse. Tatsächlich entsprechen aber 120 Objekte der Anfrage. Bei jedem Aufruf werden PageSize=20 und MaxHits=101 gesetzt. Würde man MaxHits nur auf 100 setzen, könnte man nicht erfahren, ob es über die 100 Treffer hinaus noch mehr Treffer gibt.

Seite	Eingabe	Ausgabe
1	Offset=0	<Statistics startpos='0' pagesize='20' total_hits='101' />
2	Offset=20	<Statistics startpos='20' pagesize='20' total_hits='101' />
3	Offset=40	<Statistics startpos='40' pagesize='20' total_hits='101' />
4	Offset=60	<Statistics startpos='60' pagesize='20' total_hits='101' />
5	Offset=80	<Statistics startpos='80' pagesize='20' total_hits='101' />

Nach dem Abruf der ersten Seite weiß der Aufrufende, dass es  $(101-1) / 20 = 5$  Seiten gibt. Mit der 5. Seite ist  $\text{startpos} + \text{pagesize} = 100$  und damit die gewünschte Trefferobergrenze erreicht. Da es aber  $101 > 100$  Treffer gibt, weiß der Aufrufende, dass es noch mehr Treffer gegeben hätte.

## DMS.GetObjectDetails

### Beschreibung:

Mit diesem Job lassen sich die Indexdaten eines einzelnen DMS Objektes ermitteln. Dabei ist der Standort unerheblich. Auch die Daten inaktiver Varianten lassen sich damit ermitteln. Das Ergebnis wird im [DMS Content](#) HOL Format zurückgegeben. Alle Eingabeparameter die bei dem Job [DMS.GetResultList](#) angegeben werden können, um die Ausgabeinformationen zu beeinflussen, sind auch bei GetObjectDetails analog verwendbar.

Bei diesem Job wird die Anfrageeigenschaft [FollowDocLink] im Default auf '1' gesetzt, wenn es nicht durch den Job-Parameter explizit ausgeschaltet wird. Das hat zur Folge, das im Default-Fall die Dokumenteneigenschaften des verlinkten DMS Objektes zurückgegeben werden, wenn [FileInfo] mit angefragt wird, und es sich um ein Dokumentverweis-Objekt handelt (s. [Dateiinformationen](#)).

### Parameter:

§ ObjectID (INT): Die ID der Objektinstanz

ObjectType (INT): Typ des Objekts. Wird dieser Parameter nicht oder als Wert -1 angegeben, bestimmt der Job selbst den Objekttyp.

[SystemFields] (String): Mit Semikolon getrennte Liste von zusätzlich angefragten Systemfeldern. Erwartet werden die internen Namen von Systemfeldern (siehe [Systemfelder](#)). Wenn Basisparameter angefragt werden, so haben diese Präferenz, d.h. die mit SystemFields angefragten Informationen stehen im Rückgabergebnis in dem jeweiligen Ergebnisblock (z. B. innerhalb von <baseparams>) und werden nicht redundant zurückgegeben.

Beispiel: SystemFields=OBJECT\_MEDDOCID;OBJECT\_MEDDOCNA

Weitere mögliche **Parameter**: Siehe Job [DMS.GetResultList](#)

### Rückgabewerte:

XML (BASE64): Indexdaten im XML-Format

Mit DMS.GetObjectDetails können neben den Indexdaten nur die Basisparameter der Objekte selbst aber keine weiteren Systemfelder, beispielsweise Standortdaten, ermittelt werden.

## DMS.GetDeletedObjects

### Beschreibung:

Mit diesem Job lässt sich der Inhalt des Papierkorbs des angemeldeten Benutzers ausgeben. Das Ergebnis wird im [DMS Content](#) Format zurückgegeben. Es gelten die gleichen Ausgabeformatoptionen wie für den Job [DMS.GetResultList](#).

Es kann auch eine Recherche innerhalb des Papierkorbs durchgeführt werden. Dazu muss eine Anfrage im DMSQuery XML Format formuliert werden und als 'XML' Parameter übergeben werden.

### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

§ 0x00001000 = XML Ergebnis wird UTF-8 kodiert, ansonsten UTF-16

[UserID] (INT): ID des Benutzers, dessen Papierkorb ausgelesen werden soll. Mit dem Wert -1 wird der gesamte Systempapierkorb ausgelesen, der Defaultwert 0 bezeichnet den angemeldeten Benutzer. Zum Auslesen des Systempapierkorbs bzw. des Papierkorbes eines

anderen Benutzers wird die Systemrolle 'Client: Systempapierkorb anzeigen' benötigt, andernfalls wird ein 'Zugriff verweigert' Fehlercode (-1069) zurückgeliefert.

#### § [XML] (BASE64): Anfrage im DMSQuery Format

[CheckParams] (INT): 0 = Wurde für einen Anfrageparameter kein Wert, werden Bedingungen, welche diesen Parameter referenzieren, ignoriert. Dies ist der Defaultwert. 1 = Es wird eine Fehlermeldung erzeugt, wenn ein referenzierter Parameter nicht definiert wurde.

#### Rückgabewerte:

Count (INT): Anzahl der Datensätze die zurückgegeben werden

TotalHits (INT): Anzahl der vorhandenen Treffer

In Abhängigkeit der Eingabeflags;

[XML] (BASE64): Trefferliste im XML-Format

Oder

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

## DMS.GetLinkedObjects

#### Beschreibung:

Mit diesem Job lassen sich die mit einem gegebenen Objekte verknüpften Objekte ausgeben. Das Ergebnis wird im [DMS Content](#) Format zurückgegeben. Es gelten die gleichen Ausgabeformatoptionen wie für den Job [DMS.GetResultList](#).

Die Ausgabe umfasst keine Notizen. Zusätzlich zu den Indexfeldern (vgl. auch Ausgabeformatoption 'FieldSchema') wird die Relations GUID zurückgegeben:

Bsp.: `<Column object="S/W-Scans" type="DOCUMENT" name="osrelid" system="1" datatype="TEXT" dbname="osrelid" ostype="X" osguid="1300" size="32">HYP_ID</Column>`

#### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformatates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

ObjectID (INT): ID des Objektes, dessen Verknüpfungen angezeigt werden sollen

#### Rückgabewerte:

TotalHits (INT): Anzahl der verknüpften Objekte

In Abhängigkeit der Eingabeflags;

[Result] (BASE64): Objektliste im XML-Format (oder einfachen Textformat, wenn gewünscht)

Oder

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

## DMS.GetForeignObjects

### Beschreibung:

Mit diesem Job lassen sich alle Objekte ermitteln die zu einem gegebenen Objekt eine Dokument Verweis haben (aka. grüner Pfeil Verweis). Das Ergebnis wird im [DMS Content](#) Format zurückgegeben. Es gelten die gleichen Ausgabeformatoptionen wie für den Job [DMS.GetResultList](#).

### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

ObjectID (INT): ID des Ziel Objektes, dessen Dokumenten Verknüpfungen angezeigt werden sollen. Die Ziel Objekt ID des grünen Pfeil Verweises.

[ObjectTypes]: (STRING) Eine mit Semikolon getrennte Auflistung der Verweis Typen. Wird dieser Parameter nicht gesetzt, so wird in allen Typen aus allen Schränken gesucht. Der Verweis Typ kann als Typ ID oder als interner Name des Typen übergeben werden. Wird als Typ ein Archiv (Schränk) übergeben, so werden alle enthaltenen Dokumenttypen als Quelltyp verwendet.

Beispiel: ObjectTypes=1;interne\_email;262432

Hier werden alle Dokumententypen aus dem Schränk mit der ID 1, ein Dokumenttyp mit dem internen Namen „interne\_email“ und Dokumente mit der Objekt Type ID 262432 als Quelltypen berücksichtigt.

Der Job unterstützt ansonsten die gleichen Eingabeparameter wie DMS.GetObjectDetails oder DMS.GetResultList.

### Rückgabewerte:

TotalHits (INT): Anzahl der sichtbaren Dokumente Verweis Objekte auf das Zielobjekt.

TotalForeignObjects (INT): Komplette Anzahl der Dokumenten Verweise auf das Zielobjekt. Bei Ermittlung der Anzahl wird das Sicherheitssystem nicht berücksichtigt. Wurde der Eingabeparameter „ObjectTypes“ verwendet, so wird auch nur die Anzahl der Verweise geliefert, die in den gegebenen Typen gefunden wurden.

In Abhängigkeit der Eingabeflags;

[Result] (BASE64): Objektliste im XML-Format (oder einfachen Textformat, wenn gewünscht)

Oder

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

## DMS.SelectDistinctFieldValues

### Beschreibung:

Dieser Job liefert eine Liste aller Werte eines gegebenen DMS Feldes. Tabellensteuerelemente und Mehrfachparameterfelder können nicht berücksichtigt werden.

### Parameter:

§ Flags (INT): immer 0

§ ObjectType (INT): Typ des Objekts



FieldName: (STRING):Name des Feldes

[Notation] (INT): Art des Feldnamens: 0 (Default) = DMS Name, 1 = interner Name , 2 = Datenbank Feldname

[Filter:] (STRING):Einschränkung/Suchkriterium. Für Textfelder wird automatisch ein \* angehängt.

[SortOrder] (STRING):Sortierreihenfolge. Erlaubte Werte: 'ASC' (default) oder 'DESC'

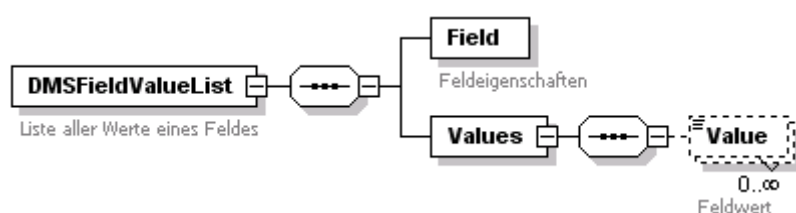
[MaxHits] (INT):Maximale Anzahl der zurückzuliefernden Werte. Default=-1 (alle)

### Rückgabewerte:

TotalHits (INT) Gesamtzahl der Feldwerte

Count (INT): Anzahl der ausgegebenen Feldwerte (vgl. Eingabeparameter Max Hits)

FieldValues (BASE64): Objektliste im UTF-8 kodierten DMSFieldValueList XML-Format:



## DMS.GetUserTrayObjects

### Beschreibung:

Dieser Job gibt sowohl typenbehaftete als auch typenlose Objekte aus der Ablage des angemeldeten Benutzers zurück. Das Ergebnis wird im [DMS Content](#) Format zurückgegeben. Es gelten die gleichen Ausgabeformatoptionen wie für den Job [DMS.GetResultList](#).

### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

### Rückgabewerte:

TotalHits (INT): Anzahl der Objekte in der Benutzerablage

In Abhängigkeit der Eingabeflags;

[Result] (BASE64): Objektliste im XML-Format

Oder

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

### Beispiel für eine Ausgabe (HOL Format):

```

<DMSContent format="HOL" version="4.60.617.4328" timestamp="2004-12-03T14:25:01"
user="LIEBE" station="MLIEBE">
<Archive name="Pressearchiv" id="1" osguid="8EE74447EFC7430F8793F2939A9C044F">
<ObjectType name="Word-Texte" id="262144" maintype="4" cotype="0"
osguid="838360EF620443EF9A66A280CF52F4AE" type="DOCUMENT" modul="WINDOWS"
table="object2">
<ObjectList>
<Object id="73145">
  
```

```

<Fields>
<Field name="links" system="1" datatype="INTEGER" dbname="links" ostyle="9"
osguid="1114" size="10">0</Field>
<Field name="anzahl" system="1" datatype="INTEGER" dbname="anzahl" ostyle="9"
osguid="1101" size="10">1</Field>
<Field value="2" name="flags" system="1" datatype="INTEGER" dbname="flags"
ostyle="9" osguid="1102" size="10">NOT_ARCHIVABLE</Field>
<Field value="0" name="lockuser" system="1" datatype="INTEGER" dbname="lockuser"
ostyle="9" osguid="1116" size="10">UNLOCKED</Field>
<Field value="4" name="haupttyp" system="1" datatype="INTEGER" dbname="haupttyp"
ostyle="9" osguid="1108" size="10">WINDOWS</Field>
<Field name="Autor" datatype="TEXT" dbname="feld1" ostyle="X"
osguid="F60B3D9B9E9C4FE8AE6CB78E4DE25826" size="50">Liebe</Field>
<Field name="Quelle" datatype="TEXT" dbname="feld2" ostyle="X"
osguid="E3EDAF50F4F4404C860E6043D7894272" size="150">Quelle1</Field>
<Field value="4711" name="Text2" datatype="TEXT" dbname="feld3" ostyle="X"
osguid="18C0D7D305DE40BB87A160B1115FC2A3" size="50">Dokumententext</Field>
</Fields>
</Object>
</ObjectList>
<Statistics startpos="0" pagesize="-1" total_hits="1"/>
</ObjectType>
</Archive>
<TypelessObjects>
<TypelessObject id="61359" maintype="2" module="BLACKWHITE" user="LIEBE"
timestamp="18.10.2002 11:19:10" pagecount="1"/>
<TypelessObject id="61360" maintype="2" module="BLACKWHITE" user="LIEBE"
timestamp="18.10.2002 11:19:12" pagecount="10"/>
<TypelessObject id="63399" maintype="3" module="COLOR" user="LIEBE"
timestamp="21.01.2003 15:23:36" pagecount="1"/>
</TypelessObjects>
<Messages/>
</DMSContent>

```

## DMS.GetWorkflowObjects

### Beschreibung:

Dieser Job gibt sowohl typenbehaftete als auch typenlose Objekte aus der Ablage des angemeldeten Benutzers zurück. Das Ergebnis wird im [DMS Content](#) Format zurückgegeben. Es gelten die gleichen Ausgabeformatoptionen wie für den Job [DMS.GetResultList](#).

### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

ID<1..n> (INT): ObjektIDs. Für jede Objektinstanz muss ein eigener Parameter angegeben werden. Also z. B.

ID1=123

ID2=245

...

### Rückgabewerte:

TotalHits (INT): Anzahl der tatsächlich gefundenen Objekte in der Workflowablage

In Abhängigkeit der Eingabeflags;

[Result] (BASE64): Objektliste im XML-Format

Oder

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

## DMS.ExecuteStoredQuery

### Beschreibung:

Mit diesem Job lassen sich gespeicherte Anfragen ausführen. Das Ergebnis wird im [DMS Content](#) Format zurückgegeben.

### Parameter:

Flags (INT): Flags zur Steuerung des Ausgabeformates

§ 0x00000010 = XML Ergebnis wird als Datei zurückgegeben, sonst als Buffer

§ 0x00001000 = XML Ergebnis wird UTF-8 kodiert, ansonsten UTF-16

QueryID (INT): ID der Anfrage

[CheckParams] (INT): 0 = Wurde für einen Anfrageparameter kein Wert definiert, werden Bedingungen, welche diesen Parameter referenzieren, ignoriert. Dies ist der Defaultwert. 1 = Es wird eine Fehlermeldung erzeugt, wenn ein referenzierter Parameter nicht definiert wurde.

Handelt es sich um eine parametrisierte Anfrage, müssen die Anfrageparameter als Jobparameter übergeben werden. Die \$-Zeichen, welche die Parameter in der Anfrage umschließen, müssen dabei weggelassen werden, also z. B. VAR1 oder STAT3.

Zusätzlich können folgende Parameter zur Formatierung des zurück gelieferten XML Dokumentes gesetzt werden: RequestType, OutputFormat, BaseParams, Offset, Pagesize, MaxHits, Rights, DateFormat, Variants, FileInfo, Baseparams. Die Beschreibung dieser Parameter findet sich in der Beschreibung des Jobs [dms.GetResultList](#)

### Rückgabewerte:

Count (INT): Anzahl der Datensätze die zurückgegeben werden

TotalHits (INT): Anzahl der vorhandenen Treffer

In Abhängigkeit der Eingabeflags;

[XML] (BASE64): Trefferliste im XML-Format

Oder

[FileCount] (INT): es wird immer nur eine Datei zurückgeliefert

[Dateiliste]: Name und Pfad der XML-Datei, die die Trefferliste enthält

## DMS.GetStoredQuery

### Beschreibung:

Mit diesem Job werden gespeicherte Anfragen im [DMS Query Format](#) zurückgegeben. Dabei werden alle Objektfelder in die Liste der angefragten Felder übernommen. Es gelten die gleichen Ausgabeoptionen wie bei [dms.GetResultList](#).

**Parameter:**

Flags (INT): Flags muss 0 sein

QueryID (INT): ID der Anfrage

[QueryMode] (INT): Legt das gewünschte Anfrageverhalten fest.

Mögliche Werte sind: Auto (-1) = Automatisch das verwenden, das in der Anfrage festgelegt wurde.

Folder(0) = Ordner werden angefragt. Register(1) = Register werden angefragt.

Dokument(2) = Dokumente werden angefragt. Ungefiltert (-2) Ordner+Register+Dokumente werden zurückgegeben.

Default ist "Ungefiltert (-2)".

**Rückgabewerte:**

[Query] (BASE64): Anfrage im XML-Format (UTF-8 kodiert)

[ExpertMode] (INT): 1, wenn die gespeicherte Anfrage eine Expertenanfrage ist.

**Beispiel:**

Gespeicherte Anfrage:

```
[196608@0]
#OSACT#=1
#OSPOS001#=$STAT1$
[SYSTEM]
NAME=Stat1
IDENT=73706
VARREQUEST=1
DEFACTION=0
```

**Konvertierte Anfrage:**

```
<DMSQuery requesttype="LOL" outputformat="LOL">
  <Params>
    <Param name="$STAT1$"></Param>
  </Params>
  <Archive name="Pressearchiv">
    <ObjectType name="Farbbilder" alias="Stat1">
      <Fields>
        <Field name="Datum" system="0"></Field>
        <Field name="Autor" system="0"></Field>
        <Field name="Quelle" system="0"></Field>
        <Field name="Inhalt" system="0"></Field>
        <Field name="ansichtsfähig" system="0"></Field>
      </Fields>
      <Conditions>
        <ConditionObject name="Farbbilder">
          <FieldCondition name="Autor" operator="=">
            <ParamValue ref="$STAT1$"></ParamValue>
          </FieldCondition>
        </ConditionObject>
      </Conditions>
    </ObjectType>
  </Archive>
</DMSQuery>
```

**DMS.AddStoredQuery****Beschreibung:**

Mit diesem Job lässt sich eine neue gespeicherte Anfrage erstellen. Die Anfrage muss im [DMS Query Format](#) übergeben werden. Die Anfrage wird in das interne Format für gespeicherte Anfragen umgewandelt. Da das Format für gespeicherte Anfragen nur eine Teilmenge der DMS Query Anfragemöglichkeiten zulässt, gib es hierbei gewisse Einschränkungen. Siehe [dms.ConvertQuery](#).

**Parameter:**

Flags (INT): Flags muss 0 sein

Name (STRING): Name der Anfrage

Query (STRING/Base64) Anfrage im DMSQuery XML Format

[TreeParent] (INT): ID des Desktop Ordners, in dem die Anfrage liegt

[Scope] (STRING): Gültigkeitsbereich. Erlaube Werte: 'Public' für öffentliche Anfragen, 'Private' für benutzerbezogene Anfragen. Default ist 'private'.

[IconID] (INT): ID eines Icons, dass im Client angezeigt wird. Default=0

[DefAction]: Aktion die beim Öffnen der gespeicherten Anfrage durchgeführt werden soll.

0=Ausführen, 1=bearbeiten, 2=Anzahl ermitteln. Default=0

**Rückgabewerte:**

QueryID (INT): ID der Anfrage

## DMS.UpdateStoredQuery

**Beschreibung:**

Mit diesem Job lassen sich eine bestehende gespeicherte Anfrage und/oder deren Eigenschaften aktualisieren. Die Anfrage muss im [DMS Query Format](#) übergeben werden. Die Anfrage wird in das interne Format für gespeicherte Anfragen umgewandelt. Da das Format für gespeicherte Anfragen nur eine Teilmenge der DMS Query Anfragemöglichkeiten zulässt, gib es hierbei gewisse Einschränkungen. Siehe [dms.ConvertQuery](#).

**Parameter:**

Flags (INT): Flags muss 0 sein

QueryID (INT): ID der Anfrage

[Query] (STRING/Base64) Anfrage im DMSQuery XML Format

[Name] (STRING): Neuer Name der Anfrage, wenn die Anfrage umbenannt werden soll

[IconID] (INT): ID eines Icons, dass im Client angezeigt wird. Default=0

[DefAction]: Aktion die beim Öffnen der gespeicherten Anfrage durchgeführt werden soll.

0=Ausführen, 1=bearbeiten, 2=Anzahl ermitteln. Default=0

**Rückgabewerte:** keine jobspezifischen

## DMS.RemoveStoredQuery

**Beschreibung:**

Mit diesem Job lässt sich eine bestehende gespeicherte Anfrage löschen.

**Parameter:**

Flags (INT): Flags muss 0 sein

QueryID (INT): ID der Anfrage

**Rückgabewerte:** keine jobspezifischen

## DMS.ConvertQuery

### Beschreibung:

Mit diesem Job lassen sich verschiedene Anfrageformate ineinander konvertieren.

Folgende Formate werden zurzeit unterstützt:

**DMS** – DMSQuery XML Format

**STQ** – Format für gespeicherte Anfragen

**ABN** – Format für Abonnements

### Parameter:

Flags (INT): Flags muss 0 sein

Query (STRING oder BASE64): Anfragetext

InputFormat (STRING): Eingabeformat (DMS, STQ, ABN)

OutputFormat (STRING): Ausgabeformat (DMS, STQ, ABN)

Wird als Ausgabeformat STQ (gespeicherte Anfrage) gewählt, müssen noch folgende Jobparameter spezifiziert werden.

Name (STRING): Name der Anfrage

QueryID (INT): ID der Anfrage

[IconID] (INT): ID eines Icons, dass im Client angezeigt wird. Default=0

[DefAction]: Aktion die beim Öffnen der gespeicherten Anfrage durchgeführt werden soll.

0=Ausführen, 1=bearbeiten, 2=Anzahl ermitteln. Default=0

Wird als Ausgabeformat ABN (Abonement) gewählt, so kann noch folgender Jobparameter spezifiziert werden:

[GarbageMode] (INT): 1=Objekte aus dem Papierkorb berücksichtigen. 0=Objekte aus dem Papierkorb nicht berücksichtigen.

### Rückgabewerte:

Query (BASE64): Anfragetext. Für das DMSQuery XML Format UTF-8 kodiert sonst ANSI.

### Einschränkungen:

Da die Anfragemöglichkeiten des DMSQuery XML Formates über die Möglichkeiten der anderen Formate hinausgehen, bestehen folgende Limitierungen für das DMSQuery XML Format als Eingabeformat:

\* Hierarchische Strukturen werden nicht unterstützt, d.h. die <ParentObjects>, <ChildObjects> und <ExternalObjects> werden ignoriert.

\* Parameternamen müssen das Format \$VARnnnn\$ bzw. \$STATnnnn\$ besitzen, wobei nnn eine Zahl zwischen 000 und 999 sein kann.

\* Es dürfen keine Feldgruppen innerhalb von Bedingungen verwendet werden.

\* Für jede Bedingung darf nur jeweils ein Wert angegeben werden.

- \* Es können keine Bedingungen für Basisparameter und Systemfelder formuliert werden.
- \* Gespeicherte Anfragen (STQ) im Expertenmodus können nicht konvertiert werden.

## DMS.GetObjectHistory

### Beschreibung:

Dieser Job liefert zu einem vorgegebenen Objekt die Bearbeitungshistorie im XML Format.

### Parameter:

Flags (INT): Flags muss 0 sein

ObjectID (INT): ID des Objektes

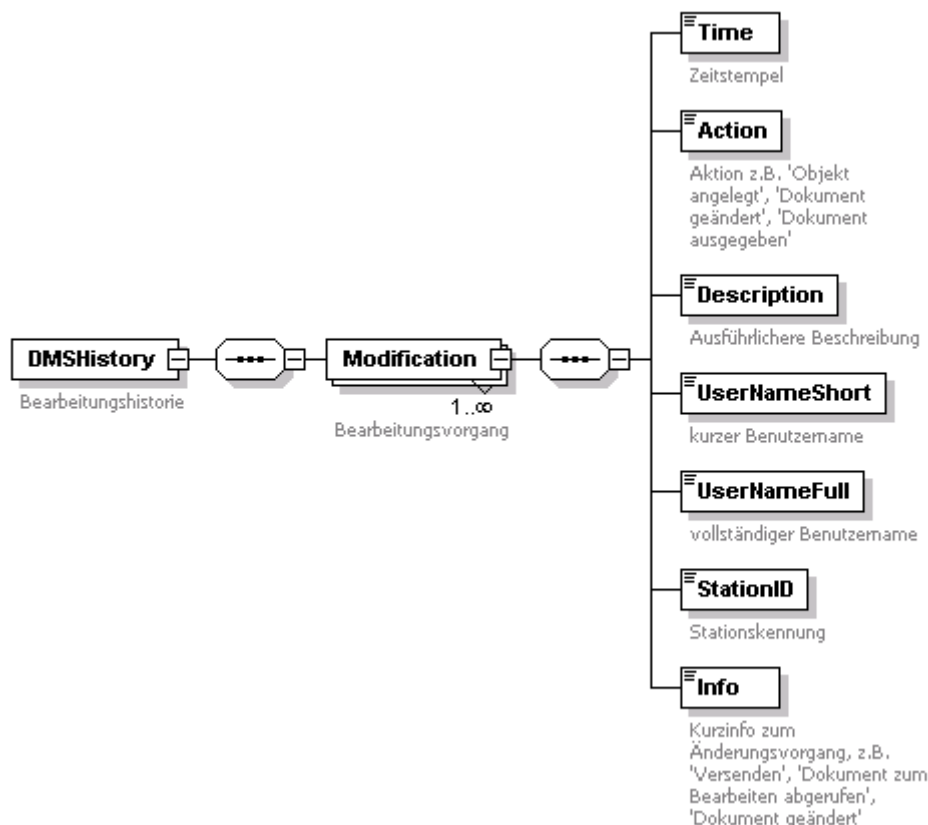
[LangID(INT)]: Sprache, in der die Aktionsbeschreibungen ausgegeben werden sollen.

Aktionsbeschreibungen liegen in den Sprachen Deutsch (7), Englisch (9) und Französisch (12) vor.

Wenn dieser Parameter nicht angegeben wird oder die Sprache nicht zur Verfügung steht, wird Deutsch als Defaultsprache verwendet.

[Encoding (STRING)]: XML Encoding für das Ergebnis. Erlaubt sind „UTF-16“ (Default) und „UTF-8“.

**Rückgabewerte:** History (Base64): Bearbeitungshistorie im XML-Format:



Anm.: Wenn der Benutzer zwischenzeitig gelöscht wurde, dann ist UserNameShort und UserNameFull nicht ausgefüllt.

## DMS.GetShadowData

### Beschreibung:

Mit diesem Job lassen sich die Indexdaten eines Objektes aus den Schattentabellen lesen.

**Parameter:**

Flags (INT): Flags muss 0 sein

Guid (String): GUID des Änderungsvorgangs aus der Historien-Tabelle à [DMS.GetObjectHistory](#)

[ObjectID (INT)]: ID des Objektes

[Encoding (STRING)]: XML Encoding für das Ergebnis. Erlaubt sind „UTF-16“ (Default) und „UTF-8“.

**Rückgabewerte:** ShadowData (Base64): Indexdaten im DMSContent –HOL -Format

## DMS.GetObjectsByDigest

**Beschreibung:**

Dieser Job liefert Objektinformationen auf Basis eines Hashwertes (Fingerprint) für eine beliebige Datei. Er kann dazu benutzt werden, zu bestimmen, ob eine Datei schon im System abgelegt wurde. Der Job benutzt seinerseits den Job 'std.FindDocumentDigest' führt allerdings anschließend noch eine Überprüfung auf Validität aus. Der Eingabeparameter 'Digest' ist ein über die Datei gebildeter Fingerabdruck der mit dem SHA2-256 Algorithmus bestimmt werden kann. In den jeweiligen Clientbibliotheken stehen Hilfsfunktionen zur Verfügung um aus einer Datei den Digest bestimmen zu können, ohne die Datei an den Server übertragen zu müssen. (siehe JDL:DigestUtil , CDL:CalcFileDigest )

**Parameter:**

Flags (INT): Flags muss 0 sein

Digest (String): Hashwert der Datei, dessen Vorhandensein überprüft werden soll. (SHA2-256Bit)

**Rückgabewerte:**

TotalHits (INT): Anzahl der Treffer

ObjectIds (String): Kommaseparierte Liste der Objekt IDs der gefundenen Dokumente.

TypeIds (String): Kommaseparierte Liste der Objekt Typen.

## DMS Referenz

§ [Systemfelder](#)

§ [Datumsformate](#)

**Systemfelder**

Folgende Systemfelder können in die XML Anfrage einbezogen werden:

Interner Name [@internal_name]	Feldnummer [@osguid]	Datenbankfeldname [@fieldname]	Typ	Länge
OBJECT_ID	1100	id	Text	10
OBJECT_COUNT	1101	anzahl	Text	10
OBJECT_FLAGS	1102	flags	Text	10
OBJECT_AVID	1103	archivar	Text	255
OBJECT_AVDATE	1104	archiviert	Datum	



OBJECT_CRID	1105	anleger	Text	255
OBJECT_CRDATE	1106	angelegt	Datum	
OBJECT_TIME	1107	zeitstempel	Zeitstempel	10
OBJECT_MAIN	1108	haupttyp	Text	10
OBJECT_CO	1109	untertyp	Text	10
OBJECT_MEDDOCID	1110	medium_doc	Text	10
OBJECT_MEDDIAID	1111	medium_dia	Text	10
OBJECT_MEDDOCNA	1112	name_doc	Text	24
OBJECT_MEDDIANA	1113	name_dia	Text	24
OBJECT_LINKS	1114	links	Text	10
OBJECT_VERID	1115	version	Text	10
OBJECT_LOCKUSER	1116	lockuser	Text	10
OBJECT_SYSTEMID	1117	systemid	Text	10
OBJECT_MODIFYTIME	1118	modifytime	Zeitstempel	10
OBJECT_MODIFYUSER	1119	modifyuser	Text	256
OBJECT_FOREIGNID	1124	foreignid	Text	10
OBJECT_USERGUID	1125	osowner	Text	32
OBJECT_DELETED	1126	deleted	Text	10
OBJECT_INDEXHISTFLAGS	1127	indexhistflags	Text	10
OBJECT_DOCHISTFLAGS	1128	dochistflags	Text	10
OBJECT_OSSD	1129	ossd	Text	32
OBJECT_MIMETYPEID	1900	mimetypeid	Text	10
OBJECT_FILESIZE	1902	filesize	Text	10
OBJECT_RETENTION_PLANNED	1903	retention_planned	Datum	10
OBJECT_RETENTION	1904	retention	Datum	10
STAMM_ID	1000	id	Text	10
STAMM_TIME	1001	zeitstempel	Zeitstempel	10
STAMM_LINKS	1002	links	Text	10
REG_ID	1120	id	Text	10
REG_STAID	1121	stamm_id	Text	10
REG_PARID	1122	parent_id	Text	10
SDSTA_ID	1130	stamm_id	Text	10

SDOBJ_ID	1131	object_id	Text	10
SDOBJTYPE	1132	objekttyp	Text	10
SDREG_ID	1133	register	Text	10
SDDEL	1134	loeschen	Text	10
SDTIME	1135	zeitstempel	Text	10
SDREG_TYPE	1136	regtype	Text	10
FOLDERID	1181	folderid	Text	10
FOLDERTYPE	1182	foldertype	Text	10
REGISTERID	1183	registerid	Text	10
REGISTERTYPE	1184	registertype	Text	10
PARENTREGID	1185	parentregid	Text	10
PARENTREGTYPE	1186	parentregtype	Text	10
MDDEL	1140	loeschen	Text	5
MDTIME	1141	zeitstempel	Zeitstempel	10
MDMAP_ID	1142	mappe_id	Text	10
MDSTA_ID	1143	stamm_id	Text	10
MDOBJ_ID	1144	object_id	Text	10
MDOBJTYPE	1145	objekttyp	Text	10
MDMOD	1146	modul	Text	5
MDIN	1147	eingang	Text	10
MDOUT	1148	ausgang	Text	10
MDCOUNT	1149	anzahl	Text	5

### Datumsformate

Formatierungsanweisungen werden durch ein Prozentzeichen (%) eingeleitet. Zeichenfolgen, die nicht mit einem % beginnen, werden unverändert in den Ergebnisstring kopiert. Folgende Formatierungsanweisungen können verwendet werden:

Formatierung	Beschreibung
%a	Abgekürzter Wochentagsname
%A	Wochentagsname
%b	Abgekürzter Monatsname
%B	Monatsname
%c	Datums- und Zeitrepräsentation entsprechend der lokalen Einstellungen
%d	Tag im Monat numerisch (01-31)
%j	Tag im Jahr numerisch (001-366)
%m	Month als Zahl (01-12)

%U	Kalenderwoche (mit Sonntag als erster Wochentag) (00-53)
%w	Wochentag als Zahl (0-6; Sonntag ist 0)
%W	Wochentag als Zahl (0-6; Montag ist 0)
%x	Datumsr�presentation enstsprechend der lokalen Einstellungen
%X	Zeitrepr�sentation enstsprechend der lokalen Einstellungen
%y	Zweistellige Jahresangabe (00-99)
%Y	Vierstellige Jahresangabe
%z, %Z	(Abgek�rzter) Name der Zeitzone; leer wenn Zeitzone unbekannt
%%	Prozentzeichen

## Sicherheitssystem

§ [DMS.CheckPermission](#)

§ [DMS.CheckPermissions](#)

§ [DMS.CopySD](#)

§ [DMS.CreateSD](#)

§ [DMS.DeleteSD](#)

§ [DMS.ReadSD](#)

§ [DMS.SetSD](#)

## Ausf hrliche Beschreibung

Mit der Version 4.50 steht, zus tzlich zum bestehenden Rechtesystem, ein Sicherheitssystem auf Objektebene (SSOL) zur Verf gung. Bei diesem System kann f r jedes Objekt ein sogenannter Security Descriptor (SD) erzeugt werden, der auf eine Access Control List (ACL), d.h. Liste von Zugriffssteuerungseintr gen (ACE-Access Control Entry) verweist. Mit einem Zugriffssteuerungseintrag lassen sich f r einen Benutzer oder eine Benutzergruppe Zugriffsberechtigungen f r das  ndern von Indexdaten und das  ndern, L schen und Exportieren von Objekten festlegen.

Im DMS Executor sind daf r Jobs zur Bearbeitung der Zugriffsstrukturen f r dieses Sicherheitssystem implementiert. Weiterhin existiert mit [DMS.CheckPermission](#) ein Job, der unabh ngig vom verwendeten Rechtesystem die Zugriffsrechte auf ein bestimmtes Objekt pr ft.

## Liste der Zugriffssteuerungseintr gen im XML-Format

Um die Liste der Zugriffssteuerungseintr ge zu beschreiben wird XML verwendet. Die Jobs DMS.CreateSD, DMS.ReadSD und DMS.SetSD verwenden das selbe XML-Schema, welches  ber den Job [DMS.GetXMLSchema](#) abgerufen werden kann.

### Beispiel:

Liste der Zugriffssteuerungseintr ge im XML-Format (DMSAccess)

```
<DMSAccess timestamp="" version="4.50">
<ACL ossd="" object_type="" object_id="">
<UserACE modify_index="0" modify_object="0" delete_object="0"
export_object="0" osuid=""/>
<GroupACE modify_index="0" modify_object="0" delete_object="0"
export_object="0" osgid=""/>
</ACL>
</DMSAccess>
```

**Erläuterung der <DMSAccess> Attribute:**

- § timestamp: Erstellzeitpunkt der [Access Control List](#) (Format: JJJ-MM-DDTHH:MM:SS)
- § version: Produktversionsnummer

**Erläuterung der <ACL> Attribute:**

- § ossd (STRING): GUID des [Security Descriptor](#)
- § obj\_type (long): Objekttyp
- § obj\_id (long): ID der Objektinstanz

**Erläuterung der <UserACE> bzw. <GroupACE> Attribute:**

- § osuid (STRING): GUID des Benutzers
- § osgid (STRING): GUID der Benutzergruppe
- § modify\_index (long): Zugriffsart Indexdaten schreiben
  - § 0 = nicht gesetzt
  - § 1 = erlaubt
  - § 2 = verboten
- § modify\_object (long): Zugriffsart Objekt bearbeiten
  - § 0 = nicht gesetzt
  - § 1 = erlaubt
  - § 2 = verboten
- § delete\_object (long): Zugriffsart Objekt löschen
  - § 0 = nicht gesetzt
  - § 1 = erlaubt
  - § 2 = verboten
- § export\_object (long): Zugriffsart Objekt exportieren
  - § 0 = nicht gesetzt
  - § 1 = erlaubt
  - § 2 = verboten

**Glossar**

- § **SSOL** - Security System at Object Level (Sicherheitssystem auf Objektebene)
- § **ACE** - Access Control Entry (Zugriffssteuerungseintrag): Enthält für einen Benutzer oder eine Benutzergruppe die erlaubten oder verbotenen Zugriffsarten
- § **ACL** - Access Control List (Zugriffssteuerungsliste): Liste aller ACEs, die einem Objekt zugeordnet sind.
- § **SD** - Security Descriptor: Für jedes Objekt kann ein Security Descriptor erzeugt werden, welcher eine ACL identifiziert. Für Objekte mit SD gilt das SSOL (s.o.), für Objekte ohne SD das bisherige Rechtesystem auf Objekttypebene.

**DMS.CheckPermission****Beschreibung:**

Mit diesem Job lassen sich die Zugriffsberechtigungen für einzelne DMS-Objekte unabhängig vom verwendeten Sicherheitssystem überprüfen. Um zu prüfen, ob ein Objekt an einem bestimmten Standort eingefügt werden darf, muss der Standort über den Parameter 'FolderID' oder den Parameter 'RegisterID' festgelegt werden, der Parameter 'ObjectID' auf 0 gesetzt und die Zugriffsart 'W' geprüft werden.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Access (STRING): zu prüfende Zugriffsart (z. B. 'RWXDU' prüft alle Zugriffsarten)

§ R = Indexdaten lesen

§ W = Indexdaten schreiben

§ X = Objekt öffnen/ausführen

§ D = Objekt löschen

§ U = Objekt schreiben

§ ObjectType (INT): Typ des Objekts

ObjectID (INT): ID des Objekts

[RegisterType] (INT): Typ des übergeordneten Registers

§ 0 = in keinem Register (direkt auf Ordner Ebene)

§ -1 = Registerunabhängig

[RegisterID] (INT): ID des übergeordneten Registers

[FolderID] (INT): ID des Ordners

**Rückgabewerte:**

Access (STRING): Gewährte Zugriffsarten (Format entspricht dem Eingabeparameter 'Access')

## DMS.CheckPermissions

**Beschreibung:**

Mit diesem Job lassen sich die Zugriffsberechtigungen für eine Liste von DMS-Objekten überprüfen. Da es bei diesem Job sehr auf Performance ankommt, werden keine Ablagenobjekte und keine inaktiven Varianten berücksichtigt.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Access (STRING): zu prüfende Zugriffsart (z. B. 'RWXDU' prüft alle Zugriffsarten)

§ R = Indexdaten lesen

§ W = Indexdaten schreiben

§ X = Objekt öffnen/ausführen

§ D = Objekt löschen

§ U = Objekt schreiben

ObjectType<n> (INT): Typ der Objekte. <n> ist dabei eine fortlaufende Nummer beginnend mit 1.

ObjectList<n> (String): Kommaseparierte Liste der Objekt IDs vom Typ ObjectType<n>

[RegisterType] (INT): Typ des übergeordneten Registers

§ 0 = in keinem Register (direkt auf Ordner Ebene)

§ -1 = Registerunabhängig

[RegisterID] (INT): ID des übergeordneten Registers

[FolderID] (INT): ID des Ordners

### **Rückgabewerte:**

ObjectType<n> (INT): n-ter Objekttyp

ObjectList<n> (String): Kommaseparierte Liste der IDs und der ermittelten Rechte zum Objekttyp n.  
Die ObjektID ist durch einen Doppelpunkt von den ermittelten Rechten getrennt. S. Beispiel unten.

### **Beispiel:**

#### Aufruf:

Flags=0

Access =WRXDU

FolderID=380

FolderType=1

RegisterID=0

RegisterType=0

ObjectType1=393216

ObjectList1=65493

ObjectType2=131072

ObjectList2=72272,72273,72274

#### Rückgabe:

ObjectType1=131072

ObjectList1=72272:RWXUD,72273:RW---,72274:R-X--

ObjectType2=393216

ObjectList2=65493:RWXUD

#### **Hinweis:**

Sowohl die Reihenfolge der Objekttypen als auch der Objekte in einer ID Liste muss nicht mit der Reihenfolge in den Eingabeparametern übereinstimmen.

## **DMS.CopySD**

### **Beschreibung:**

Dieser Job weist eine Kopie der Zugriffssteuerungseinträge eines Objektes einem oder mehreren anderen Objekten zu. Beim Kopieren werden nur für solche Benutzer bzw. Benutzergruppen die Zugriffsrechte auf die Zielobjekte übertragen, für die es noch keine Einträge in dem jeweiligen Zielobjekt gibt. Umgekehrt werden bestehende Zugriffsrechte von Benutzern, für die keine Einträge im Quellobjekt existieren, nicht berührt.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

ObjectID (INT): ID des Ausgangsobjektes

ObjectType (INT): Typ des Ausgangsobjektes

Destination (STRING): Liste der Zielobjekte im XML-Format

**Beispiel:**

Aufbau von Destination

```
<DMSAccess>
<ACL object_type="XXX1" object_id="YYY1" />
<ACL object_type="XXX2" object_id="YYY2" />
<ACL object_type="XXX3" object_id="YYY3" />
</DMSAccess>
```

**Hinweis:**

Genauere Beschreibung von Destination

§ object\_type: Typ des Objekts

§ object\_id: ID des Objekts

## DMS.CreateSD

**Beschreibung:**

Dieser Job erzeugt einen [Security Descriptor](#) für jede im Parameter 'XmlInfo' definierte Objektinstanz.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

XmlInfo (STRING): Liste von Objekten im XML Format, für die Security Descriptoren erzeugt werden soll

**Rückgabewerte:**

XmlInfo: (STRING): ACL Liste im XML-Format

**Beispiel:**

Aufbau des Eingabeparameters XmlInfo

```
<DMSAccess>
<ACL obj_type="1" obj_id="61967" />
<ACL obj_type="1" obj_id="61968" />
</DMSAccess>
```

**Beispiel:**

zurückgelieferte ACL-Liste im XML-Format (Beschreibung: siehe [ACL-XML-Schema](#))

```
<DMSAccess timestamp="2004-04-08T12:59:25" version="4.50">
<ACL ossd="6DBEC785D3CEB4D894B" obj_type="1" obj_id="61967" />
<ACL ossd="2AB8AB82E0CEB4D8BDD" obj_type="1" obj_id="61968" />
</DMSAccess>
```

**Siehe auch:**

[DMS.SetSD](#)

## DMS.DeleteSD

### Beschreibung:

Mit diesem Job lässt sich der [Zugriffssteuerungseintrag](#) eines Benutzers/Gruppe oder die gesamte [Zugriffssteuerungsliste](#) für ein angegebenes Objekt löschen.

### Parameter:

Flags (INT): Flags kann folgende Werte annehmen

- § 0 - Das Objekt wird über die Parameter 'ObjectType' und 'ObjectID' festgelegt. Der SD des Objekts und damit alle ACEs zu diesem Objekt werden gelöscht
- § 1 - Der Security Descriptor (und damit das Objekt) wird über den Parameter 'Ossd' festgelegt. Es werden alle ACEs gelöscht, die zu dem über den Parameter 'Osuid' gekennzeichneten Benutzer / der Benutzergruppe gehören.
- § 2 - Objekt wird über die Parameter 'ObjectType' und 'ObjectID' festgelegt. Es werden alle ACEs gelöscht, die zu dem über den Parameter 'Osuid' gekennzeichneten Benutzer / der Benutzergruppe gehören.
- § ObjectType (INT): Typ des Objekts

ObjectID (INT): ID der Objektinstanz

Ossd (STRING): GUID des Security Descriptors

Osuid (STRING): GUID des Benutzers/der Benutzergruppe, dessen Zugriffsrechte gelöscht werden sollen

## DMS.ReadSD

### Beschreibung:

Dieser Job liefert die ACEs für einen Benutzer bzw. eine Benutzergruppe oder aller Benutzer bzw. Benutzergruppen für ein gegebenes Objekt im XML Format.

### Parameter:

Flags (INT): 0 = [ACL](#) des Objects wird zurückgegeben; 1 = [ACE](#) für Benutzer/Benutzergruppe wird zurückgegeben. (s. Parameter 'Osuid')

ObjectId (INT): ID der Objektinstanz

ObjectType (INT): Typ des Objekts

[Osuid] (STRING): Benutzer oder Gruppen GUID

### Rückgabewerte:

XmlInfo (BASE64): ACL im XML Format

### Beispiel:

Aufbau von XmlInfo (Beschreibung: siehe [DMSAccess](#))

```
<DMSAccess>
<ACL ossd="XXX1">
<UserACE osuid="YYY1" modify_object="1" export_object="1"/>
<UserACE osuid="YYY2" modify_object="2" export_object="1"/>
<GroupACE osuid="YYY3" export_object="1"/>
</ACL>
</DMSAccess>
```



## DMS.SetSD

### Beschreibung:

Dieser Job erstellt einen oder mehrere [Zugriffssteuerungseinträge](#).

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

XmlInfo (BASE64): Xml-formatierter String, welcher die ACEs enthält

### Beispiel:

Aufbau von XmlInfo (Beschreibung: siehe [DMSAccess](#))

```
<DMSAccess>
<ACL ossd="XXX1">
<UserACE osuid="YYY1" modify_object="1" export_object="1"/>
</ACL>
<ACL ossd="XXX2">
<UserACE osuid="YYY1" modify_object="1" export_object="1"/>
<UserACE osuid="YYY2" modify_object="2" export_object="1"/>
</ACL>
<ACL ossd="XXX3">
<GroupACE osuid="YYY3" export_object="1"/>
</ACL>
</DMSAccess>
```

## Relationen und Relationstexte

§ [DMS.AddRel](#)

§ [DMS.AddRelText](#)

§ [DMS.AddRelTextLang](#)

§ [DMS.DelRel](#)

§ [DMS.DelRelText](#)

§ [DMS.ModRel](#)

§ [DMS.ModRelText](#)

§ [DMS.ModRelTextLang](#)

§ [DMS.RetrieveRelations](#)

§ [DMS.RetrieveRelTexts](#)

### Ausführliche Beschreibung

Objekte können ab Version 4.50 per Relationen verknüpft werden. Dabei wird der Beziehung zwischen zwei Objekten eine vordefinierte Bezeichnung (Relationstext) vergeben. Diese Beziehungen können pro Paar von Objekttypen definiert werden.

### Relationen im XML-Format

#### Beispiel:

```
<Relations>
<Relation obj_id1="1" obj_typ1="2" obj_id2="3" obj_typ2="4"
valid_from="3443234" valid_to="433444"
ensured="1" checked="8"
reltextid="AF32988234DEFA78979879"
IObjId="9" IObjType="10" >
```

```
</Relation>
</Relations>
```

**Bemerkungen:**

Das Wurzelement kann entfallen, wenn nur ein Eintrag definiert wurde.

**Beschreibung der Attribute des Elements 'RelationXML '**

- § obj\_id1 (LONG): ID des 1. Objekts
- § obj\_typ1 (LONG): Typ des 1. Objekts
- § obj\_id2 (LONG): ID des 2. Objekts
- § obj\_typ2 (LONG): Typ des 2. Objekts
- § valid\_from (LONG): Zeitstempel für Gültigkeitsbeginn
- § valid\_to (LONG): Zeitstempel für Gültigkeitsende
- § ensured (LONG): Flag: 1=gesichert, 0=nicht sicher
- § checked (LONG): Flag: 1=überprüft, 0=ungeprüft
- § created (LONG): Zeitstempel der Erzeugung
- § created\_to (LONG): Endzeitstempel der Erzeugung (nur bei Suchanfrage von Bedeutung)
- § modified (LONG): Zeitstempel der letzten Änderung
- § modified\_to (LONG): Endzeitstempel der letzten Änderung (nur bei Suchanfrage von Bedeutung)
- § deletetime (LONG): Zeitstempel des Löschezitpunktes
- § deletetime\_to (LONG): Endzeitstempel des Löschezitpunktes (nur bei Suchanfrage von Bedeutung)
- § deleteuser (STRING): Name des Benutzers, der die Relation aus dem System gelöscht hat.
- § relid (STRING): GUID der Relation
- § reltextid (STRING): Verweis auf den Relationstext (GIUD)
- § iobjid (LONG): ID des Objektes, das die Grundlage für die Relation bildet
- § iobjtype (LONG): Typ des Objektes, das die Grundlage für die Relation bildet

**Relationstexte im XML-Format****Beispiel:**

```
<RelTexts>
<RelText shortrel="Eigentum"
active="1"
objtype1="2"
objtype2="4"
to="ist Eigentümer von"
reverse="ist Eigentum von"
langid="9"/>
</RelTexts>
```

**Bemerkungen:**

Das Wurzelement <RelTexts> kann entfallen, wenn nur ein <RelText> Eintrag definiert wurde.

**Erläuterung der <RelText> Attribute:**

- § shortrel (STRING): Kurzbeschreibung des Relationstextes (max. 32 Zeichen)
- § active (INT): Aktivitätsstatus 0=nicht aktiv, 1=aktiv

§ objtype1 (INT): ID des 1. Objekts

§ objtype2 (INT): ID des 2. Objekts

§ to (STRING): Beschreibung der Relation von Objekttyp 1 zu Objekttyp 2

§ reverse (STRING): Beschreibung der Relation von Objekttyp 2 zu Objekttyp 1

§ langid (INT): Sprach-ID (s. [Sprachcode](#))

### Sprachcodes

Der DMS Executor unterstützt in mehreren Jobs Mehrsprachigkeit. Die Sprachen müssen zuvor im enaio® System konfiguriert werden. Als Sprachcodes werden die 'Windows Language Identifier' Codes verwendet, es werden allerdings nur die Hauptsprachen unterstützt. Die folgende Tabelle enthält die ersten 31 Sprachcodes (hexadezimal):

ID	Sprache	ID	Sprache
0x01	Arabic	0x11	Japanese
0x02	Bulgarian	0x12	Korean
0x03	Catalan	0x13	Dutch
0x04	Chinese	0x14	Norwegian
0x05	Czech	0x15	Polish
0x06	Danish	0x16	Portuguese
0x07	German	0x17	Arabic
0x08	Greek	0x18	Romanian
0x09	English	0x19	Russian
0x0A	Spanish	0x1A	Croatian/Serbian
0x0B	Finnish	0x1B	Slovak
0x0C	French	0x1C	Albanian
0x0D	Hebrew	0x1D	Swedish
0x0E	Hungarian	0x1E	Thai
0x0F	Icelandic	0x1F	Turkish
0x10	Italian		

### DMS.AddRel

#### Beschreibung:

Dieser Job erstellt eine Relation zwischen zwei Objekten.

#### Parameter:

Flags (INT): z. Z. nicht unterstützt -> 0 übergeben

RelationXML (BASE64): Relationseigenschaften im XML-Format

**Rückgabewerte:**

RelID (STRING): GUID, die als ID für diese Verknüpfung erzeugt wurde

**Beispiel:**

Aufbau von RelationXML (Beschreibung: siehe [XMLRELATION](#))

```
<Relation obj_id1="1" obj_typ1="2" obj_id2="3" obj_typ2="4"
valid_from="3443234" valid_to="433444"
ensured="1" checked="0"
reltextid="AF32988234DEFA78979879"
IObjId="9" IObjType="10" >
</Relation>
```

**Siehe auch:**

[DMS.AddRelText](#), [DMS.AddRelTextLang](#)

## DMS.AddRelText

**Beschreibung:**

Dieser Job erstellt einen Relationstext in der Defaultsprache.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelTextXML (BASE64): Relationstext und Relationseigenschaften im XML-Format

**Rückgabe:**

RelTextID (STRING): GUID, die als ID für diesen Relationseintrag definiert wurde

**Beispiel:**

Aufbau von RelTextXML (Beschreibung: siehe [XMLRELTEXT](#))

```
<RelText shortrel="Eigentum" active="1" objtyp1="2" objtype2="4"
to="ist Eigentümer von" reverse="ist Eigentum von">
</RelText>
```

## DMS.AddRelTextLang

**Beschreibung:**

Dieser Job fügt einem bestehenden Relationstexteintrag einen Relationstext in einer anderen Sprache hinzu.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelTextXML (BASE64): Relationstext und Relationseigenschaften im XML Format

**Rückgabe:**

RelTextID (STRING): GUID, die als ID für diesen Relationseintrag definiert wurde

**Beispiel:**

Aufbau von RelTextXML (Beschreibung: siehe [XMLRELTEXT](#))

```
<RelText reltextid="ABC79324DEF879342" langid="9" to="is owner"
reverse="is owned by">
</RelText>
```

**Siehe auch:**[DMS.AddRelText](#)

## DMS.DeIRel

**Beschreibung:**

Dieser Job löscht die angegebene Relation.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelID (STRING): GUID der zu löschenden Relation

**Siehe auch:**[DMS.DelRelText](#)

## DMS.DeIRelText

**Beschreibung:**

Dieser Job löscht einen Relationstext. Existieren noch Referenzen auf den Relationstext, wird ein Fehler generiert.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelTextID (STRING): GUID des zu löschenden Relationstextes

LangID (INT): Sprachcode des zu löschenden Textes (s. [Languagecodes](#)) (0 = Relationstexte aller Sprachen werden gelöscht)

## DMS.ModRel

**Beschreibung:**

Dieser Job ändert die Eigenschaften eines Relationseintrages. Folgende [Relationsattribute](#) können geändert werden:

§ reltextid

§ ensured

§ checked

§ valid\_from

§ valid\_to

§ iobjid

§ iobjtype

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelationXML (BASE64): Relationen und die zu ändernden Eigenschaften im XML Format

**Beispiel:**

Aufbau von RelationXML (Beschreibung: siehe [XMLRELATION](#))

```
<DMSRelations>
<Relation relid="D2A0988234DEFA78979879" valid_from="3443234"
valid_to="433444" ensured="1" checked="8"
reltextid="AF32988234DEFA78979879" IObjId="9" IObjType="10">
</Relation>
</DMSRelations>
```

## DMS.ModRelText

### Beschreibung:

Dieser Job ändert die Eigenschaften eines oder mehrerer Relationstexte. Soll der Text in einer anderen als der Defaultsprache verändert werden, so muss der Job [DMS.ModRelTextLang](#) gewählt werden.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelTextXML (BASE64): Relationstext im XML-Format

### Beispiel:

Aufbau von RelTextXML (Beschreibung: siehe [XMLRELTEXT](#))

```
<RelText reltextid="ABC79324DEF879342" shortrel="Eigentum" active="1"
objtype1="2" objtype2="4" to="ist Eigentümer von" reverse="ist
Eigentum von">
</RelText>
```

## DMS.ModRelTextLang

### Beschreibung:

Dieser Job ändert den Relationstext für die im XML jeweils festgelegte Sprache.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelTextXML (BASE64): Relationstext im XML-Format

### Beispiel:

Aufbau von RelTextXML (Beschreibung: siehe [XMLRELTEXT](#))

```
<RelText reltextid="ABC79324DEF879342" langid="9" to="is owner"
reverse="is owned by">
</RelText>
```

## DMS.RetrieveRelations

### Beschreibung:

Mit diesem Job lassen sich Relationen recherchieren. Die Suchkriterien werden über die XML Attribute definiert. Bei den Zeitattributen created,modified und deletetime, lässt sich durch die Verwendung der XML Attribute created\_to, modified\_to und deletetime\_to eine obere Grenze angeben. Es wird dann über den jeweilig angegebenen Zeitraum gesucht. Die Verwendung von -1 bedeutet, dass die obere bzw. untere Grenze offen gelassen werden soll.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelationXML (BASE64): Suchparameter im XML-Format

Created\_to (INT): Suchparameter zur Eingrenzung des Erzeugungszeitraums

Modifytime\_to (INT): Suchparameter zur Eingrenzung des Zeitraums der letzten Änderung

Deleted\_to (INT): Suchparameter zur Eingrenzung des Zeitraums der Löschung

Direction (INT): Flag, das angibt, ob auch in der Rückrichtung gesucht werden soll, wenn im XML Objekttyp1 und/oder Objekttyp2 angegeben sind. 0 = unidirektionale, 1=bidirektionale Suche.

#### Rückgabe:

RelationsXML (BASE64): alle Relationen im XML-Format, welche die Suchkriterien erfüllen

#### Beispiel:

Aufbau von RelationXML (Beschreibung: siehe [XMLRELATION](#))

```
<Relation obj_id1="" obj_typ1="" obj_id2="" obj_typ2="" valid_from=""
valid_to="" ensured="" checked="" reltextid="" created="-1" created_to="4322344"
IObjId=""
IObjType="" >
</Relation>
```

#### Beispiel:

Aufbau von RelationsXML (Beschreibung: siehe [XMLRELATION](#))

```
<Relations>
<Relation obj_id1="" obj_typ1="" obj_id2="" obj_typ2=""
valid_from="" valid_to="" ensured="" checked="" reltextid=""
IObjId="" IObjType="" >
</Relation>
<Relation obj_id1="" obj_typ1="" obj_id2="" obj_typ2=""
valid_from="" valid_to="" ensured="" checked="" reltextid=""
IObjId="" IObjType="" >
</Relation>
</Relations>
```

## DMS.RetrieveRelTexts

#### Beschreibung:

Dieser Job liefert alle Relationstexte, die der übergebenen Suchkriterien entsprechen.

#### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

RelTextXML (BASE64): Suchkriterien im XML-Format

#### Rückgabewerte:

RelTextsXML (BASE64): Liste aller Relationstexte, die den Suchkriterien entsprechen im XML-Format

#### Beispiel:

Aufbau von RelTextXML (Beschreibung: siehe [XMLRELTEXT](#))

```
<RelText shortrel="" active="" objtypel="" objtype2="" to="" reverse=""
langid="" />
```

#### Beispiel:

Aufbau von RelTextsXML (Beschreibung: siehe [XMLRELTEXT](#))

```
<RelTexts>
<RelText reltextid="88E28DCDA7414D0B82AC2AFD0CB354EC"
shortrel="Eigentum" langid="7"
```

```

active="1" objtype1="2" objtype2="4" to="ist Eigentümer von"
reverse="ist Eigentum von"></RelText>
<RelText reltextid="79E28DCDA7414D0B82AC2AFDASB354EC"
shortrel="Dokument" langid="7"
active="1" objtype1="5" objtype2="7" to="ist Dokument von"
reverse="ist Dokument von"></RelText>
</RelTexts>

```

## Mappen (Portfolios)

§ [DMS.AddPortfolio](#)

§ [DMS.DelPortfolio](#)

§ [DMS.ModPortfolio](#)

§ [DMS.RemoveFromPortfolio](#)

§ [DMS.RetrievePortfolios](#)

## Ausführliche Beschreibung

Diese Jobs dienen den Anfragen und Bearbeiten von Mappen.

### Portfolio XML Format

#### Beispiel:

```

<Portfolios>
<Portfolio id="123" created="135233432" creator=""
recipient="" subject="">
<Objects>
<Object objecttype_id="13072" id="12">
</Object>
<Object>
</Object>
</Objects>
</Portfolio>
</Portfolios>

```

### Beschreibung der Elemente und Attribute

**Element 'Portfolios':** Liste von Mappen (Das Wurzelement kann entfallen, wenn nur ein Eintrag definiert wurde.)

**Attribute des Elements 'Portfolio':**

§ id (long): ID der Mappe

§ created (long): Zeitstempel der Erzeugung

§ creator (STRING): Name des Anlegers

§ recipient (STRING): Empfänger

§ subject (STRING): Titel der Mappe

**Element 'Objects':** Liste von Objekten (Ordner, Register, Dokumente), die die Mappe beinhaltet

**Attribute des Elements 'Object':**

§ objecttype\_id (LONG): Objekttyp

§ id (LONG): ID der Objektinstanz



## DMS.AddPortfolio

### Beschreibung:

Dieser Job legt eine neue Mappe an.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

PortfolioXML (BASE64): Beschreibung der Mappe im XML-Format

Mode (INT): 0 (Default)=Sucht nach vorhandenen Mappen anhand des Empfängers oder Betreffs.  
Sind eine oder mehrere Mappen vorhanden, werden diese beziehungsweise die erste Mappe verwendet.  
1=Legt eine neue Mappe an.

### Rückgabewerte:

PortfolioID (INT): ID der neuen Mappe (-1 = Job fehlgeschlagen)

### Beispiel:

Aufbau von PortfolioXML (Beschreibung: siehe [Portfolio XML Format](#))

```
<Portfolios>
<Portfolio created="" creator="" recipient=""
subject="">
</Portfolio>
</Portfolios>
```

## DMS.DeletePortfolio

### Beschreibung:

Dieser Job löscht eine Mappe. Die zu löschende Mappe kann entweder über die Mappen ID oder über Empfänger (recipient/recipient\_id) und den Titel (subject) der Mappe identifiziert werden.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

PortfolioXML (BASE64): Beschreibung der Mappe im XML-Format

### Beispiel:

Aufbau von PortfolioXML (Beschreibung: siehe [Portfolio XML Format](#))

```
<Portfolio id="">
</Portfolio>

<!--oder-->

<Portfolio recipient="" subject="">
</Portfolio>
```

## DMS.RemoveFromportfolio

### Beschreibung:

Dieser Job löscht Objekte in der Mappe.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

PortfolioXML (BASE64): Beschreibung der Mappe im XML-Format

**Beispiel:**

Aufbau von PortfolioXML (Beschreibung: siehe [Portfolio XML Format](#))

```
<Portfolio id="" created="" creator="" recipient=""
subject="">
<Objects>
<Object objecttype_id="" id="">
</Object>
</Objects>
</Portfolio>
```

## DMS.ModPortfolio

**Beschreibung:**

Dieser Job fügt einer bestehenden Mappe Objekte hinzu oder löscht alle Objekte in der Mappe.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Mode (INT): 1 = alle Objekte, die die Mappe beinhaltet werden gelöscht, ansonsten 0

PortfolioXML (BASE64): Beschreibung der Mappe im XML-Format

**Beispiel:**

Aufbau von PortfolioXML (Beschreibung: siehe [Portfolio XML Format](#))

```
<Portfolio id="" created="" creator="" recipient=""
subject="">
<Objects>
<Object objecttype_id="" id="">
</Object>
</Objects>
</Portfolio>
```

## DMS.RetrievePortfolios

**Beschreibung:**

Dieser Job liefert alle Mappen, die der angegebenen Suchkriterien entsprechen.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Created\_to (Long): Zeitsempel, bis zu dem die Mappe erstellt wurde. 0=Uneingeschränkt

PortfolioXML (BASE64): Suchkriterien für Mappe im XML-Format

GarbageMode (INT): (Optional) 1=Nur Objekte aus dem Papierkorb berücksichtigen. 0=Objekte aus dem Papierkorb nicht berücksichtigen.

**Rückgabewerte:**

PortfoliosXML (BASE64): Liste aller gefundenen Mappen im XML-Format

**Beispiel:**

Aufbau von PortfolioXML (Beschreibung: siehe [Portfolio XML Format](#))

```
<Portfolio id="" created="" creator="" recipient=""
subject="">
</Portfolio>
```

## Benutzerbezogene Daten

§ [DMS.DeleteUserData](#)

§ [DMS.GetUserData](#)

§ [DMS.GetUserDataNames](#)

§ [DMS.IsUserData](#)

§ [DMS.SetUserData](#)

### Ausführliche Beschreibung

Die Jobs dieses Bereiches dienen der Verwaltung beliebiger benutzerbezogener Daten. Jeder Dateneintrag ist durch einen frei wählbaren Namen (maximale Länge 100), einen Typ (s.u.) und der Benutzer ID gekennzeichnet. Die Benutzer ID wird von den Jobs automatisch aus der ID des angemeldeten Benutzers bestimmt. Der Wert des Eintrags wird in ein BLOB Feld geschrieben, kann also beliebigen Inhalt haben.

Mit Hilfe des Typs werden die Art der Daten definiert. Die unten stehende Tabelle (Stand: 18.03.2004) gibt eine Übersicht über die bisher vergebenen Nummern.

**Achtung:** Der Typ ist nicht frei wählbar. Wird ein neuer Typ benötigt, so muss dieser zunächst registriert werden.

Typ	Beschreibung
1	gespeicherte Anfragen
2	Externe Programme
3	AS.INI
4	Ordnerstruktur unter 'Desktop'
5	Erweiterte Anfragen
6	Daten aus jetziger aslisten.dat
7	AS-Objekt im Desktopbereich
8	Einträge für Listenfelder
9	reserviert
10	Importkonfiguration dBase III
12	Importkonfiguration dBase IV
13	Importkonfiguration dBase V
13	Importkonfiguration ASCII mit Trennzeichen
14	Importkonfiguration ASCII feste Feldlänge
15	reserviert
16	reserviert
17	Importkonfiguration XML linear
18	Importkonfiguration MS Access

19	Importkonfiguration Excel 3
20	Importkonfiguration Excel 4
21	Importkonfiguration Excel 5
22	Importkonfiguration Excel 8
23	externe Importkonfiguration
24	Importkonfiguration ODBC
25-30	reserviert für Importkonfigurationen
31	Benutzer/Gruppen Exportkonfigurationen
32	Benutzer/Gruppen Importkonfigurationen
33	Konfiguration autom. löschen von Benutzerpapierkörben
50	Client/Index Scankonfiguration
51	Stempelvorlagen für Folien
52	Nächste Position für öffentliche gespeicherte Anfragen
53	Favoriten für WF Addon
54	Einstellungen für Window-Gadget
80-85	Benutzerbezogene Konfigurationsinformationen für Zusatzanwendungen

## DMS.DeleteUserData

### Beschreibung:

Dieser Job löscht den über den Namen und Typ festgelegten benutzerbezogenen Datensatz.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Type (INT): siehe [Datentypen](#)

Name (STRING): Bezeichner des Datensatzes

## DMS.GetUserData

### Beschreibung:

Dieser Job liest benutzerbezogene Daten für den angegebenen Namen und Typ aus der Datenbank.

### Parameter:

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Type (INT): siehe [Datentypen](#)

Name (STRING): Bezeichner für den Datensatz

### Rückgabewerte:

IsUserData (INT): 1 = wenn der angeforderte Eintrag existiert, ansonsten 0

Value (BASE64): angeforderter Wert

**Siehe auch:**

[DMS.GetUserDataNames](#) , [DMS.SetUserData](#) , [DMS.DeleteUserData](#)

## DMS.GetUserDataNames

**Beschreibung:**

Dieser Job listet für den angemeldeten Benutzer die Namen aller Einträge für den gegebenen Typ auf.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Type (INT): siehe [Datentypen](#)

**Rückgabewerte:**

Name[1..n] (STRING): Name des Eintrags

**Siehe auch:**

[DMS.GetUserData](#)

## DMS.IsUserData

**Beschreibung:**

Dieser Job prüft, ob für den angegebenen Namen und Typ bereits ein Eintrag existiert. Standardmäßig wird die Existenz nur für den Benutzer geprüft. Soll die Prüfung benutzerunabhängig – also nur für den Typ – durchgeführt werden, muss das entsprechende Flag gesetzt werden.

**Parameter:**

Flags (INT): 1 = die Existenz des Eintrags wird benutzerunabhängig geprüft, ansonsten 0

Type (INT): siehe [Datentypen](#)

Name (STRING): Bezeichner für den Datensatz

**Rückgabewerte:**

IsUserData (INT): 1 = wenn der angeforderte Eintrag existiert, ansonsten 0

## DMS.SetUserData

**Beschreibung:**

Dieser Job speichert benutzerbezogene Daten in der Datenbank. Existiert noch kein Eintrag für den angegebenen Namen und Typ, wird ein neuer Eintrag erzeugt. Die Prüfung, ob ein Eintrag bereits existiert, kann über den Job [DMS.IsUserData](#) geschehen

**Parameter:**

Flags (INT): reserviert, muss 0 sein

Type (INT): siehe [Datentypen](#)

Name (STRING): Bezeichner für den Datensatz

Value (BASE64): zu speichernder Wert

## DMS.GetXMLSchema

### **Beschreibung:**

Dieser Job liefert das angegebene XML-Schema als Datei zurück.

### **Parameter:**

Flags (INT): muss 0 sein

Schema (STRING): Name des Schemas

§ DMSData = Schema für Import

§ DMSQuery = Schema für DMS Anfragen

§ DMSContent = Schema für DMS Ergebnisse

§ DMSAccess = Schema für [SSOL](#) Jobs

§ DMSObjDef = Schema für die Objektdefinition

§ Relation = Schema für Relationen

§ RelText = Schema für Relationstexte

§ Portfolio = Schema für Portfolios (Mappen)

### **Rückgabewerte:**

Dateiliste: enthält Pfad und Name der XSD-Datei

## Medizin-Engine (Namespace med)

Die Medizin-Engine ermöglicht den Zugriff auf medizinische Informationen. Die derzeitige Aufgabe der Jobs besteht in der Zusammenfassung und Rückgabe von Laborwerten, basierend auf dem LOINC-System (siehe <http://www.loinc.org/>).

Für die Jobs SaveMedicalRecord, GetMedicalRecord und NotifyMedicalRecord wird auf die Spezifikation HL7 Version 3 verwiesen (siehe <http://www.hl7.org/>).

- § [LoincResults](#)
- § [LoincObservations](#)
- § [LoincUnits](#)
- § [LoincViewSets](#)
- § [PatientData](#)
- § [CreateLaboratoryReport](#)
- § [UpdatePatientId](#)
- § [UpdateVisitId](#)
- § [ObservationInsert](#)
- § [ObservationRequestHistory](#)
- § [ObservationResultHistory](#)
- § [ObservationValues](#)
- § [SaveMedicalRecord](#)
- § [GetMedicalRecord](#)
- § [NotifyMedicalRecord](#)
- § [GetSystemOID](#)

### Feldbezeichner

Die einzelnen Feldbezeichner sind in Gruppen organisiert. Diese Gruppen entsprechen derzeit den Tabellen in der zugrunde liegenden Datenbank. Werden in einem Job für den Parameter 'FieldSelectors' ein oder mehrere Feldbezeichner angegeben, werden nur diese berücksichtigt. Wird kein Feldbezeichner angegeben, werden alle Werte dieser Gruppe zurückgeliefert. Die einzelnen Feldbezeichner liegen im Ergebnisdokument jeweils auf einer Ebene des XML-Ergebnisses.

### Gruppe Observations

In dieser Gruppe werden die Stammdaten aller Untersuchungen wie z. B. Labor- und Vitalwerte oder EKG, zusammengefasst. Die Kodierung erfolgt lt. LOINC-Standard! (führende Observation-ID->LoincID). Auf die Observations wird bei den ObservationResults (über die LOINC-ID) referenziert.

FieldSelector	Beschreibung
ObsID	LOINC-ID
ObsLastupdatedTS	Zeitpunkt der letzten Änderung in Form eines Zeitstempels
ObsName	Untersuchungsbezeichnung

ObsStdValType_ID	Default-Datenformat des Untersuchungsergebnisses (-> mdObsValueTypes)
ObsStdUnit_ID	standardmässig benutzte Masseinheit (-> Units)
ObsStdRefRange	Normwertbereich (= Wertebereich zwischen OBS_STDMINVALUE und OBS_STDMAXVALUE) bezogen auf die Standard-Masseinheit (nur relevant für numerische ValueTypes obsOvtObjectID) Alternative Namen aus der LOINC-Standard-DB (separiert durch ;)
ObsStdMinValue	untere Normbereich-Grenze bezogen auf die Standard-Masseinheit (nur relevant für numerische ValueTypes obsOvtObjectID)
ObsStdMaxValue	obere Normbereich-Grenze bezogen auf die Standard-Masseinheit (nur relevant für numerische ValueTypes obsOvtObjectID)
ObsPosition	Anzeige-Position (Spez. Verwendung in axvbPatCurve.dll). Durchnummerierung in 10er-Schritten
ObsColor	Anzeige-Farbe als Hex-RGB-Wert (Spez. Verwendung in axvbPatCurve.dll)
ObsScsClass	Untersuchungs-Klassifizierung nach dem SCS-System
ObsLnComponent	1. Teil des LoincNamens. Syntax: [analyte].[subclass].[sub-subclass]^[time delay] post [amount] [substance] [route]^adjustment. <b>Hinweis:</b> LoincName=[LnComponent]:[LnProperty]:[LnTimeAspct]:[LnSystem]:[LnScaleType]:[LnMethodType]
ObsLnProperty	2. Teil des LoincNamens. Physikalische MessGrösse. Korrespondiert zu LOINC-Tabelle Properties.
ObsLnTimeAspct	3. Teil des LoincNamens. MessSzenario. Syntax: [TimeAspect]^[Modifier] 1st Part korrespondiert zu LOINC-Tabelle TimeAspects.
ObsLnSystem	4. Teil des LoincNamens. UntersuchungsObjekt. Syntax: [System]^[Supersystem] Wenn Supersystem nicht explizit kodiert, dann gilt Default-Supersystem=Patient
ObsLnScaleType	5. Teil des LoincNamens. Skalentyp (kontinuierlich oder diskontierlich). Korrespondiert zu Tabelle ScaleTypes
ObsLnMethodType	6. Teil des LoincNamens. MessMethode. Codes sind selbsterklärend, ansonsten siehe Abkürzungen Table 14 in LOINCManual.pdf
ObsLnRelatedNames	originale alternative Namen aus der LOINC-Standard-DB (separiert durch ;)
ObsLnClass	LOINC-Klassifizierung. Korrespondiert zu LOINC-Tabelle Classes-ClassTypes.
ObsLnClassType	Korrespondiert zu LOINC-Tabelle ClassTypes: 1 - Laboratory Class, 2 - Clinical Class, 3 - Claims Attachments, 4 - Surveys
ObsIUPAC	Alternative IUPAC-Codierung
ObsMolarMass	Molare Masse (nur relevant bei Molekülbestimmungen)



### Gruppe Obsrequests

In dieser Gruppe werden die Daten der Laboranfrage (Request) zusammengefasst. Einzelne Requests sind immer Aufgehalten eines Patienten untergeordnet. Die Ergebnisse sind in `tdObservationResults` zu finden. Von Subsystemen per HL7 übermittelte Ergebnisse (z. B. Laborbefunde) sind als in Kraft einzufügen. Datenänderungsmöglichkeiten hängen von State ab.

FieldSelector	Beschreibung
ObrID	eindeutige ObjectID im Sinne eines PrimaryKeys (GUID)
ObrLastUpdatedTS	Zeitstempel der letzten Änderung
ObrState	derzeit ungültig
ObrCreationDT	Erstellungszeitpunkt
ObrCreationUser	Anwender, der erstellt hat
ObrReleaseDT	Zeitpunkt des In-Kraft-Setzens
ObrCancelDT	Stornierungszeitpunkt
ObrCancelUser	Anwender, der storniert hat
ObrCancelReason	Stornierungsgrund
ObrCommonOrder_ID	Verweis auf den optional zugehörigen Auftrag (-> <code>tdCommonOrders</code> ). Achtung: Für diese Relation wird keine referentielle Integrität erzwungen!
ObrPlacerOrderID	Auftragsnr. des Auftraggebers (Nur relevant in Verbindung mit Order/Entry <code>tdCommonOrders</code> )
ObrFillerOrderID	Bearbeitungsnr. der Leistungsstelle und dient als HL7-Schlüsselfeld zur Identifikation einer Untersuchung
ObrObservationDT	Zeitpunkt der Untersuchung bzw. der Probenentnahme
ObrObsOrderSet_ID	Anforderungsset-LOINC-ID (-> <code>mdObsOrderSets</code> ). Repräsentiert ein oder mehrere Teiluntersuchung(en). Letzteres wird auch als Profil, Set oder Sammelverfahren bezeichnet (z. B. Grosses Blutbild).
ObrArchiveDocID	Verweis auf das Befunddokument im Archiv (ObjectID)

### Gruppe Obsresults

In dieser Gruppe werden Untersuchungsergebnisse bzw. Befunddaten zusammengefasst. Dieses Objekt ist einer/m Untersuchungsanforderung/bericht untergeordnet, das auch den Untersuchungszeitpunkt enthält. Achtung: Datenänderungsmöglichkeiten hängen vom State des übergeordneten `ObservationRequest` ab!

FieldSelector	Beschreibung
ObxID	eindeutige ObjectID im Sinne eines PrimaryKeys (GUID)
ObxLastUpdatedTS	Zeitstempel der letzten Änderung

ObxState	Freigabestatus -> moReleaseStateEnum
ObxObservation_ID	LOINC-ID der Untersuchung (-> mdObservations)
ObxObservation_CE	Untersuchungsbezeichnung als Kopie HL7-CodedElement: identifizier^text^coding system^Alternate identifier^alternate text^alternate coding system
ObxSubObs_ID	
ObxValueType_ID	Datenformat des Feldes Value (-> mdObsValueTypes)
ObxValue	Ergebnis bzw. Messwert gemäss Format obxOvtObjectID
ObxUnit_ID	benutzte Masseinheit (-> Units)
ObxRefRange	Referenzbereich als Text im Original-HL7-Format (siehe OBX-7)
ObxMinValue	untere Normbereich-Grenze (nur relevant für numerische ValueTypes obxOvtObjectID)
ObxMaxValue	obere Normbereich-Grenze (nur relevant für numerische ValueTypes obxOvtObjectID)
ObxAbnormFlag_ID	Bewertung Value bezgl. des Normbereichs. (-> mdObsAbnormalFlags). Achtung: Bei einem über HL7 importierten ObsResult wird nur das erste übermittelte AbnormalFlag übernommen.
ObxResultState_ID	Befundstatus (-> mdObsResultStates)
ObxFooterID	Anwenderkommentar: Fussnote-ID (spez. Verwendung in axvbPatCurve.dll)
ObxFooterText	Anwenderkommentar: Fussnote-Text (spez. Verwendung in axvbPatCurve.dll)
ObxComment	Anwenderkommentar (spez. Verwendung in axvbPatCurve.dll)
ObxCreationDT	Erstellungszeitpunkt
ObxCreationUser	Ersteller
ObxReleaseDT	Zeitpunkt des In-Kraft-Setzens
ObxCancelDT	Stornierungszeitpunkt
ObxCancelUser	Anwender, der storniert hat
ObxCancelReason	Stornierungsgrund

### Gruppe Units

In dieser Gruppe werden die für die Quantifizierung von Untersuchungen (tdObservationResults, mdObservations) verwendeten Maßeinheiten zusammengefasst. (->Siehe auch Zusammenstellung in Tabelle \_PropertiesAndUnits\_ der LOINC-MDB). Umrechnungen stehen in Tabelle Conversions. 'ID' = HL7-Bezeichnung

FieldSelector	Beschreibung
UntID	ID gemäss HL7 2.4 Spezifikation
UntLastupdatedTS	Zeitstempel der letzten Änderung

UntName	ausführliche Bezeichnung der Masseinheit
UntAbbreviation	gängige Abkürzung der Einheit, wie sie dem Benutzer angezeigt werden soll

### Gruppe ViewSets

Bei der Gruppe der Viewsets handelt es sich um eine hierarchische Abbildung der Viewsets mit den dazugehörigen Views. Alle Bezeichner, die mit Ovs beginnen, gehören dem Viewset an. Die Bezeichner, die mit Ovd beginnen, sind die untergeordneten Views. ViewSets: Enthält für die Anzeige in axvbPatCurve.dll zusammengestellten Untersuchungs-Sets. Ein Satz kann ein oder mehrere Einzeluntersuchungen repräsentieren. Die mit dem Set verknüpften Untersuchungen sind der mmt\_obsviewingsetdt zu entnehmen. Views: Enthält die zu einem Anzeigeset (mdObsViewingSets) zugehörigen 1..N Einzeluntersuchungen.

FieldSelector	Beschreibung
OvsID	eindeutige ObjectID im Sinne eines PrimaryKeys (GUID)
OvsLastUpdatedTS	Zeitstempel der letzten Änderung
OvsName	Bezeichnung des Anzeigesets
OvdID	eindeutige ObjectID im Sinne eines PrimaryKeys (GUID)
OvdLastUpdatedTS	Zeitstempel der letzten Änderung
OvdPosition	Anzeigeposition (1..N) der Untersuchung
OvdObservationID	Untersuchungs-LOINC-ID (-> mdObservations)

### med.LoincResults

#### Beschreibung:

Dieser Job ermittelt die Werte des LOINC-Systems für Patienten und ihre Aufenthalte. Die möglichen Filter werden unter den dazugehörigen Eingangsparametern beschrieben. Alle Parameter werden UND-verknüpft. Werden keine Parameter angegeben, liefert der Aufruf alle Ergebnisse zurück.

Sortierung der Ergebnisse:

1. PatientenID
2. AccObjectID
3. VisitID
4. ObservationDT
5. CreationDT

#### Parameter:

[Visits] (STRING): enthält eine Komma-separierte Liste von IDs für die Aufenthalte (s. Namespace)

[FieldSelectors] (STRING): enthält eine Komma-separierte Liste der angeforderten Felder; (siehe: [Feldbezeichner](#))

[Namespace] (STRING): gibt an, aus welchem System die Werte für Patients und Visits stammen; Leer = beide Parameter werden erwartet

[ResultType] (STRING): gibt an, in welchem Format der Ausgabeparameter bereitgestellt werden (DEFAULT BASE64)

§ STRING = Ergebnis wird als String-Parameter zurückgeliefert

§ BASE64 = Ergebnis wird base64-kodiert zurückgeliefert

[Patients] (STRING): z.Z nicht implementiert (enthält eine Komma-separierte Liste von IDs für Patienten (s. Namespace))

[From] (STRING): z.Z nicht implementiert (wird From angegeben, werden nur die Ergebnisse ab (einschließlich) dem angegebenen Zeitpunkt zurückgeliefert. Berücksichtigt wird das ObservationDT des Requests. Format: MM/DD/YYYY)

[To] (STRING): z.Z nicht implementiert (wird To angegeben, werden nur die Ergebnisse bis zum angegebenen Zeitpunkt zurückgeliefert. Berücksichtigt wird das ObservationDT des Requests. Zu beachten ist, dass die Angabe des Datums in To die dazugehörige Uhrzeit automatisch auf 00:00 Uhr setzt. Besitzen die dazugehörigen Daten in der Datenbank jedoch eine Uhrzeit, werden diese nicht mit angezeigt, da diese Daten dann größer als das gesuchte Datum sind. Format: MM/DD/YYYY)

[Status] (STRING): z.Z nicht implementiert (enthält eine Komma-separierte Liste der Status. Es werden nur diejenigen LoincResults zurückgegeben, die einen der angegebenen Status besitzen. Wird Fields nicht angegeben, werden die LoincResults zurückgeliefert, die den Status 1 besitzen. mögliche Werte: (wird derzeit nur durch die Datenbankergebnisse geprüft)

§ 0 - ungültiger Datensatz

§ 1 - aktueller Datensatz

§ 2 - stornierter Datensatz

[Extremis] (STRING): z.Z nicht implementiert (min = liefert den jeweils kleinsten Wert für alle angeforderten Felder zurück; max = liefert den jeweils größten Wert für alle angeforderten Felder zurück)

### Rückgabewerte:

Result (STRING): liefert das Ergebnis der Anfrage als XML-Dokument zurück

### Beispiel:

Das folgende Beispiel wurde mit 'ObsID,ObrID,ObxID,ObxValue,UntID' als Wert in FieldSelectors abgerufen.

```
<med>
<Patients>
<Patient ID="252" CabinetType="3">
<Visit ID="253">
<Requests>
<OBR ID="E3C1787E-69C2-11D6-82E2-0000D19D9210">
<Results>
<OBX ID="E3C1787F-69C2-11D6-82E2-0000D19D9210" Value="28.0" />
<OBX ID="E3C1789A-69C2-11D6-82E2-0000D19D9210" Value="neg" />
<OBX ID="E3C1789B-69C2-11D6-82E2-0000D19D9210" Value="massh." />
<OBX ID="E3C1789C-69C2-11D6-82E2-0000D19D9210" Value="neg" />
<OBX ID="E3C1789D-69C2-11D6-82E2-0000D19D9210" Value="neg" />
<OBX ID="E3C1789E-69C2-11D6-82E2-0000D19D9210" Value="(+) " />
<OBX ID="E3C1789F-69C2-11D6-82E2-0000D19D9210" Value="Urate+" />
<OBX ID="E3C178A0-69C2-11D6-82E2-0000D19D9210" Value="neg" />
</Results>
</OBR>
<OBR ID="E3C18537-69C2-11D6-82E2-0000D19D9210">
<Results>
```

```
<OBX ID="E3C18538-69C2-11D6-82E2-0000D19D9210" Value="25.0" />
<OBX ID="E3C18539-69C2-11D6-82E2-0000D19D9210" Value="neg" />
<OBX ID="E3C1853A-69C2-11D6-82E2-0000D19D9210" Value="6.60" />
<OBX ID="E3C1853E-69C2-11D6-82E2-0000D19D9210" Value="87.8" />
<OBX ID="E3C1853F-69C2-11D6-82E2-0000D19D9210" Value="30.5" />
<OBX ID="E3C18540-69C2-11D6-82E2-0000D19D9210" Value="237" />
</Results>
</OBR>
<OBR ID="8775BE70-68EA-11D6-82E2-0000D19D9210">
<Results>
<OBX ID="8775BE72-68EA-11D6-82E2-0000D19D9210" Value="142" />
<OBX ID="8775BE73-68EA-11D6-82E2-0000D19D9210" Value="4.15" />
<OBX ID="8775BE74-68EA-11D6-82E2-0000D19D9210" Value="2.36" />
</Results>
</OBR>
<OBR ID="E3C19CDA-69C2-11D6-82E2-0000D19D9210">
<Results>
<OBX ID="E3C19CDB-69C2-11D6-82E2-0000D19D9210" Value="0.900" />
</Results>
</OBR>
</Requests>
</Visit>
</Patient>
</Patients>

<Observations>
<OBS ID="4537-7" />
<OBS ID="1988-5" />
<OBS ID="789-8" />
<OBS ID="5787-7" />
<OBS ID="5783-6" />
<OBS ID="5769-5" />
<OBS ID="8246-1" />
</Observations>
<Units>
<Unit ID="mg/dL" />
<Unit ID="Mill/cmm" />
<Unit ID="mm n.W." />
<Unit ID="mmol/L" />
<Unit ID="mmol/l" />
<Unit ID="pg" />
<Unit ID="U/l" />
<Unit ID="x1000/cmm" />
</Units>
<ViewSets>
<ViewSet OvsID="615B8200-1635-4D89-B5EC-4977802D7719"
OvsLastUpdatedTS="1013087583" OvsName="Set Cholangiolithiasis">
<View OvdID="642345256923659265956346543656239562"
OvdLastUpdatedTS="1013087623" OvdPosition="130"
OvdObservationID="1988-5" />
<View OvdID="385743657265762475643756783645726526"
OvdLastUpdatedTS="1013087623" OvdPosition="160"
OvdObservationID="2324-2" />
<View OvdID="298436375638567834568734568734567834"
OvdLastUpdatedTS="1013087623" OvdPosition="180"
OvdObservationID="3040-3" />
</ViewSet>
<ViewSet OvsID="00C00F47-176E-4A1E-B152-1D7C25865F71"
OvsLastUpdatedTS="1013087583" OvsName="Set Herzinfarkt">
<View OvdID="A795DBFE-4440-4248-A358-29D57DB02E90"
OvdLastUpdatedTS="1013087623" OvdPosition="10"
OvdObservationID="1920-8" />
<View OvdID="A8D5D3A1-4EB3-492C-9CAB-E7908D217784"
OvdLastUpdatedTS="1013087623" OvdPosition="20"
```

```
OvdObservationID="1677-4"/>
<View OvdID="345346536536534625762357632756327465"
OvdLastUpdatedTS="1013087623" OvdPosition="60"
OvdObservationID="6598-7"/>
<View OvdID="094632784523856827345823568235872385"
OvdLastUpdatedTS="1013087623" OvdPosition="170"
OvdObservationID="1798-8"/>
</ViewSet>
</ViewSets>
</med>
```

## med.LoincObservations

### Beschreibung:

Dieser Job ermittelt die Stammdaten der Observations des LOINC-Systems. Die möglichen Filter werden unter den dazugehörigen Eingangsparametern beschrieben. Alle Parameter werden UND-verknüpft. Werden keine Parameter angegeben, liefert der Aufruf alle Observations zurück.

### Parameter:

[Visits] (STRING): Komma-separierte Liste von IDs für die Aufenthalte (s. Namespace)

[Namespace] (STRING): gibt an, aus welchem System die Werte für Patients und Visits stammen; Leer = beide Parameter werden erwartet

[ObsIDs] (STRING): enthält eine Komma-separierte Liste der angeforderten Observations

[FieldSelectors] (STRING): enthält eine Komma-separierte Liste der angeforderten Felder; siehe [Feldbezeichner](#)

[ResultType] (STRING): gibt an, in welchem Format der Ausgabeparameter bereitgestellt werden (DEFAULT BASE64)

§ STRING = Ergebnis wird als String-Parameter zurückgeliefert

§ BASE64 = Ergebnis wird base64-kodiert zurückgeliefert

[Patients] (STRING): z.Z nicht implementiert (enthält eine Komma-separierte Liste von IDs für Patienten (s. Namespace))

[ViewSets] (STRING): z.Z nicht implementiert (wird ShowSets angegeben, werden für jede Observation die Sets mit ausgegeben, in denen sich die jeweilige Observation mit befindet (Standardwert: false))

### Rückgabewerte:

Result (STRING): liefert das Ergebnis der Anfrage als XML-Dokument zurück

## med.LoincUnits

### Beschreibung:

Dieser Job ermittelt die Stammdaten der Units des LOINC-Systems. Die möglichen Filter werden unter den dazugehörigen Eingangsparametern beschrieben. Alle Parameter werden über UND verknüpft. Werden keine Parameter angegeben, liefert der Aufruf alle Units zurück.

### Parameter:

[Visits] (STRING): Komma-separierte Liste von IDs für die Aufenthalte (s. Namespace).

[Namespace] (STRING): gibt an, aus welchem System die Werte für Patients und Visits stammen; Leer = beide Parameter werden erwartet

[UnitIDs] (STRING): enthält eine Komma-separierte Liste von IDs für die Units

[FieldSelectors] (STRING): enthält eine Komma-separierte Liste der angeforderten Felder; siehe: [Feldbezeichner](#)

[ResultType] (STRING): gibt an, in welchem Format der Ausgabeparameter bereitgestellt werden (DEFAULT BASE64)

§ STRING = Ergebnis wird als String-Parameter zurückgeliefert

§ BASE64 = Ergebnis wird base64-kodiert zurückgeliefert

[Patients] (STRING): z.Z nicht implementiert (enthält eine Komma-separierte Liste von IDs für Patienten (s. Namespace))

#### **Rückgabewerte:**

Result (STRING): liefert das Ergebnis der Anfrage als XML-Dokument zurück

### [med.LoincViewSets](#)

#### **Beschreibung:**

Dieser Job ermittelt die Stammdaten der ViewSets des LOINC-Systems.

#### **Parameter:**

[ResultType] (STRING): gibt an, in welchem Format der Ausgabeparameter bereitgestellt werden (DEFAULT BASE64)

§ STRING = Ergebnis wird als String-Parameter zurückgeliefert

§ BASE64 = Ergebnis wird base64-kodiert zurückgeliefert

#### **Rückgabewerte:**

Result (STRING): liefert das Ergebnis der Anfrage als XML-Dokument zurück

### [med.PatientData](#)

#### **Beschreibung:**

Dieser Job ermittelt die Daten eines oder mehrerer Patienten und liefert diese in einer vorgegeben Form, unabhängig vom Aufbau des Archives, zurück. Es wird damit dem Aufrufer die Möglichkeit gegeben, auf Patientendaten zuzugreifen, ohne die interne Struktur des Archivs zu kennen.

Die Daten, auf die dieser Job zugreift, können sich je nach Aufbau des Archivs, an unterschiedlichen Stellen befinden. Um sicherzustellen, dass immer die gleiche Ergebnisstruktur zurückgeliefert wird, wird ein externes Query (query\_patients.xml) und ein externes Stylesheet (query\_patients.xsl) verwendet. Diese beiden Dateien befinden sich im Serververzeichnis etc/med und müssen auf die jeweilige Struktur des Archives angepasst werden.

Bei dem Query handelt es sich um eine DMS-Executor-Anfrage, in die vom PatientData-Job die übergebenen Job-Parameter eingearbeitet werden. Der DMS-Executor berücksichtigt in der Anfrage nur die Bereiche, für die Param-Tags in der DMS-Anfrage angelegt wurden. Folgende DMS-Query-Param-Tags werden eingefügt:

§ PatientID - es wird der im PatientID-Parameter übergebene Wert eingefügt

§ Surname - es wird der im Surname-Parameter übergebene Wert eingefügt

§ Firstname - es wird der im Firstname-Parameter übergebene Wert eingefügt

§ Visit - es wird für jeden Wert in der im Firstname-Parameter übergebenen Liste ein Tag eingefügt

**Parameter:**

[PatientID] (STRING): enthält die Patienten-ID der angeforderten Daten, der Einsatz von Wildcard ist für diesen Parameter möglich (Bei der Patienten-ID handelt es sich um einen kundenspezifischen Wert, der nichts mit den internen IDs des Archiv-, bzw. Loinc-Systems zu tun hat.)

[Surname] (STRING): enthält den Nachnamen des angeforderten Patienten, der Einsatz von Wildcard ist für diesen Parameter möglich

[Firstname] (STRING): enthält den Vornamen des angeforderten Patienten, der Einsatz von Wildcard ist für diesen Parameter möglich

[Visits] (STRING): enthält eine komma-separierte Liste von IDs für die Aufenthalte (s. Namespace), der Einsatz von Wildcard ist für diesen Parameter möglich

[Namespace] (STRING): gibt an, aus welchem System die Werte für Visits stammen; Leer = beide Parameter werden erwartet

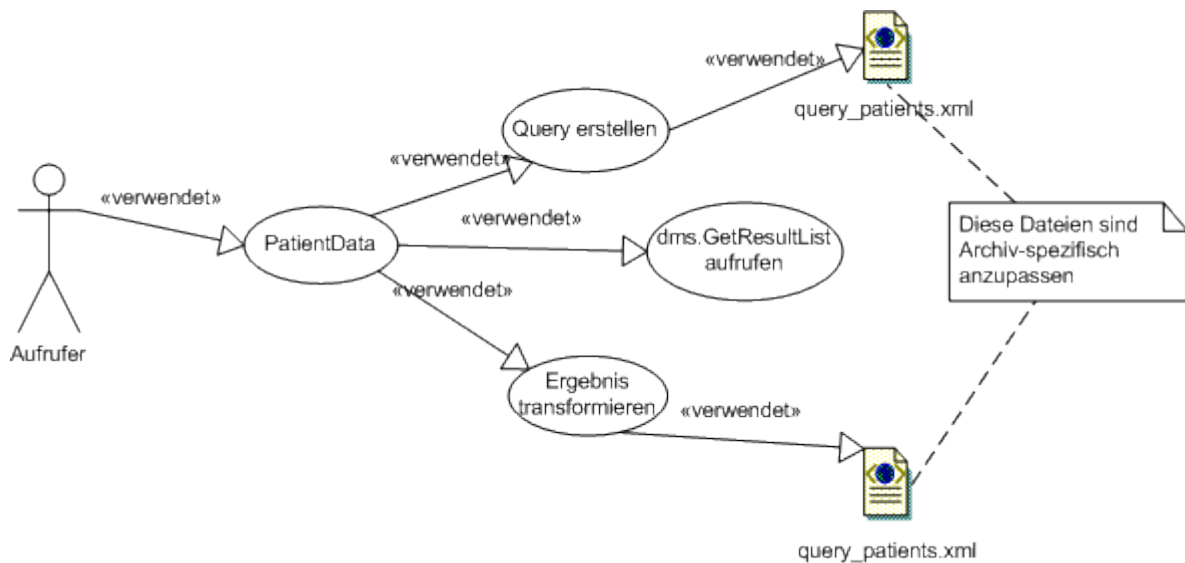
[ResultType] (STRING): gibt an, in welchem Format der Ausgabeparameter bereitgestellt werden (DEFAULT BASE64)

§ STRING = Ergebnis wird als String-Parameter zurückgeliefert

§ BASE64 = Ergebnis wird base64-kodiert zurückgeliefert

**Rückgabewerte:**

Result (STRING): liefert das Ergebnis der Anfrage als XML-Dokument zurück

**Interner Aufbau des Jobs:**

Die folgenden Beispiele zeigen den Aufbau der hinterlegten XSL-Dokumente und die dazu erzeugten XML-Dokumente.

**Hinterlegtes DMS-Query (query\_patients.xml):**

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<DMSQuery>
<Params/>
<Archive name="PATIENT">
<ObjectType name="Patient">
<Fields>
<Field name="Patienten-ID"/>
<Field name="Name"/>

```



```

<Field name="Vorname" />
</Fields>
<Conditions>
<ConditionObject name="Patient">
<FieldCondition name="Vorname">
<ParamValue ref="Firstname" />
</FieldCondition>
<FieldCondition name="Name">
<ParamValue ref="Surname" />
</FieldCondition>
<FieldCondition name="Patienten-ID">
<ParamValue ref="PatientID" />
</FieldCondition>
</ConditionObject>
<ConditionObject name="Aufenthalt">
<FieldCondition name="Fall-ID">
<ParamValue ref="Visit" />
</FieldCondition>
</ConditionObject>
</Conditions>
<ChildObjects child_schema="def" export_depth="5">
<SubObjectType name="Aufenthalt">
<Fields>
<Field name="Fall-ID" />
<Field name="Aufnahmedatum" />
</Fields>
</SubObjectType>
<SubObjectType name="Bewegung">
<Fields fields_schema="ALL" />
</SubObjectType>
</ChildObjects>
</ObjectType>
</Archive>
</DMSQuery>

```

### DMS-Query mit eingefügten Parametern zur Übergabe an den DMS-Executor:

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<DMSQuery>
<Params>
<Param name="Surname">Meier</Param>
<Param name="Firstname">Er*</Param>
<Param name="Visit">28*</Param>
</Params>
<Archive name="PATIENT">
<ObjectType name="Patient">
<Fields>
<Field name="Patienten-ID" />
<Field name="Name" />
<Field name="Vorname" />
</Fields>
<Conditions>
<ConditionObject name="Patient">
<FieldCondition name="Vorname">
<ParamValue ref="Firstname" />
</FieldCondition>
<FieldCondition name="Name">
<ParamValue ref="Surname" />
</FieldCondition>
<FieldCondition name="Patienten-ID">
<ParamValue ref="PatientID" />
</FieldCondition>
</ConditionObject>
<ConditionObject name="Aufenthalt">

```

```

<FieldCondition name="Fall-ID">
  <ParamValue ref="Visit"/>
</FieldCondition>
</ConditionObject>
</Conditions>
<ChildObjects child_schema="def" export_depth="5">
  <SubObjectType name="Aufenthalt">
    <Fields>
      <Field name="Fall-ID"/>
      <Field name="Aufnahmedatum"/>
    </Fields>
  </SubObjectType>
  <SubObjectType name="Bewegung">
    <Fields fields_shema="ALL" />
  </SubObjectType>
</ChildObjects>
</ObjectType>
</Archive>
</DMSQuery>

```

### Hinterlegtes Stylesheet (query\_patients.xml):

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:element name="PatientData">
      <xsl:apply-templates
select="DMSContent/Archive/ObjectType[@name='PATIENT']/ObjectList"
mode="Patient"/>
    </xsl:element>
  </xsl:template>
  <!--
Patientendaten ausgeben
-->
  <xsl:template match="Object" mode="Patient">
    <xsl:element name="Patient">
      <xsl:attribute name="OSID">
        <xsl:value-of select="@id"/>
      </xsl:attribute>
      <xsl:attribute name="Surname">
        <xsl:value-of select="Fields/Field[@name='Name']"/>
      </xsl:attribute>
      <xsl:attribute name="Firstname">
        <xsl:value-of select="Fields/Field[@name='Vorname']"/>
      </xsl:attribute>
      <xsl:attribute name="PatientID">
        <xsl:value-of select="Fields/Field[@name='Patienten-ID']"/>
      </xsl:attribute>
      <xsl:apply-templates
select="ChildObjects/ObjectType[@name='Aufenthalt']/ObjectList"
mode="Aufenthalt"/>
    </xsl:element>
  </xsl:template>
  <!--
Aufenthalte ausgeben
-->
  <xsl:template match="Object" mode="Aufenthalt">
    <xsl:element name="Aufenthalt">
      <xsl:attribute name="OSID">
        <xsl:value-of select="@id"/>
      </xsl:attribute>

```

```

<xsl:attribute name="Fall-ID">
<xsl:value-of select="Fields/Field[@name='Fall-ID']"/>
</xsl:attribute>
<xsl:attribute name="Aufnahmedatum">
<xsl:value-of select="Fields/Field[@name='Aufnahmedatum']"/>
</xsl:attribute>

<xsl:apply-templates
select="ChildObjects/ObjectType[@name='Bewegung']/ObjectList"
mode="Bewegung"/>
</xsl:element>
</xsl:template>
<!--
Bewegungen ausgeben
-->
<xsl:template match="Object" mode="Bewegung">
<xsl:element name="Bewegung">
<xsl:attribute name="OSID">
<xsl:value-of select="@id"/>
</xsl:attribute>
<xsl:attribute name="Zeitpunkt">
<xsl:value-of select="Fields/Field[@name='Zeitpunkt']"/>
</xsl:attribute>
<xsl:attribute name="Bewegungsart">
<xsl:value-of select="Fields/Field[@name='Bewegungsart']"/>
</xsl:attribute>
<xsl:attribute name="Patientenstatus">
<xsl:value-of select="Fields/Field[@name='Patientenstatus']"/>
</xsl:attribute>
<xsl:attribute name="KSt">
<xsl:value-of select="Fields/Field[@name='KSt']"/>
</xsl:attribute>
<xsl:attribute name="Fachabteilung">
<xsl:value-of select="Fields/Field[@name='Fachabteilung']"/>
</xsl:attribute>
<xsl:attribute name="interne-Bew-ID">
<xsl:value-of select="Fields/Field[@name='interne Bew-ID']"/>
</xsl:attribute>
<xsl:attribute name="externe-Bewegungs-ID">
<xsl:value-of select="Fields/Field[@name='externe Bewegungs-ID']"/>
</xsl:attribute>
<xsl:attribute name="Herkunft">
<xsl:value-of select="Fields/Field[@name='Herkunft']"/>
</xsl:attribute>
<xsl:attribute name="storniert">
<xsl:value-of select="Fields/Field[@name='storniert']"/>
</xsl:attribute>
</xsl:element>
</xsl:template>
</xsl:stylesheet>

```

### Erzeugtes Ergebnis:

```

<?xml version="1.0" encoding="UTF-16"?>
<PatientData>
<Patient OSID="575" Surname="Meier" Firstname="Erwin" PatientID="188123">
<Aufenthalt OSID="576" Fall-ID="287084" Aufnahmedatum="26.08.2003">
<Bewegung OSID="577" Zeitpunkt="20030826101449" Bewegungsart=""
Patientenstatus="I" KSt="" Fachabteilung="" interne-Bew-ID=""
externe-Bewegungs-ID="" Herkunft="" storniert="" />
</Aufenthalt>
</Patient>
</PatientData>

```

## med.CreateLaboratoryReport

### Beschreibung:

Dieser Job erstellt aus den vorhandenen Labordaten zu einem Aufenthalt, einen Laborbericht. Der Laborbericht wird als PDF Dokument erstellt und in dem Aufenthalt als Dokument abgelegt. Wenn dieser Job benutzt wird muss in der Objektdefinition der folgende Dokumenttyp vorhanden sein:

### Parameter:

[Visits] (STRING): Die ObjektID des Aufenthaltes in dem der Laborbericht erstellt werden soll

### Rückgabewerte:

[ObjectID] (INTEGER): ObjektID des Laborberichtes der dem Aufenthalt hinzugefügt wurde

### Interner Aufbau des Jobs:

Der Job ermittelt die Labordaten zu dem Aufenthalt über den Job med.ObservationValues, die Patientendaten werden über den Job med.PatientData ermittelt. Anschliessend wird eine XSL Transformation durchgeführt. Das entsprechende Stylesheet labreport.xslt liegt im Verzeichnis ./etc/med. Das Ergebnis dieser Transformation ist ein XSL:FO Datei. Diese Datei wird über den Job cnv.ConvertDocument in ein PDF Dokument konvertiert. Anschließend wird das PDF Dokument dem Aufenthalt, über den Job dms.XMLInsert, hinzugefügt.

## med.UpdatePatientId

### Beschreibung:

Dieser Job setzt die Patienten-ID für alle Einträge im Laborsystem mit der übergebenen PatientenID neu.

### Parameter:

[OldPatientID] (STRING): ObjektID des Patienten, die durch die neue ObjektID ersetzt werden soll

[NewPatientID] (STRING): neue ObjektID des Patienten, die die alte ObjektID ersetzen soll

[VisitID] (STRING): ObjektID des Aufenthaltes.

Wird diese ID angegeben, werden nur die PatientenIDs der Daten verändert, die auch die betreffende AufenthaltsID beinhalten.

### Rückgabewerte:

## med.UpdateVisitId

### Beschreibung:

Dieser Job setzt die ObjektID des Aufenthaltes für alle Einträge im Laborsystem mit der übergebenen ObjektID des Aufenthaltes neu.

**Parameter:**

[OldVisitID] (STRING): ObjektID des Aufenthaltes, die durch die neue ObjektID des Aufenthaltes ersetzt werden soll

[NewVisitID] (STRING): neue ObjektID des Aufenthaltes, die die alte ObjektID ersetzen soll

**Rückgabewerte:****med.ObservationInsert****Beschreibung:**

Fügt neue Laborwerte in das System ein. Über diesen Job ist es möglich, Loinc-Werte für Patienten transaktionssicher in das Laborsystem einzufügen. Leer übergebene Datums- und Zeitwerte werden als NULL in der Datenbank abgespeichert.

**Parameter:**

[Input] (BASE64): enthält die hinzuzufügenden Daten in einer XML-Struktur

Unterstützt werden folgende XML-Codierungen:

UTF-8 (mit und ohne BOM)

wird keine BOM für UTF-8 gefunden, wird überprüft, ob in den XML-Daten die Kennung UTF-8 vorhanden ist. Dies erfolgt derzeit nur über einen Stringvergleich!

iso-8859-1 Es wird in den XML-Daten nach der Kennung iso-8859-1 gesucht. Wird diese gefunden, werden die Daten als ASCII interpretiert. Dies erfolgt derzeit nur über einen Stringvergleich!

UTF-16-little endian (mit und ohne BOM)

Die Erkennung erfolgt in folgender Reihenfolge:

Überprüfung der UTF-16-BOM

Überprüfung der UTF-8-BOM

Überprüfung der XML-Daten auf die UTF-8-Kennung

Überprüfung der XML-Daten auf die ISO-8859-1-Kennung

alle anderen XML-Daten werden als UTF-16 interpretiert

Die Überprüfung auf die betreffende Kennung erfolgt kontextinsensitiv. Nicht unterstützte Formate werden als UTF16 interpretiert.

[Encoding] (STRING): gibt an, in welchem Encoding-Format das Rückgabe-XML geliefert werden soll. Wird dieser Parameter nicht angegeben oder ist der Parameter nicht unten aufgeführt, wird UTF-16 geliefert. Wird der Parameter ResultType mit dem Wert String aufgerufen, wird dieser Parameter nicht berücksichtigt und das Ergebnis mit einer UTF-16-Kennung zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

UTF-8 - Die Daten werden UTF-8-kodiert geliefert.

ASCII - Die Daten werden ASCII-kodiert geliefert (ISO-8859-1).

[ResultType] (STRING): Wird dieser Parameter mit dem Wert STRING angegeben, wird das Ergebnis als String- und nicht als base64-kodierter Parameter zurückgeliefert. Dieser Parameter sollte nur für Testzwecke verwendet werden. Wird dieser Parameter nicht angegeben, werden die Daten base64-kodiert zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgende Parameterwerte werden unterstützt:

STRING - Wird dieser Wert übergeben, wird das Ergebnis im Ausgabeparameter <Result> im String-Format geliefert.

BASE64- Wird dieser Wert übergeben, wird das Ergebnis base64-encoded zurückgeliefert.

#### **Rückgabewerte:**

[Result] (BASE64/STRING): liefert das um die zusätzlich vom Job erstellten Felder angereicherte XML zurück. Dieses enthält zusätzlich die erzeugten IDs, Statusinformationen und UpdateCounts.

## med.ObservationResultHistory

### **Beschreibung:**

Job zum Ermitteln der History eines Results (OBX)

Über diesen Job kann die Updategeschichte einzelner Ergebnisse nachverfolgt werden

Alle ermittelten Ergebnisse werden unter dem aktuellen Response(OBR) zurückgeliefert.

Die Sortierung erfolgt in folgender Reihenfolge:

Patienten-ID (obr\_PatObjectID)

Aufenthalts-ID (obr\_VisObjectID)

Request-ID (obr\_ID)

Resultstatus (obx\_State)

Result-Updatecount (umgekehrt) (obx\_UpdateCount DESC)

### **Parameter:**

[ResultIds] (STRING): IDs der Results (OBX), für die die Updategeschichte angezeigt werden soll Die einzelnen Results sind kommasepariert anzugeben. Bei diesen IDs handelt es sich um die IDs die über den Job med.ObservationValues zurückgeliefert werden. Nicht um die LOINC-IDs.

[ShowTestValues] (STRING): gibt an, ob die als Test markierten Werte zurückgegeben werden sollen bzw. die nicht als Test markierten. Wird dieser Parameter nicht angegeben, werden die nicht als Test markierten Daten zurückgeliefert. Für die Berücksichtigung müssen sowohl die Werte des Requests, als auch die Werte der Results die gleiche Einstellung besitzen! Wird dieser Parameter nicht angegeben, werden die Daten zurückgeliefert, die nicht als Test markiert wurden.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

true - es werden die als Test markierten Werte zurückgeliefert

false - es werden die nicht als Test markierten werte zurückgeliefert

[Encoding] (STRING): gibt an, in welchem Encoding-Format das Rückgabe-XML geliefert werden soll. Wird dieser Parameter nicht angegeben oder ist der Parameter nicht unten aufgeführt, wird UTF-16 geliefert. Wird der Parameter ResultType mit dem Wert String aufgerufen, wird dieser Parameter nicht berücksichtigt und das Ergebnis mit einer UTF-16-Kennung zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

UTF-8 - Die Daten werden UTF-8-kodiert geliefert.

ASCII - Die Daten werden ASCII-kodiert geliefert (ISO-8859-1).

[ResultType] (STRING): Wird dieser Parameter mit dem Wert STRING angegeben, wird das Ergebnis als String- und nicht als base64-kodierter Parameter zurückgeliefert. Dieser Parameter sollte nur für Testzwecke verwendet werden. Wird dieser Parameter nicht angegeben, werden die Daten base64-kodiert zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgende Parameterwerte werden unterstützt:

STRING - Wird dieser Wert übergeben, wird das Ergebnis im Ausgabeparameter <Result> im String-Format geliefert.

BASE64- Wird dieser Wert übergeben, wird das Ergebnis base64-encoded zurückgeliefert.

#### **Rückgabewerte:**

[Result] (BASE64/STRING): liefert die History der angefragten Results.

## **med.ObservationRequestHistory**

### **Beschreibung:**

Job zum Ermitteln der History eines Requests Über diesen Job kann die Updategeschichte einzelner Requests(OBRs) nachverfolgt werden

### **Parameter:**

[RequestIds] (STRING): IDs der OBRs, für die die Updategeschichte angezeigt werden soll Die einzelnen OBRs sind kommasepariert anzugeben. Eine Nachverfolgung ist nur für aktuelle OBRs möglich. Bei der Angabe von alten (bereits geupdateten) OBRs werden keine OBRs zurückgeliefert. Die jeweils aktuellen OBRs können mittels med.LoincValues unter Angabe des betreffenden Aufenthaltes ermittelt werden.

[ShowTestValues] (STRING): gibt an, ob die als Test markierten Werte zurückgegeben werden sollen bzw. die nicht als Test markierten. Wird dieser Parameter nicht angegeben, werden die nicht als Test markierten Daten zurückgeliefert. Für die Berücksichtigung müssen sowohl die Werte des Requests, als auch die Werte der Results die gleiche Einstellung besitzen! Wird dieser Parameter nicht angegeben, werden die Daten zurückgeliefert, die nicht als Test markiert wurden.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

true - es werden die als Test markierten Werte zurückgeliefert

false - es werden die nicht als Test markierten werte zurückgeliefert

[Encoding] (STRING): gibt an, in welchem Encoding-Format das Rückgabe-XML geliefert werden soll. Wird dieser Parameter nicht angegeben oder ist der Parameter nicht unten aufgeführt, wird UTF-16 geliefert. Wird der Parameter ResultType mit dem Wert String aufgerufen, wird dieser Parameter nicht berücksichtigt und das Ergebnis mit einer UTF-16-Kennung zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

UTF-8 - Die Daten werden UTF-8-kodiert geliefert.

ASCII - Die Daten werden ASCII-kodiert geliefert (ISO-8859-1).

[ResultType] (STRING): Wird dieser Parameter mit dem Wert STRING angegeben, wird das Ergebnis als String- und nicht als base64-kodierter Parameter zurückgeliefert. Dieser Parameter sollte nur für Testzwecke verwendet werden. Wird dieser Parameter nicht angegeben, werden die Daten base64-kodiert zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgende Parameterwerte werden unterstützt:

**STRING** - Wird dieser Wert übergeben, wird das Ergebnis im Ausgabeparameter `<Result>` im String-Format geliefert.

**BASE64**- Wird dieser Wert übergeben, wird das Ergebnis base64-encoded zurückgeliefert.

### **Rückgabewerte:**

[Result] (BASE64/STRING): liefert die History der angefragten Ohrs.

## **med.ObservationValues**

### **Beschreibung:**

Liefert Werte des Laborsystems. Über diesen Job können Werte aus dem Laborsystem ermittelt werden. Zusätzlich werden die zugehörigen Stammdaten mitgeliefert. Die Units-Stammdaten werden entsprechend des Ergebnisses gefiltert. Die Viewsets werden immer komplett zurückgeliefert.

### **Parameter:**

[Patients] (STRING): ObjektIDs der Patienten, für die die Werte ermittelt werden sollen. Sind die Daten für mehrere Patienten zu ermitteln, müssen die IDs kommasepariert übergeben werden. Wird dieser Parameter leer übergeben, wird ein Fehler zurückgeliefert. Wird dieser Parameter nicht angegeben, wird nicht auf Patienten selektiert.

[Visits] (STRING): ObjektIDs der Aufenthalte, für die die Werte ermittelt werden sollen. Sind die Daten für mehrere Aufenthalte zu ermitteln, müssen die IDs kommasepariert übergeben werden. Wird dieser Parameter leer übergeben, wird ein Fehler zurückgeliefert. Wird dieser Parameter nicht angegeben, wird nicht auf Aufenthalte selektiert.

[State] (STRING): schränkt die Abfrage nach dem Status ein

Der Parameterwert ist kontextinsensitiv, folgende Parameterwerte werden unterstützt:

**ALL** - Es werden sowohl die aktuellen, als auch die geupdateten Werte ausgegeben. Dies entspricht der Angabe `<1,2>`, ist jedoch performanter.

**1** - Es werden nur die aktuellen Werte ausgegeben. Dies entspricht dem Aufruf dieses Jobs ohne den Parameter State.

Über den Wert 2 wird durch die Veränderung der Relationen beim Update von Daten kein gültiges Ergebnis geliefert!

[ShowTestValues] (STRING): gibt an, ob die als Test markierten Werte zurückgegeben werden sollen bzw. die nicht als Test markierten. Wird dieser Parameter nicht angegeben, werden die nicht als Test markierten Daten zurückgeliefert. Für die Berücksichtigung müssen sowohl die Werte des Requests, als auch die Werte der Results die gleiche Einstellung besitzen! Wird dieser Parameter nicht angegeben, werden die Daten zurückgeliefert, die nicht als Test markiert wurden.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

**true** - es werden die als Test markierten Werte zurückgeliefert

**false** - es werden die nicht als Test markierten Werte zurückgeliefert

[Optimize] (STRING): Ermöglicht die Optimierung der Anfrage bzw. der Rückgabe des Ergebnisses.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

**NoEmptyFields**

wird dieser Wert übergeben, werden alle Felder, die einen Leerstring enthalten in der Ausgabe unterdrückt. Dies führt zu einer Verringerung der Größe des Ausgabedokuments und einer



Netzwerkentlastung. Wird dieser Parameter nicht angegeben werden die betreffenden Felder mit einem Leerstring zurückgeliefert.

[Encoding] (STRING): gibt an, in welchem Encoding-Format das Rückgabe-XML geliefert werden soll. Wird dieser Parameter nicht angegeben oder ist der Parameter nicht unten aufgeführt, wird UTF-16 geliefert. Wird der Parameter ResultType mit dem Wert String aufgerufen, wird dieser Parameter nicht berücksichtigt und das Ergebnis mit einer UTF-16-Kennung zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgender Parameterwert wird unterstützt:

UTF-8 - Die Daten werden UTF-8-kodiert geliefert.

ASCII - Die Daten werden ASCII-kodiert geliefert (ISO-8859-1).

[ResultType] (STRING): Wird dieser Parameter mit dem Wert STRING angegeben, wird das Ergebnis als String- und nicht als base64-kodierter Parameter zurückgeliefert. Dieser Parameter sollte nur für Testzwecke verwendet werden. Wird dieser Parameter nicht angegeben, werden die Daten base64-kodiert zurückgeliefert.

Der Parameterwert ist kontextinsensitiv, folgende Parameterwerte werden unterstützt:

STRING - Wird dieser Wert übergeben, wird das Ergebnis im Ausgabeparameter <Result> im String-Format geliefert.

BASE64- Wird dieser Wert übergeben, wird das Ergebnis base64-encoded zurückgeliefert.

#### **Rückgabewerte:**

[Result] (BASE64/STRING): liefert die angeforderten Labor-Werte als XML-Dokument

## [med.SaveMedicalRecord](#)

### **Beschreibung:**

Mit diesem Job können medizinische Dokumente im Archiv gespeichert werden. Diese wird von der Optimal Systems GmbH zur Verfügung gestellt.

Es können nur Dokumente gespeichert werden, wenn der Status Code des Medical Records den Wert 'new', 'active' oder 'completed' hat. Bei einer Status Code 'obsolete' oder 'cancelled' wird dieser Job mit einer Ausnahme quittiert. Dem Aufrufer wird dabei eine Fehlermeldung, die programmatisch empfangen werden kann, übermittelt.

Der Storage Code für zu speichernde oder anzufragende Dokumente muss Konsistenzanforderungen an Medical Record aus der HL/V2.x Spezifikation (Kapitel 9 'Medical Records/Information Management (Document Management)') entsprechen.

Attachments und Dokumente, die kein CDA sind, müssen in einer Dateiliste geliefert werden. Das erste Dokument der Liste ist dabei das Medical Record Dokument und die restlichen Dokumente sind Attachments, die mit diesem Medical Record im System gespeichert werden sollen. Die Verweise der Attachments werden über den Parameter MedicalRecordAttachments aufgelöst.

Die Parameter MedicalRecord.id.root oder MedicalRecord.id.extension sind optional. Werden diese nicht angegeben, generiert das System diese IDs.

### **Parameter:**

Wenn kein Parameter ClinicalDocument übergeben wird

[Patientnumber] (STRING): Patientennummer des Patienten, für den der Medical Record angefragt wird.

[Visitnumber] (STRING): Fallnummer des Aufenthaltes des Patienten, für den der Medical Record angefragt wird.

[MedicalRecordContainer] (STRING): Das Dokument im Archiv, das zum Speichern des Medical Records verwendet werden soll.

[MedicalRecordAttachmentsContainer] (STRING): Das Dokument im Archiv, das zum Speichern von Attachments des Medical Records verwendet werden soll.

[MedicalRecord.statusCode] (STRING): Status-Code des Medical Record. Mögliche Werte sind:

new

active

completed

obsolete

cancelled

[MedicalRecord.completionCode] (STRING): Completion-Code des Medical Record. Mögliche Werte sind:

AU - authenticated

DI - dictated

DO - documented

IP - in progress

IN - incomplete

LA - legally authenticated

PA pre-authenticated

[MedicalRecord.availabilityTime] (STRING): Zeitpunkt der Verfügbarkeit des Dokumentes. Dieser Wert wird als HL7 Zeitstempel geliefert.

[MedicalRecord.storageCode] (STRING): Storage-Code des Medical Record. Mögliche Werte sind:

AC - active

AA - active and archived

AR - archived (not active)

PU - purged

### **Entweder**

[ClinicalDocument] (BASE64): CDA-Dokument, das als Medical Record gespeichert werden soll.

### **Oder**

[MedicalRecord.id.root] (STRING): ROOT-ID des Medical Records.

[MedicalRecord.id.extension] (STRING): Extension-ID des Medical Records.

### **Optional**

Wenn nicht angegeben, wird davon ausgegangen, dass keine Relation zu einem Elternobjekt besteht

[MedicalRecord.relatedDocument.parentDocument.id.root] (STRING): Root-ID des in Relation stehenden Dokument.

[MedicalRecord.relatedDocument.parentDocument.id.extension] (STRING): Extension-ID des in Relation stehenden Dokument.

[MedicalRecord.relatedDocument.typeCode] (STRING): Relation zum Elternobjekt. Mögliche Werte sind:

RPLC - Replace, das in die Beziehung stehende Dokument soll ersetzt werden. Es wird eine neue Hauptvariante angelegt.

XFRM - Transform, das gelieferte Dokument ist eine Transformation des in Beziehung stehenden Dokumentes. Z. B. ein digital signierte Version. Es wird eine neue Untervariante angelegt.

APND – Append, das gelieferte Dokument ist als Anhang/zusätzliches Dokument zu dem in Beziehung stehenden Dokumente geliefert worden. Es wird eine neue Hauptvariante 1.0 erstellt.

### **Rückgabewerte:**

[ObjectID] (STRING): Objekt-ID unter der das Dokument im Archiv gespeichert wurde. Diese ID wird vom System erzeugt und kann nicht vom Aufrufer erzeugt werden.

[MedicalRecord.id.root] (STRING): Root der ID des anzufragenden Dokumentes.

[MedicalRecord.id.extension] (STRING): Extension der ID des anzufragenden Dokumentes.

[MedicalRecord.statusCode] (STRING): Status-Code des Medical Record. Mögliche Werte sind:

new

active

completed

obsolete

cancelled

[MedicalRecord.completionCode] (STRING): Completion-Code des Medical Record. Mögliche Werte sind:

AU authenticated

DI dictated

DO documented

IP in progress

IN incomplete

LA legally authenticated

PA pre-authenticated

[MedicalRecord.availabilityTime] (STRING): Zeitpunkt der Verfügbarkeit des Dokumentes. Dieser Wert wird als HL7 Zeitstempel geliefert.

[MedicalRecord.storageCode] (STRING): Storage-Code des Medical Record. Mögliche Werte sind:

AC active

AA active and archived

AR archived (not active)

PU purged

## med.GetMedicalRecord

### Beschreibung:

Mit diesem Job kann ein medizinisches Dokument, das mittels dem Job med.SaveMedicalRecord im Archiv gespeichert wurde, anfragt werden.

Für Abfragen können zwei verschiedene Modus verwendet werden.

Kombination aus OID(entspricht dem root aus der id) und Instanz-ID(entspricht der extension aus der id).

Die Objekt-ID aus dem Archiv

Nachfolgend werden die weiteren Parameter angeführt, die bei einem Jobaufruf angegeben werden können/müssen.

Es werden nur Dokumente geliefert, wenn der Status Code des Medical Records den Wert 'new', 'active' oder 'completed' hat. Bei eine Status Code 'obsolete' oder 'cancelled' wird dem Aufrufer lediglich der Status Code des Medical Records übermittelt.

Der Storage Code für zu speichernde oder anzufragende Dokumente muss Konsistenzanforderungen an Medical Record aus der HL/ V2.x Spezifikation (Kapitel 9 'Medical Records/Information Managment (Document Managemnt)' ) entsprechen.

Die Dokumente werden in einer Dateiliste geliefert. Das erste Dokument der Liste ist dabei das Medical Record Dokument und die restlichen Dokumente sind Attachments, die mit diesem Medical Record im System gespeichert wurden. Die Verweise der Attachments können über den Ausgabeparameter MedicalRecordAttachments aufgelöst werden.

### Parameter:

[Patientnumber] (STRING): Patientennummer des Patienten, für den der Medical Record angefragt wird.

[Visitnumber] (STRING): Fallnummer des Aufenthaltes des Patienten, für den der Medical Record angefragt wird.

[MedicalRecordContainer] (STRING): Das Dokument im Archiv, das zum Speichern des Medical Records verwendet werden soll.

[MedicalRecordAttachmentsContainer] (STRING): Das Dokument im Archiv, das zum Speichern von Attachments des Medical Records verwendet werden soll.

### Entweder

[MedicalRecord.id.root] (STRING): Root der ID des anzufragenden Dokumentes.

[MedicalRecord.id.extension] (STRING): Extension der ID des anzufragenden Dokumentes.

### oder

[ObjectID] (STRING): Objekt-ID aus dem Archiv des anzufragenden Dokumentes.

### Rückgabewerte:

[DMSContent] (STRING): DMSContent, der durch diese Anfrage generiert wurde.

[MedicalRecord.statusCode] (STRING): Status-Code des Medical Record. Mögliche Werte sind:

new

active

completed

obsolete

cancelled

[MedicalRecord.completionCode] (STRING): Completion-Code des Medical Record. Mögliche Werte sind:

AU authenticated

DI dictated

DO documented

IP in progress

IN incomplete

LA legally authenticated

PA pre-authenticated

[MedicalRecord.availabilityTime] (STRING): Zeitpunkt der Verfügbarkeit des Dokumentes. Dieser Wert wird als HL7 Zeitstempel geliefert.

[MedicalRecord.storageCode] (STRING): Storage-Code des Medical Record. Mögliche Werte sind:

AC active

AA active and archived

AR archived (not active)

PU purged

[MedicalRecordAttachments] (STRING): Liste der Namen/Verweise der Attchments als kommaseparierte Liste.

[FileCount] (INT): Anzahl der gelieferten Dokumente/Dateien einschließlich dem Medical Record.

## med.NotifyMedicalRecord

### Beschreibung:

Mit diesem Job kann der Dokumenten Status und die Transitionen bearbeitet werden. Für weitere Informationen wird auf HL7 V3 Spezifikation verwiesen.

### Parameter:

[Patientnumber] (STRING): Patientennummer des Patienten, für den der Medical Record angefragt wird.

[Visitnumber] (STRING): Fallnummer des Aufenthaltes des Patienten, für den der Medical Record angefragt wird.

[MedicalRecordContainer] (STRING): Das Dokument im Archiv, das zum Speichern des Medical Records verwendet werden soll.

[MedicalRecordAttachmentsContainer] (STRING): Das Dokument im Archiv, das zum Speichern von Attachments des Medical Records verwendet werden soll.

[MedicalRecord.id.root] (STRING): Root der ID des anzufragenden Dokumentes.

[MedicalRecord.id.extension] (STRING): Extension der ID des anzufragenden Dokumentes.

[MedicalRecord.statusCode] (STRING): Status-Code des Medical Record. Mögliche Werte sind:

new

active

completed

obsolete

cancelled

[MedicalRecord.completionCode] (STRING): Completion-Code des Medical Record. Mögliche Werte sind:

AU authenticated

DI dictated

DO documented

IP in progress

IN incomplete

LA legally authenticated

PA pre-authenticated

[MedicalRecord.availabilityTime] (STRING): Zeitpunkt der Verfügbarkeit des Dokumentes. Dieser Wert wird als HL7 Zeitstempel geliefert.

[MedicalRecord.storageCode] (STRING): Storage-Code des Medical Record. Mögliche Werte sind:

AC active

AA active and archived

AR archived (not active)

PU purged

#### **Rückgabewerte:**

[MedicalRecord.id.root] (STRING): Root der ID des anzufragenden Dokumentes.

[MedicalRecord.id.extension] (STRING): Extension der ID des anzufragenden Dokumentes.

[MedicalRecord.statusCode] (STRING): Status-Code des Medical Record. Mögliche Werte sind:

new

active

completed

obsolete

cancelled

[MedicalRecord.completionCode] (STRING): Completion-Code des Medical Record. Mögliche Werte sind:

AU authenticated

DI dictated

DO documented

IP in progress

IN incomplete

LA legally authenticated

PA pre-authenticated

[MedicalRecord.availabilityTime] (STRING): Zeitpunkt der Verfügbarkeit des Dokumentes. Dieser Wert wird als HL7 Zeitstempel geliefert.

[MedicalRecord.storageCode] (STRING): Storage-Code des Medical Record. Mögliche Werte sind:

AC active

AA active and archived

AR archived (not active)

PU purged

## med.GetSystemOID

### Beschreibung:

Mit diesem Job kann die für das System eindeutige OID ermittelt werden.

ASN.1 Notation: {iso(1) member-body(2) de(276) din-certco(0) gesundheitswesen(76) instanzen-identifikatoren(3) organisationen(1) optimal-systems(11)}

dot notation: 1.2.276.0.76.3.1.11

URN-notation: urn:oid:1.2.276.0.76.3.1.11

### Rückgabewerte:

[OID] (STRING): liefert die OID für das System

## MNG-Engine (Namespace mng)

Diese Engine stellt Jobs für die Verwaltung von Gruppen und Benutzern von enaio® bereit.

- § [mng.AddUserGroupAsc](#)
- § [mng.CreateGroup](#)
- § [mng.CreateUser](#)
- § [mng.DeleteGroup](#)
- § [mng.DeleteUser](#)
- § [mng.EmptyGroup](#)
- § [mng.GetGroupAttributes](#)
- § [mng.GetGroupList](#)
- § [mng.GetGroupMembers](#)
- § [mng.GetUserAttributes](#)
- § [mng.GetUserGroups](#)
- § [mng.GetUserProfile](#)
- § [mng.GetUserList](#)
- § [mng.RemoveUserGroupAsc](#)
- § [mng.SetGroupAttributes](#)
- § [mng.SetUserAttributes](#)
- § [mng.StoreUserProfile](#)

Folgende Jobs können nur von Benutzern mit Administratorrechten ausgeführt werden:

- § mng.AddUserGroupAsc
- § mng.CreateGroup
- § mng.CreateUser
- § mng.DeleteGroup
- § mng.DeleteUser
- § mng.EmptyGroup
- § mng.RemoveUserGroupAsc
- § mng.SetGroupAttributes
- § mng.SetUserAttributes

### mng.AddUserGroupAsc

#### **Beschreibung:**

Dieser Job fügt die angegebenen Benutzer einer Gruppe hinzu. Der Benutzer und die Gruppe kann entweder über die GUID oder die ID angegeben werden.

#### **Parameter:**

Flags (INT4): z. Z. nicht genutzt



## AdmInfo (BASE64): Gruppenzuordnung im XML-Format

### Beispiel:

#### Aufbau von AdmInfo

```
<AdmInfo>
<Associations>
<Association osuid="" osgid="" />
<Association osuid="" osgid="" />
<!-- -ORDER- - >
<Association user_id="" group_id="" />
<Association user_id="" group_id="" />
</Associations>
</AdmInfo>
```

### Hinweis:

#### Genauere Beschreibung von AdmInfo

§ [osuid] (STRING): GUID des Benutzers

§ [osgid] (STRING): GUID der Gruppe

§ [user\_id] (STRING): ID des Benutzers

§ [group\_id] (INT): ID der Gruppe

### Siehe auch:

[mng.RemoveUserGroupAsc](#)

## mng.CreateGroup

### Beschreibung:

Dieser Job legt eine neue Benutzer-Gruppe an. Es wird ein Eintrag in die DB-Tabelle 'gruppen' erstellt. Die ID und Osguid werden durch den Job generiert und im XML zurückgegeben.

### Parameter:

Flags (INT4): z. Z. nicht genutzt

GroupInfo (BASE64): Eigenschaften der Gruppe im XML-Format

HasEncoding (boolean): GroupInfo beinhaltet ncoding (z. B. UTF-8)

### Rückgabewerte:

GroupInfo (BASE64): Eigenschaften der Gruppe im XML-Format (id und osguid sind gesetzt)

### Beispiel:

#### Aufbau von GroupInfo

```
<AdmInfo>
<Groups>
<Group name="Test" profil="0" description="" />
</Groups>
</AdmInfo>
```

### Hinweis:

#### Genauere Beschreibung von GroupInfo

§ id (INT): ID der Gruppe

§ name (STRING): Name der Gruppe

- § osguid (STRING): GUID der Gruppe
- § profil (LONG): ID des Profilusers, welcher der Gruppe zugeordnet ist
- § description (STRING): Beschreibung zur Gruppe

## mng.CreateUser

### Beschreibung:

Dieser Job legt einen neuen Benutzer an. Es wird ein neuer Datensatz in der DB-Tabelle 'benutzer' erstellt. Die ID und Osguid werden durch den Job generiert und im XML zurückgegeben.

### Parameter:

Flags (INT4): z. Z. nicht genutzt

UserInfo (BASE64): Eigenschaften des Benutzers im XML-Format

HasEncoding (boolean): UserInfo beinhaltet ncoding (z. B. UTF-8)

### Rückgabewerte:

UserInfo (BASE64): Eigenschaften der Benutzers im XML-Format (id und osguid sind gesetzt)

### Beispiel:

#### Aufbau von UserInfo

```
<AdmInfo>
<Users>
<User account_type="0" bemerkung="" benutzer="TESTUSER" flags="1"
geaendert="1" langid="0" locked="0" logincount="0"
loginstation="" logintime="0" name="Peter Muster"
osemail=""
password="B62441422712357307" profil="-1" server_id="3"
station="" supervisor="0" validfrom="" validto="">
</User>
</Users>
</AdmInfo>
```

### Hinweis:

#### Genauere Beschreibung von XmlInfo

User: enthält Informationen zu einem Benutzer

- § account\_type (INT): Anmeldungstyp
  - § NULL/0 = Benutzeranmeldung
  - § 1 = Anmeldung des Applikationsservers
  - § 2 = Anmeldung von ANONYMOUS
  - § 3 = Anmeldung des Applikationsservers (z. B. Java-Server)
- § bemerkung (STRING): Feld für Bemerkungen (z. B. Telefonnummer)
- § benutzer (STRING): Benutzername
- § flags (INT): 0 = normaler Benutzer, 1 = Server oder ANONYMOUS
- § geaendert (INT): 0 = Profil wurde nicht verändert; 1 = an dem Profil wurden Änderungen vorgenommen
- § id (INT): ID des Benutzers
- § langid (INT): ID der verwendeten Sprache (leer = deutsch)

- § locked (INT): 1 = Benutzer ist gesperrt, ansonsten 0
- § logincount (INT): Anzahl der Loginversuche
- § loginstation (STRING): Name der letzten Einlogstation
- § logintime (INT): Zeitpunkt des Logins (Zeitstempel)
- § name (STRING): vollständiger Name des Benutzers
- § osemail (STRING): E-Mail der Benutzers
- § osguid (STRING): GUID des Benutzers
- § passwort (STRING): kodierte Passwort des Benutzers
- § profil (INT): -1 = Benutzer hat kein Profil; 0 = Benutzer-Profil; >0 = ID des zugeordneten Benutzer-Profiles
- § server\_id (INT): ID des Servers
- § station (STRING): Name der Arbeitsstation des Benutzers
- § supervisor (INT): -1 = Supervisor, ansonsten 0
- § validfrom (INT): Benutzerkonto gültig von (Zeitstempel)
- § validto (INT): Benutzerkonto gültig bis (Zeitstempel)

## mng.DeleteGroup

### Beschreibung:

Dieser Job löscht eine Gruppe aus der DB-Tabelle ‚gruppen‘. Eine Gruppe kann nur gelöscht werden, wenn ihr keine Benutzer mehr zugeordnet sind (DB-Tabelle ‚bgrel‘).

### Parameter:

Flags (INT4): gibt an, über welchen Parameter die Gruppe identifiziert werden soll

- § 0 - Parameter GroupGuid
- § 1 - Parameter GroupId
- § 2 - Parameter GroupName

[GroupGuid] (BASE64): Guid der Gruppe

[GroupId] (BASE64): ID der Gruppe

[GroupName] (BASE64): Name der Gruppe

### Siehe auch:

[mng.EmptyGroup](#)

## mng.DeleteUser

### Beschreibung:

Dieser Job löscht einen Benutzer aus der DB-Tabelle ‚benutzer‘. Es werden auch die Gruppenzuordnungen (bgrel), Systemrollen (ossysroles), Abonnements (osabonnement) und persönlichen Einstellungen (osconf) des Benutzers gelöscht. Außerdem ist es möglich die Mappen und den Inhalt des Postfachs an einen anderen Benutzer weiterzuleiten, bzw. auch diese Daten vollständig zu löschen.

### Parameter:

Flags (INT4): gibt an, über welchen Parameter der Benutzer identifiziert werden soll

§ 0 - Parameter User/ggf. Target

§ 1 - Parameter UserGuid/ggf. TargetGuid

§ 2 - Parameter UserId/ggf. TargetId

InheritanceFlags (INT4): gibt an, ob Mappen und der Inhalt des Postfachs an einen anderen Benutzer weitergeleitet werden soll

§ 0 – Mappen und Mails werden gelöscht

§ 1 – Mappen werden weitergeleitet

§ 2 – Mails werden weitergeleitet

§ 3 - Mappen und Mails werden weitergeleitet

[User] (BASE64): Benutzername

[UserGuid] (BASE64): ID des Benutzers

[UserId] (BASE64): Name des Benutzers

[Target] (BASE64): Benutzername (erhält Mappen/Mails)

[TargetGuid] (BASE64): ID des Benutzers (erhält Mappen/Mails)

[TargetId] (BASE64): Name des Benutzers (erhält Mappen/Mails)

## mng.EmptyGroup

### Beschreibung:

Dieser Job leert eine Gruppe. Es werden alle Benutzerzuordnungen gelöscht (DB-Tabelle 'bgrel').

### Parameter:

Flags (INT4): gibt an, über welchen Parameter die Gruppe identifiziert werden soll

§ 0 - Parameter GroupGuid

§ 1 - Parameter GroupId

§ 2 - Parameter GroupName

[GroupGuid] (BASE64): Guid der Gruppe

[GroupId] (BASE64): ID der Gruppe

[GroupName] (BASE64): Name der Gruppe

## mng.GetGroupAttributes

### Beschreibung:

Dieser Job liefert die Eigenschaften der angegebenen Gruppe.

### Parameter:

Flags (INT4): z. Z. nicht genutzt

Group (STRING): Name der Gruppe

### Rückgabewerte:

XmlInfo (STRING): Eigenschaften der Gruppe im XML-Format

**Beispiel:****Aufbau von XmlInfo**

```
<AdmInfo>
<Groups>
<Group id="0" name="STANDARD" osguid="C9BBC4B0D7754065B3EA6232D7B70003"
profil="0" description="">
</Group>
</Groups>
</AdmInfo>
```

**Hinweis:**

Genauere Beschreibung von XmlInfo

§ id (INT): ID der Gruppe

§ name (STRING): Name der Gruppe

§ osguid (STRING): GUID der Gruppe

§ profil (LONG): ID des Profilusers, welcher der Gruppe zugeordnet ist

§ description (STRING): Beschreibung zur Gruppe

**Siehe auch:**

[mng.GetGroupList](#) , [mng.SetGroupAttributes](#)

**mng.GetGroupList****Beschreibung:**

Dieser Job liefert eine Liste aller Gruppen.

**Parameter:**

Flags (INT4): z. Z. nicht genutzt

**Rückgabewerte:**

GroupList (STRING): Liste aller definierten Gruppen im XML-Format

**Beispiel:****Aufbau von GroupList**

```
<AdmInfo>
<Groups>
<Group id="0" name="STANDARD" osguid="C9BBC4B0D7754065B3EA6232D7B70003"
profil="0" description=""></Group>
<Group id="157" name="TEST" osguid="B36506740D764731836365D04333D3AD"
profil="79" description=""></Group>
<Group id="18" name="ALLE MITARBEITER" osguid="65A56409BB3FFFC687FCC9B90"
profil="0" description=""></Group>
</Groups>
</AdmInfo>
```

**Hinweis:**

Genauere Beschreibung von GroupList

§ id (INT): ID der Gruppe

§ name (STRING): Name der Gruppe

§ osguid (STRING): GUID der Gruppe

§ profil (LONG): ID des Profilusers, welcher der Gruppe zugeordnet ist

§ description (STRING): Beschreibung zur Gruppe

**Siehe auch:**

[mng.GetGroupAttributes](#) , [mng.GetGroupMembers](#)

## mng.GetGroupMembers

**Beschreibung:**

Dieser Job liefert eine Liste aller Mitglieder der angegebenen Gruppe.

**Parameter:**

Flags (INT4): gibt an, über welchen Parameter gesucht werden soll

§ 0 = suchen über Parameter GroupName

§ 1 = suchen über Parameter GroupGUID

§ 2 = suchen über Parameter GroupID

[GroupName] (STRING): Name der Gruppe

[GroupGUID] (STRING): GUID der Gruppe

[GroupID] (INT4): ID der Gruppe

**Rückgabewerte:**

UserList (STRING): Liste aller Gruppenmitglieder

**Beispiel:**

Aufbau von UserList

```
<AdmInfo>
<Users>
<User benutzer="ROOT" id="2" name=" "
osguid="C97ABFC32E09431192E4B13CF47293D6"></User>
<User benutzer="Testuser" id="49" name="Peter Muster"
osguid="6759985B74A44747ACC93F031913006C"></User>
</Users>
</AdmInfo>
```

**Hinweis:**

Genauere Beschreibung von UserList

Users: Liste aller Gruppenmitglieder

§ benutzer (STRING): Benutzername

§ id (INT): ID des Benutzers

§ name (STRING): vollständiger Name des Benutzers

§ osguid (STRING): GUID des Benutzers

## mng.GetUserAttributes

**Beschreibung:**

Dieser Job liefert die Eigenschaften des angegebenen Benutzers.

**Parameter:**

Flags (INT4): z. Z. nicht genutzt

User (STRING): Name des Benutzer aus DB-Feld 'benutzer.benutzer'

### Rückgabewerte:

XmlInfo (STRING): Informationen zum Benutzer im XML-Format

### Beispiel:

Aufbau von XmlInfo

```
<AdmInfo>
<Users>
<User account_type="0" bemerkung="" benutzer="TESTUSER" flags="1"
geaendert="1" id="70" langid="0" locked="0" logincount="0"
loginstation="" logintime="0" name="Peter Muster"
osemail="" osguid="A95EA1EEA16A4EDF916400F6E2F5BCF9"
password="B62441422712357307" profil="-1" server_id="3"
station="" supervisor="0" validfrom="" validto="">
</User>
</Users>
</AdmInfo>
```

### Hinweis:

Genauere Beschreibung von XmlInfo

User: enthält Informationen zu einem Benutzer

§ account\_type (INT): Anmeldungstyp

§ NULL/0 = Benutzeranmeldung

§ 1 = Anmeldung des Applikationsservers

§ 2 = Anmeldung von ANONYMOUS

§ 3 = Anmeldung des Applikationsservers (z. B. Java-Server)

§ bemerkung (STRING): Feld für Bemerkungen (z. B. Telefonnummer)

§ benutzer (STRING): Benutzername

§ flags (INT): 0 = normaler Benutzer, 1 = Server oder ANONYMOUS

§ geaendert (INT): 0 = Profil wurde nicht verändert; 1 = an dem Profil wurden Änderungen vorgenommen

§ id (INT): ID des Benutzers

§ langid (INT): ID der Sprache der Objektdefinition (leer = deutsch)

§ locked (INT): 1 = Benutzer ist gesperrt, ansonsten 0

§ logincount (INT): Anzahl der Loginversuche

§ loginstation (STRING): Name der letzten Einlogstation

§ logintime (INT): Zeitpunkt des Logins (Zeitstempel)

§ name (STRING): vollständiger Name des Benutzers

§ oemail (STRING): E-Mail der Benutzers

§ osguid (STRING): GUID des Benutzers

§ profil (INT): -1 = Benutzer hat kein Profil; 0 = Benutzer-Profil; >0 = ID des zugeordneten Benutzer-Profiles

§ server\_id (INT): ID des Servers

- § station (STRING): Name der Arbeitsstation des Benutzers
- § supervisor (INT): -1 = Supervisor, ansonsten 0
- § validfrom (INT): Benutzerkonto gültig von (Zeitstempel)
- § validto (INT): Benutzerkonto gültig bis (Zeitstempel)

**Siehe auch:**

[mng.SetUserAttributes](#)

## mng.GetUserGroups

### **Beschreibung:**

Dieser Job liefert eine Liste aller Gruppen, in denen sich der angegebene Benutzer befindet.

### **Parameter:**

Flags (INT4): z. Z. nicht genutzt

UserGUID (STRING): GUID des Benutzers

### **Rückgabewerte:**

GroupList (STRING): Liste aller Gruppen, in denen sich der Benutzer befindet

### **Beispiel:**

Aufbau von GroupList

```
<AdmInfo>
<Groups>
<Group id="0" name="STANDARD" osguid="C9BBC4B0D7754065B3EA6232D7B70003"
profil="0" description="" "></Group>
<Group id="157" name="TEST" osguid="B36506740D764731836365D04333D3AD"
profil="79" description="" "></Group>
</Groups>
</AdmInfo>
```

### **Hinweis:**

Genauere Beschreibung von GroupList

Groups: Liste aller Gruppen, in welchen sich der Benutzer befindet

- § id (INT): ID der Gruppe
- § name (STRING): Name der Gruppe
- § osguid (STRING): GUID der Gruppe
- § profil (LONG): ID des Profilusers, welcher der Gruppe zugeordnet ist
- § description (STRING): Beschreibung zur Gruppe

## mng.GetUserList

### **Beschreibung:**

Dieser Job liefert eine Liste aller Benutzer.

### **Parameter:**

Flags (INT4): z. Z. nicht genutzt



**Rückgabewerte:**

UserList (STRING): Liste aller Benutzer

**Beispiel:**

Aufbau von UserList

```
<AdmInfo>
<Users>
<User benutzer="ROOT" id="2" name=" "
osguid="C97ABFC32E09431192E4B13CF47293D6"></User>
<User benutzer="Testuser" id="49" name="Peter Muster"
osguid="6759985B74A44747ACC93F031913006C"></User>
</Users>
</AdmInfo>
```

**Hinweis:**

Genauere Beschreibung von UserList

Users: Liste aller Benutzer

§ benutzer (STRING): Benutzername

§ id (INT): ID des Benutzers

§ name (STRING): vollständiger Name des Benutzers

§ osguid (STRING): GUID des Benutzers

**Siehe auch:**

[mng.GetUserAttributes](#)

[mng.GetUserProfile](#)

**Beschreibung:**

Der Job übergibt dem Client das Profil für einen Anwender.

**Parameter:**

Flags (INT):

§ HIWORD(Flags) = 1: LowDateTime und HighDateTime werden zurückgegeben

§ HIWORD(Flags) = 2: LowDateTime wird zurückgegeben

UserProfile (STRING): UserProfile

**Rückgabewerte:**

FileCount (INT): FileCount gleich 1

[LowDateTime] (INT): Datumsstempel des Userprofils im LowDateTime-Format

[HighDateTime] (INT): Datumsstempel des Userprofils HighDateTime

Dateiliste: Dateiname mit dem vollständigen Pfad

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[mng.StoreUserProfile](#)

## mng.RemoveUserGroupAsc

### Beschreibung:

Dieser Job löscht eine Zuordnung eines Benutzers zu einer Gruppe (DB-Tabelle bgrel).

### Parameter:

Flags (INT4): gibt an, welcher Parameter verwendet werden soll

§ 0 - AdmInfo

§ 1 – UserGuid

[AdmInfo] (BASE64): Gruppenzuordnungen, die gelöscht werden sollen

[UserGUID] (STRING): GUID des Benutzers (Benutzer wird aus allen Gruppen gelöscht, in denen er sich befindet)

### Beispiel:

Aufbau von AdmInfo

```
<AdmInfo>
<Associations>
<Association user_id="" group_id="" />
<!-- ODER -->
<Association osuid="" osgid="" />
</Associations>
</AdmInfo>
```

### Hinweis:

Genauere Beschreibung von AdmInfo

§ [osuid] (STRING): GUID des Benutzers

§ [osgid] (STRING): GUID der Gruppe

§ [user\_id] (STRING): ID des Benutzers

§ [group\_id] (INT): ID der Gruppe

## mng.SetGroupAttributes

### Beschreibung:

Dieser Job setzt die Eigenschaften einer Gruppe.

### Parameter:

Flags (INT4): z. Z. nicht genutzt

GroupInfo (BASE64): Eigenschaften der Gruppe im XML-Format

HasEncoding (boolean): GroupInfo beinhaltet ncoding (z. B. UTF-8)

### Beispiel:

Aufbau von GroupInfo

```
<AdmInfo>
<Groups>
<Group id="0" name="STANDARD" osguid="AE38D1BB1F1C4CB98B5695A2935E0169"
profil="0" description="Test"></Group>
</Groups>
</AdmInfo>
```

**Hinweis:**

Genauere Beschreibung von GroupInfo

- § id (INT): ID der Gruppe
- § name (STRING): Name der Gruppe
- § osguid (STRING): GUID der Gruppe
- § profil (LONG): ID des Profilusers, welcher der Gruppe zugeordnet ist
- § description (STRING): Beschreibung zur Gruppe

**Siehe auch:**

[mng.GetGroupAttributes](#)

## mng.SetUserAttributes

**Beschreibung:**

Dieser Job setzt die Eigenschaften eines Benutzers.

**Parameter:**

Flags (INT4):

UserInfo (BASE64): Eigenschaften im XML-Format

HasEncoding (boolean): UserInfo beinhaltet ncoding (z. B. UTF-8)

**Beispiel:**

Aufbau von UserInfo

```
<AdmInfo>
<Users>
<User account_type="0" bemerkung="" benutzer="Test" flags="1"
geaendert="1" id="67" langid="0" locked="0" logincount="0"
loginstation="" logintime="0" name="Peter Muster" osemail=""
osguid="EF989801BA8847199335DD4FDEF30BC5"
password="BF754341546553351243620206006521266514574240603407"
profil="66" server_id="3" station="" supervisor="0" validfrom=""
validto="" />
</Users>
</AdmInfo>
```

**Hinweis:**

Genauere Beschreibung von XmlInfo

User: enthält Informationen zu einem Benutzer

- § account\_type (INT): Anmeldungstyp
  - § NULL/0 = Benutzeranmeldung
  - § 1 = Anmeldung des Applikationsservers
  - § 2 = Anmeldung von ANONYMOUS
  - § 3 = Anmeldung des Applikationsservers (z. B. Java-Server)
- § bemerkung (STRING): Feld für Bemerkungen (z. B. Telefonnummer)
- § benutzer (STRING): Benutzername
- § flags (INT): 0 = normaler Benutzer, 1 = Server oder ANONYMOUS

- § **geaendert** (INT): 0 = Profil wurde nicht verändert; 1 = an dem Profil wurden Änderungen vorgenommen
- § **id** (INT): ID des Benutzers
- § **langid** (INT): ID der verwendeten Sprache (leer = deutsch)
- § **locked** (INT): 1 = Benutzer ist gesperrt, ansonsten 0
- § **logincount** (INT): Anzahl der Loginversuche
- § **loginstation** (STRING): Name der letzten Einlogstation
- § **logintime** (INT): Zeitpunkt des Logins (Zeitstempel)
- § **name** (STRING): vollständiger Name des Benutzers
- § **osemail** (STRING): E-Mail der Benutzers
- § **osguid** (STRING): GUID des Benutzers
- § **passwort** (STRING): kodierte Passwort des Benutzers
- § **profil** (INT): -1 = Benutzer hat kein Profil; 0 = Benutzer-Profil; >0 = ID des zugeordneten Benutzer-Profils
- § **server\_id** (INT): ID des Servers
- § **station** (STRING): Name der Arbeitsstation des Benutzers
- § **supervisor** (INT): -1 = Supervisor, ansonsten 0
- § **validfrom** (INT): Benutzerkonto gültig von (Zeitstempel)
- § **validto** (INT): Benutzerkonto gültig bis (Zeitstempel)

**Siehe auch:**

[mng.GetUserList](#) , [mng.GetUserAttributes](#)

## mng.StoreUserProfile

### **Beschreibung:**

Der Job speichert das vom Client erhaltene Anwender-Profil und schreibt eine Historiendatei (gleicher Name mit Endung bac). Die übergebene Profildatei wird am Client gelöscht.

### **Parameter:**

**Flags** (INT):

- § **HIWORD(Flags)** = 2: Datumsstempel im LowDateTime-Format speichern; ansonsten LowDateTime-Format und HighDateTime-Format verwenden

**UserProfile** (STRING): Name unter welchem die Datei gespeichert werden soll

**LowDateTime** (INT): Datumsstempel im LowDateTime-Format

**HighDateTime** (INT): Datumsstempel im HighDateTime -Format

**Dateiliste:** Name und Pfad der Profildatei

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[mng.GetUserProfile](#)

## OCR-Engine (Namespace ocr)

Diese Engine stellt Jobs zur Texterkennung bereit.

§ [ocr.DoOCR](#)

§ [ocr.DoDocOCR](#)

### ocr.DoOCR

**Beschreibung:**

Dieser Job führt für die übergebene Datei eine Texterkennung durch und liefert den erkannten Text als Textdatei zurück.

**Parameter:**

Flags (INT4): z. Z. nicht genutzt

Type (INT4): 1 = mehrspaltige Texte (z. B. Zeitungsausschnitte) sollen erkannt werden, sonst 0

Dateiliste: Name und Pfad der Datei, für die Texterkennung durchgeführt werden soll

**Rückgabewerte:**

Dateiliste: Name und Pfad der Text-Datei, die erkannten Text enthält

### ocr.DoDocOCR

**Beschreibung:**

Dieser Job führt für das angegebene enaio®-Dokument eine Texterkennung durch und liefert den erkannten Text als Textdatei zurück.

**Parameter:**

Flags (INT4): z. Z. nicht genutzt

Type (INT4): 1 = mehrspaltige Texte (z. B. Zeitungsausschnitte) sollen erkannt werden, sonst 0

DocID (INT4): ID des Dokuments

**Rückgabewerte:**

Dateiliste: Name und Pfad der Text-Datei, die erkannten Text enthält

## Standard-Engine (Namespace std)

In der Standard-Engine werden Funktionen zum dateiorientierten Dokumentenmanagement implementiert. Das sind insbesondere Funktionen zum Speichern und Laden von Dokumenten, zur Archivierung und zur Realisierung des Dokumentenaustausches zwischen mehreren Servern.

Der Standard-Engine unterliegt die Verwaltung des Work-, Cache- und Archivbereichs des Applikationsservers.

§ [Work-, Cache- und Archiv-Verwaltung](#)

§ [Datei-Verwaltung](#)

§ [Interne Jobs](#)

§ [Sonstige Jobs](#)

### Work-, Cache- und Archiv-Verwaltung

§ [std.CleanUpCache](#)

§ [std.ClearFromCache](#)

§ [std.DoArchive](#)

§ [std.DoPrefetch](#)

§ [std.MoveToCache](#)

§ [std.StoreInCache](#)

§ [std.StoreInCacheByID](#)

§ [std.StoreInCacheDirect](#)

§ [std.StoreInWork](#)

§ [std.UndoArchive](#)

### std.CleanUpCache

#### **Beschreibung:**

Dieser Job bereinigt den Inhalt des CACHE.

#### **Parameter:**

[Flags] (INT): Priorität (Standardwert 2)

§ 0 = es werden solange Dateien gelöscht, bis die minimale Cachegröße erreicht ist

§ 1 = es werden Dateien gelöscht, die länger als die maximalen Tage im CACHE liegen

§ 2 = Kombination aus Priorität 0 und 1: Es wird versucht, solange Dateien zu löschen bis die minimale Cachegröße erreicht ist, dabei werden aber nur Dateien gelöscht, die länger als die maximalen Tage im CACHE liegen. Dabei kann es sein, dass die maximale Cachegröße nicht unterschritten wird

§ 8 = es werden solange Dateien gelöscht, bis die minimale Dokumentenanzahl erreicht ist (siehe Parameter Low)

High (INT): maximale Cachegröße in MB, ist dieser Wert überschritten wird der CACHE bereinigt

Low (INT): minimale Cachegröße in MB; wenn Flags = 8 wird hier die minimale Dokumentenanzahl in tausend Stück angegeben

Days (INT): Anzahl der Tage, die Dokumente max. im Cache gespeichert bleiben sollen

[ExtendDiagnostic] (INT): Optionen für Protokollierung

§ 0 = in diesem Fall werden nur Dateien in Report geschrieben, die nicht gelöscht werden konnten

§ 1 = alle gelöschten Dateien werden protokolliert (Default=0)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.ClearFromCache

**Beschreibung:**

Dieser Job löscht das angegebene Objekt aus dem CACHE.

**Parameter:**

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.DoArchive

**Beschreibung:**

Dieser Job archiviert alle auf archivierbar gesetzten Objekte eines angegeben Typs. Dieser Job kann nur funktionieren, wenn vorher die Archivierung richtig eingerichtet ist (Medien, Sets,...).

**Parameter:**

dwObjectType (INT): Typ der zu archivierenden Objekte

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.DoPrefetch

**Beschreibung:**

Dieser Job lädt das angegebene Objekt aus einem Speichermedium (z. B. Jukebox) in den Cache um Zugriffszeiten zu verringern.

**Parameter:**

Flags (INT): Optionen für die Übertragung

§ 0 = Dia-Datei und Objekt werden übertragen

§ 1 = nur die Objektdateien werden übertragen

§ 2 = nur die Dia-Datei des Objekts wird übertragen

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

DocState (INT): enthält den Zustand des Dokuments im LOWORD und das Read-Write-Flag im HIWORD (Job verarbeitet nur archivierte Dokumente)

FileCount (INT): Anzahl der Dateien, die hier aber keine Rolle spielen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.GetDocumentSlide

**Beschreibung:**

Dieser Job erstellt Dia zu einem Dokument, und liefert es zurück.

**Parameter:**

ObjectID (INT): ID des Objekts

Flags (INT):

GETDOCSLIDE\_FLAG\_NEEDEDONLY(0): angeforderte Groesse wird zurückgegeben, und falls nicht vorhanden, dann Fehler

GETDOCSLIDE\_FLAG\_DEFAULTALLOWED(1): angeforderte Groesse wird zurückgegeben, und falls nicht vorhanden, dann 96x96; dabei auch Widht/Height, um zu wissen, was geliefert ist.

GETDOCSLIDE\_FLAG\_BOTH(2): angeforderte Groesse Groesse wird zurückgegeben, falls vorhanden, und 96x96

GETDOCSLIDE\_FLAG\_ALL(3): alle Dias

Bei GETDOCSLIDE\_FLAG\_ALL und GETDOCSLIDE\_FLAG\_BOTH werden mehrere Dateien geliefert.

Sie haben Endungen DIA001, DIA002 usw.

Dabei gibt es auch Ausgabeparameter DIA001, DIA002 usw., die die Groesse jeweilige Dia festlegen:

DIA001 = 10x10

DIA002 = 96x96 usw.

Bei GETDOCSLIDE\_FLAG\_NEEDEDONLY und GETDOCSLIDE\_FLAG\_DEFAULTALLOWED kann Windth und/oder Height 0 sein,

dann werden Defaultwerte genommen:

if (dwWidth == 0) dwWidth = DEFAULT\_DIA\_WIDTH;

if (dwHeight == 0) dwHeight = DEFAULT\_DIA\_HEIGHT;

Height (INT): Höhe des Dia; falls 0, dann wird 96 genommen

Width (INT): Breite des Dia; falls 0, dann wird 96 genommen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.MoveToCache

**Beschreibung:**

Dieser Job überträgt das angegebene Objekt von einer Server-Gruppe auf eine andere. Der Client kann dieses Objekt nur zum Lesen öffnen.

**Parameter:**



dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.StoreInCache

**Beschreibung:**

Dieser Job sendet die angegebenen Dokument-Dateien an einen enaio® client. Wenn der enaio® client das Dokument noch im Cache hat, kann der Digest-Wert berechnet und dem Job übergeben werden. Der Server berechnet ebenfalls den Digest-Wert des angegebenen Dokuments. Sind beide Digest-Werte identisch, sendet der Server das Dokument nicht an den enaio® client.

**Parameter:**

Flags (INT): Optionen für die Übertragung

§ 0 = Dia-Datei und Objekt werden übertragen

§ 1 = nur die Objektseiten werden übertragen

§ 2 = nur die Dia-Datei des Objekts wird übertragen

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des ausgewählten Dokuments

DocState (INT): gibt an, ob das Dokument zu Lesen oder Schreiben geöffnet werden soll

§ HIWORD = 0 -> Dokument wird zum Schreiben geöffnet

§ HIWORD = 1 -> Dokument wird zum Lesen geöffnet

FileCount (INT): Anzahl der Dateien, die hier aber keine Rolle spielen

[Digest] (STRING): von Client-Anwendung berechneter Digest-Wert für das angeforderte Dokument.

[IncludeDeleted] (BOOLEAN): bei true werden auch Dokumente berücksichtigt, die im Papierkorb liegen.

[IgnoreHashCheck] (BOOLEAN): bei true wird die Prüfung von Hashwert/Signature abgeschaltet, obwohl sie möglicherweise in der Registry eingeschaltet sind.

[AddAnnotations] (BOOLEAN): falls vorhanden und true, dann werden in die Image-Dateien auch Annotations eingebrannt..

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

FileCount (INT): Anzahl der Dateien, die in den Cache übertragen wurden

## std.StoreInCacheByID

**Beschreibung:**

Dieser Job sendet die angegebenen Dokument-Dateien an einen enaio® client.

**Parameter:**

Flags (INT): Optionen für die Übertragung

§ 0 = Dia-Datei zum Objekt wird mit übertragen

§ 1 = nur die Objektdaten an sich werden übertragen

§ 2 = nur die Dia-Datei des Objekts wird übertragen

dwObjectID (INT): ID des Objekts

[Convert] (INT): 0 = keine Konvertierung

1 = Dokumente mit Haupttyp 1, 2, 3 oder 4 werden nach PDF konvertiert,

8 = TIFF Dokumente mit Haupttyp 2 oder 3 werden zu einem Multipage TIFF zusammengefasst

[WhenCOLDThenTIFF] (INT): 1 = ASCII Cold Dateien werden im TIFF Format zurückgeliefert (wird nur beachtet, wenn Convert= 0 gesetzt ist)

[AddAnnotations] (BOOLEAN): falls vorhanden und true, dann werden in die Image-Dateien auch Annotations eingebrannt.

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

FileCount (INT): Anzahl der Dateien, die in den Cache übertragen wurden

Dateiliste: Pfad und Name der Dateien, die in den Cache übertragen wurden

## **std.StoreInCacheDirect**

### **Beschreibung:**

Dieser Job liefert ein angegebenes Dokument über seine ID. Der Job kann benutzt werden, wenn der enaio® server und enaio® client auf demselben Rechner laufen.

### **Parameter:**

Flags (INT): Optionen für die Übertragung

§ 0 = Dia-Datei zum Objekt wird mit übertragen

§ 1 = nur die Objektdaten an sich werden übertragen

§ 2 = nur die Dia-Datei des Objekts wird übertragen

dwObjectID (INT): ID des Objekts

Path (STRING): Pfad, in den die die Objektdaten geschrieben werden

[AddAnnotations] (BOOLEAN): falls vorhanden und true, dann werden in die Image-Dateien auch Annotations eingebrannt.

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

FileCount (INT): Anzahl der Dateien, die in den Cache übertragen wurden.

File\_N (STRING): Name der N-ten Quelldatei

## std.StoreInWork

### Beschreibung:

Dieser Job kopiert alle angegebenen Dokumente in das angegebene Work-Verzeichnis (dieses ergibt sich aus Objekttyp und Objekt-ID). Existieren dort bereits Dateien mit der angegebenen ObjectID, so müssen diese Dateien zunächst gelöscht werden.

### Parameter:

Flags (INT):

§ 0 = Flags: die Dateien werden gelöscht

§ 1 & Flags: die Dateien werden nicht gelöscht (bei 'StoreInWorkDirect' benutzen)

§ 2 & Flags: es werden keine HardLink für Dateien angelegt (bei Variantenverwaltung benutzen)

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

FileCount (INT): Anzahl der Dateien

Dateiliste: Name und Pfad der Dateien, die in das Work-Verzeichnis geschrieben werden sollen.

bAddFiles (INT): bei 1 werden die neue Dateien nicht die alte ersetzen, sondern sie werden hinzugefügt.

bAddFront (INT): bei 1 werden die neue Dateien vorne hinzugefügt.

DocFlagsNeeded (INT): falls vorhanden, dann wird die Spalte 'Flags' in der Objekt-Tabelle auf diesen Wert gesetzt..

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.UndoArchive

### Beschreibung:

Dieser Job stellt das angegebene Dokument aus der Jukebox im Workverzeichnis wieder her.

### Parameter:

Flags (INT): Optionen für Dokumentstatus

§ 1 = Dokument die ins Workverzeichnis geschrieben werden, erhalten den Status archivierbar

§ 0 = Dokument die ins Workverzeichnis geschrieben werden erhalten den Status nicht archivierbar

dwObjectType (INT): Typ des zu dearchivierenden Objekts

dwObjectID (INT): ID des Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Datei-Verwaltung

§ [std.DeleteDocumentVersion](#)

§ [std.DeleteObject](#)

§ [std.DeleteDocument](#)

- § [std.DeleteRemark](#)
- § [std.FindDokumentDigest](#)
- § [std.GetDocStatistics](#)
- § [std.GetDocStream](#)
- § [std.GetDocumentDigest](#)
- § [std.GetDocumentPage](#)
- § [std.GetDocumentStream](#)
- § [std.GetDocVariant](#)
- § [std.SetActiveVariant](#)
- § [std.GetDocVersion](#)
- § [std.GetObjectInfo](#)
- § [std.GetRemark](#)
- § [std.GetSignedDocument](#)
- § [std.MergeDocuments](#)
- § [std.MergeFolder](#)
- § [std.RestoreDocVersion](#)
- § [std.RestoreObject](#)
- § [std.SetHistory](#)
- § [std.StoreRemark](#)
- § [std.StoreSignedDocument](#)
- § [std.Unknown2Known](#)
- § [std.SetPlannedRetention](#)

## std.FindDocumentDigest

### **Beschreibung:**

Dieser Job sucht nach einem Dokument, das den gleichen Hashwert hat. Gesucht wird in der Tabelle osdochash, dabei werden nur Einträge mit osguid = NULL oder osguid = " berücksichtigt, d.h. Versionen werden nicht berücksichtigt.

### **Parameter:**

Flags (INT): momentan nicht verwenden, sollte 0 sein.

Digest (STRING): der Hashwert, nach dem gesucht wird.

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

Result: DokumentID's und Typen von gefundenen Dokumenten, in folgender Form:

11=131072;22=131072; usw.

## std.CalcDocumentDigest

### Beschreibung:

Dieser Job hinterlegt den Hashwert und optional die Signatur für ein gegebenes Dokument. Die Sicherstellung des Hashwertes und der Signatur unterliegt dem Server, für die Verwendung dieses Jobs kontaktieren Sie bitte den Support der Optimal Systems GmbH.

### Parameter:

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des ausgewählten Objekts

Flags (INT): momentan nicht verwendet, sollte 0 sein.

Pwd (STRING): falls Passwort nicht stimmt, wird nichts gemacht.

Sign (BOOLEAN): falls true, dann wird das Dokument auch signiert.

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Digest (STRING): Hashwert von Dateien.

## std.DeleteDocumentVersion

### Beschreibung:

Dieser Job löscht eine bestimmte Version eines Dokuments und ggf. die dazugehörige digitale Signatur.

### Parameter:

Flags (INT): Optionen fürs Löschen

§ 0 = löscht nur Dokumentversion

§ 1 = löscht digitale Signatur

OSOBJID (INT): ID des Dokuments

OSOBJTYP (INT): Typ des Dokuments

OSID (STRING): Versionsid des Dokuments

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.DeleteObject

### Beschreibung:

Dieser Job identifiziert ein Objekt (Dokument, Register, Ordner) über die angegebenen Parameter und löscht es. Die Parameter 'dwParentID' und 'dwParentType' werden nur für Dokumente beachtet. Wenn ein Dokument einige Einträge in der Tabelle 'sdrel' hat und dwParentID = 0 ist, dann werden alle Einträge in dieser Tabelle gelöscht. Wenn dieser Parameter nicht 0 ist, dann wird nur 1 Eintrag gelöscht.

### Parameter:

sDeleteMethod (STRING): Methode zum Löschen

§ 'Delete' -> Objekt wird gelöscht (ohne Papierkorb) oder Objekte die schon im Papierkorb liegen werden endgültig gelöscht

§ 'DeleteWithDocs' -> gültig für Schrank/Register; es werden alle im Objekt enthaltenen Register/Dokumente gelöscht

§ 'Recycle' -> Objekt wird in den Papierkorb verschoben

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des zu löschenden Objekts

dwParentID (INT): ID des Parent des ausgewählten Objekts

dwParentType (INT): Typ des Parent des zu löschenden Objekts

[sSpecific] (STRING): Children = Unterobjekte werden gelöscht, ansonsten leer

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

[sInfo]: wird nur zurückgegeben, wenn ein zu löschendes Register/Schrank Unterobjekte enthält

[Clause]: immer 1

## **std.DeleteDocument**

### **Beschreibung:**

Dieser Job löscht zu einem Dokument die Dokumentdateien. Das Dokument verbleibt als Dokument ohne Seiten.

### **Parameter:**

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des zu löschenden Objekts

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## **std.DeleteRemark**

### **Beschreibung:**

Dieser Job löscht eine zu einem Dokument hinterlegte Notiz.

### **Parameter:**

RemIdent (INT): ID der Notiz

dwObjectType (INT): Typ des Objekts

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## **std.FindDocumentDigest**

### **Beschreibung:**

Dieser Job sucht nach einem Dokument, das den gleichen Hashwert hat. Gesucht wird in der Tabelle osdochash, dabei werden nur Einträge mit osguid = NULL oder osguid = " berücksichtigt, d.h. Versionen werden nicht berücksichtigt.

**Parameter:**

Flags (INT): momentan nicht verwenden, sollte 0 sein.

Digest (STRING): der Hashwert, nach dem gesucht wird.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result: DokumentID's und Typen von gefundenen Dokumenten, in folgender Form:

11=131072;22=131072; usw.

## std.GetDocStatistics

**Beschreibung:**

Dieser Job liefert zum angegebenen Objekt die Dokumentinformationen.

**Parameter:**

dwObjectType (INT): Typ des Objekts

dwObjectID (INT): ID des Objekts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Statistics (STRING): ein durch '@' getrennte Dokumentinformationen

§ Timestamp des Dokuments

§ Nr. der Servergruppe, wo das Dokument liegt

§ Name des Mediums, auf dem das Dokument liegt ('WORK' ist auch ein gültiger Mediennamen)

§ Anzahl der Dokumente des Objekts

§ Dateigröße in Byte des Dokuments (nur wenn Anzahl = 1, DIA wird nicht mit berechnet)

§ Dateiname (inkl. Pfad)

§ Nr. des Modultyps

§ Notizinformation, wenn Notiz an Objekt so ist der Inhalt hier 'NOTE'

§ aktueller Dokumentenzustand (DocState)

§ Objekttyp

## std.GetDocStream

**Beschreibung:**

Dieser Job wurde durch [GetDocumentStream](#) ersetzt.

## std.GetDocumentDigest

### Beschreibung:

Dieser Job liefert den Digest-Wert der Dokumentdateien.

### Parameter:

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Dokuments

[dwFlags] (INT): 0 = Digest-Wert wird aus DB gelesen; 1 = Digest-Wert wird berechnet.

[CheckSignature] (BOOLEAN): bei 1 wird auch die Signature geprüft. In diesem Fall wird Ausgabeparameter Signature zurückgegeben.

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

[LocalDigest]: Digest-Wert der berechnet wurde (nur bei Flags = 1)

TableDigest: Digest-Wert aus DB (Flags = 0,1)

[Signature]:

0 – Signature vorhanden und verifiziert.

1 – Signature nicht vorhanden

2 – Signature konnte wegen einen technischen Fehler nicht verifiziert werden

3 – Signature vorhanden, und ist falsch

## std.GetDocumentPage

### Beschreibung:

Dieser Job liefert eine angegebene Seite eines angegebenen Dokuments.

### Parameter:

dwObjectID (INT): ID des Objekts

Page (INT): Seitennummer des Dokuments

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

nModul (INT): Haupttyp des Dokuments

Dateiliste: Pfad und Name der Datei, die übertragen wurde

## std.GetDocumentStream

### Beschreibung:

Dieser Job liest aus einer Datei ein Dateibereich, erstellt daraus eine neue Datei und liefert diese zurück.

### Parameter:



dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

PageNum (INT): Seitennummer, aus welcher gelesen werden sollen

dwOffset (INT): Offset (ab diesem Byte wird ausgelesen)

dwLength (INT): Anzahl der Bytes, die gelesen werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Dateiliste: Pfad und Name der erstellten Datei

Count (INT): Anzahl der erstellten Dateien

**Hinweis:**

Fehler die in diesem Job auftreten können.

§ ERR\_WRONGPARAM: Der Offset ist größer als die Datei.

§ ERR\_GETOBJECTDEFPARAM: Der Objekttyp ist unbekannt.

§ ERR\_GETPARAMETER: Zu wenige (gültige) Parameter übergeben.

§ ERR\_SQLSELECT: Bei DB-Anfragen sind Fehler aufgetreten.

§ ERR\_NOTDEFINED\_YET: Das Object befindet sich auf einem entfernten Server.

§ ERR\_INSOURCE: Die Datei kann nicht gefunden oder geöffnet werden.

§ ERR\_INDESTINATION: Die Zieldatei kann nicht angelegt werden.

## std.GetDocVariant

**Beschreibung:**

Dieser Job erstellt eine neue Variante des ausgewählten Dokuments.

**Parameter:**

sDocVer (STRING): Nummer der neuen Variante

bTransferPlannedRetention (BOOLEAN) besagt ob die geplante Retentionzeit übernommen werden soll oder nicht.

dwParentID (INT): ID des Objekts

dwObjectType (INT): Typ des ausgewählten Objekts

dwMainType (INT): Haupttyp der neuen Variante

bAddFiles (INT): bei 1 werden die neue Dateien (falls mitgeschickt) nicht die alte ersetzen, sondern sie werden hinzugefügt.

bAddFront (INT): bei 1 werden die neuen Dateien vorne hinzugefügt.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

dwVariantID: ID der neu erstellten Variante

## std.SetActiveVariant

### Beschreibung:

Dieser Job aktiviert eine Variante.

### Parameter:

dwPrevActVarID (INT): ID der aktuellen Variante

dwNextActVarID (INT): ID der Variante, die aktiv gemacht werden soll

dwObjectType (INT): Typ des ausgewählten Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

## std.GetDocVersion

### Beschreibung:

Dieser Job ermittelt die Version des ausgewählten Dokuments.

### Parameter:

GUID (STRING): ID der Version

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des ausgewählten Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Dateiliste: Pfad und Name der Datei, die zurückgeliefert wurde

## std.GetObjectInfo

### Beschreibung:

Dieser Job ermittelt die gewünschten Informationen (Status oder Grösse) des ausgewählten Objekts.

### Parameter:

dwInfoFlag (INT): 0 = Status des Objekts wird ermittelt, 1 = Größe des Objekts wird ermittelt

dwObjectType (INT): Typ des Objekte

dwObjectID (INT): ID des Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

ServerInfo (STRING): enthält die angefragte Information

§ dwInfoFlag = 0: Status wird ermittelt

- § 0 = Dokument hat keine Seiten
- § 1 = Dokument ist archiviert
- § 2 = Dokument ist nicht archiviert/archivierbar
- § 3 = Zustand des Dokuments ist nicht bestimmbar
- § dwInfoFlag = 1: Grösse der Dokumente wird ermittelt (Summe aller Objekt-Dokumente)
  - § 0 = Dokument hat keine Seiten
  - § [N] = ist die Grösse der Dokumentdateien (inkl. DIA) in Byte

## std.GetRemark

### Beschreibung:

Dieser Job liefert eine Notiz, die durch den Objekttyp und die RemIdent (Notiz-ID) identifiziert wird.

### Parameter:

Flags (INT): muss immer 0 sein

RemIdent (INT): ID der Notiz

dwObjectType (INT): Typ des Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

FileCount (INT): ist immer 1, da nur eine Datei ausgegeben wird

Dateiliste: Pfad und Dateiname der gesuchten Notiz

## std.GetSignedDocument

### Beschreibung:

Dieser Job liefert ein digital signiertes Dokument vom Server und bringt es zum Client.

### Parameter:

OSOBJID (INT): ID des ausgewählten Objekts

OSID (STRING): ID der Version

OSOBJTYP (INT): Typ des ausgewählten Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

nFileCount (INT): Anzahl der Dateien

dwUserID (INT): ID des Benutzers

dwTimeStamp (INT): Zeitpunkt der Erstellung

sTrust (STRING):

sFirstName (STRING): Vorname des Benutzers

sStation (STRING): Computernamen

sName (STRING): Name des Benutzers

sClass (STRING):

## std.MergeDocuments

### Beschreibung:

Dieser Job fügt die Dokumentseiten eines angegebenen Objekts einem weiteren Ziel-Objekt hinzu. Der Dokument-Typ beider Objekte sollte dazu gleich sein. Das Ausgangs-Objekt 'verliert' nach dieser Aktion seine Dokumentseiten.

### Parameter:

dwObjectID1 (INT): ID des Ziel-Objekts

dwObjectType1 (INT): Typ des Ziel-Objekts

PageCount1 (INT): Seitenanzahl des Ziel-Objekts

dwObjectID2 (INT): ID des anzuhängenden Objekts

dwObjectType2 (INT): Typ des anzuhängenden Objekts

PageCount2 (INT): Seitenanzahl des anzuhängenden Objekts

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.MergeFolder

### Beschreibung:

Dieser Job fügt zwei Ordner innerhalb der Datenbank zusammen, indem alle Dokumente und Register aus dem Quellordner in den Zielordner geschoben werden.

### Parameter:

Flags (INT): 1 = wird der Quellordner wird gelöscht, ansonsten 0

dwObjectType (INT): Typ der Objekte

FolderDest (INT): ID des Zielordners

FolderSource (DWORD): ID des Quellordners

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.RestoreDocVersion

### Beschreibung:

Dieser Job stellt eine Dokumentenversion als aktuelles Dokument wieder her.

### Parameter:

GUID (STRING): ID der Version, die wiederhergestellt werden soll

OSSTATION (STRING): Computernamen des aufrufenden Clients

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

dwUserId (INT): Benutzer-ID

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.RestoreObject

**Beschreibung:**

Diese Job stellt ein Objekt aus dem Papierkorb wieder her.

**Parameter:**

sRestoreMethod (STRING): String, der die Methode des Wiederherstellens beschreibt

§ SingleRestore = nur das angegebene Objekt wiederherstellen

§ RestoreAllEntity = Unterobjekte des angegebenen Objekts wiederherstellen

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

dwParentID (INT): ID des Objekts, wohin das angegebene Objekt wiederhergestellt werden soll

dwParentType (INT): Typ des Objekts, wohin das angegebene Objekt wiederhergestellt werden soll

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.SetHistory

**Beschreibung:**

Dieser Job fügt einen Eintrag in DB-Tab 'osobjhist' für das angegebene Objekt ein.

**Parameter:**

sInfo (STRING): Informationen über Aktion, die durchgeführt wurde

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.StoreRemark

**Beschreibung:**

Dieser Job speichert eine Notiz in ein, durch den Objekttyp identifiziertes, Verzeichnis innerhalb des NOTE-Verzeichnis.

**Parameter:**

RemIdent (INT): ID der Notiz

dwObjectType (INT): Typ des Objekts, daraus wird das Verzeichnis innerhalb des NOTE-Verzeichnisse ermittelt und danach erstellt

Dateiliste: Pfad und Name der Notiz-Datei (TXT), die gespeichert werden soll

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.StoreSignedDocument

### Beschreibung:

Dieser Job speichert ein signiertes Dokument am Server.

### Parameter:

OSFIRSTNAME (STRING): Vorname des Benutzers

OSNAME (STRING): Name des Benutzers

OSOBJCLASS (STRING): Objektklasse

OSSTATION (STRING): Arbeitsstation

OSTRUST (STRING):

OSOBJID (INT): ID des ausgewählten Objekts

OSTIMESTAMP (INT): Zeitstempel

OSOBJTYP (INT): Typ des ausgewählten Objekts

OSUSERID (INT): ID des Benutzers

OSSIGNATUR-Datei: Datei mit Inhalt Signatur - wird gelöscht

OSSIGHEADER-Datei: Datei mit Inhalt Signatur-Header - wird gelöscht

OSSIGTEXT-Datei: Datei mit Inhalt Signaturtext - wird gelöscht

Datei zur Speicherung: Datei die gespeichert werden soll

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.Unknown2Known

### Beschreibung:

Dieser Job konvertiert ein Dokument ohne Typ in ein Dokument mit dem übergebenen Typ.

### Parameter:

Flags (INT):

§ LOWORD(Flags) = 1 dann:

§ HIWORD(Flags) = 0 --> ursprüngliche Modulnummer wird aus Parameter HIWORD(dwObjectType) ermittelt

§ HIWORD(Flags) != 0 --> ursprüngliche Modulnummer wird aus Parameter HIWORD(Flags) ermittelt

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Objekttyp, in das das angegebene Dokument überführt werden soll

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.SetPlannedRetention

### Beschreibung:

Dieser Job setzt die geplante Retentionzeit für ein Dokument.

### Parameter:

Flags (INT): z. Z. nicht unterstützt, soll 0 sein

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Objekttyp, in das das angegebene Dokument überführt werden soll

sRetentionDate (STRING): Retentionzeit in Form DD.MM.YYYY

### Ausgabeparameter:

sRetentionDate (STRING): gesetzte Retentionzeit in Form DD.MM.YYYY

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.AdjustRetentions

### Beschreibung:

Dieser Job passt das Retention-Datum eines archivierten Dokuments an das geplante Retention-Datum an.

### Parameter:

Flags (INT): z. Z. nicht unterstützt, soll 0 sein

dwObjectID (INT): ID des Objekts (optional)

dwObjectType (INT): Objekttyp. Falls dwObjectID fehlt, werden Dokumente dieses Types angepasst

### Ausgabeparameter:

sRetentionDate (STRING): gesetzte Retentionzeit in Form DD.MM.YYYY; dieser Parameter wird nur gesetzt, wenn dwObjectID definiert wurde.

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Interne Jobs

Die Internen Jobs werden im Allgemeinen nur von der Standard-DMS-Engine selbst verwendet.

### § [std.ObjectTransfer](#)

## std.ObjectTransfer

### Beschreibung:

Dieser Job transferiert ein Objekt von einem fremden (nicht lokalen) Workverzeichnis in das lokale Work- oder Cacheverzeichnis oder schreibt das komplette lokale Workverzeichnis in das lokale Cacheverzeichnis oder löscht das lokale Workverzeichnis.

### Parameter:

sAction (STRING): Aktion, die Ausgeführt werden soll

- § FromForeignWorkToLocalWork = transferiert ein Objekt von einem fremden(nicht lokalen) Workverzeichnis ins lokale Workverzeichnis
- § FromForeignWorkToLocalCache = transferiert ein Objekt von einem fremden(nicht lokalen) Workverzeichnis ins lokale Cacheverzeichnis
- § MoveLocalWorkToLocalCache = schreibt das komplette lokale Workverzeichnis in das lokale Cacheverzeichnis
- § DeleteLocalWork = löscht das lokale Workverzeichnis

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des ausgewählten Objekts

#### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Sonstige Jobs

- § [std.CheckSource](#)
- § [std.ConfigVarc](#)
- § [std.DiskSpace](#)
- § [std.FileTransfer](#)
- § [std.GetTemplates](#)
- § [std.IndexDataChanged](#)
- § [std.PackDirectory](#)
- § [std.TransformIndexData](#)
- § [std.ZipDocument](#)

### std.CheckSource

#### **Beschreibung:**

Dieser Job ersetzt eine Pfadvariable mit dem Format '%ETCPATH%' durch den absoluten Pfad und überprüft, ob die angegebene Datei in diesem Pfad existiert.

#### **Parameter:**

Flags (INT): z. Z. nicht unterstützt-> 0 übergeben

Source (STRING): Pfadvariable die ersetzt werden soll und Name der Datei, für die Existenz überprüft werden soll

#### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

#### **Rückgabewerte:**

State (INT): 0 = Datei existiert nicht; 1 = Datei existiert

#### **Hinweis:**

unterstützte Pfadvariablen

- § %SERVERROOT%



§ %ETCPATH%

§ %WORKPATH%

§ %LOGPATH%

§ %CLIENTETC%

## std.ConfigVarc

### Beschreibung:

Dieser Job kann nur aus dem Enterprise Manager aufgerufen werden. Dieser Job konfiguriert die Parameter der Virtuellen Archive.

### Parameter:

SAction (STRING): Aktion, die ausgeführt werden soll

§ Get = liefert die Parameter der VARC

§ Set = setzt die Parameter der VARC

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.DiskSpace

### Beschreibung:

Dieser Job ermittelt den freien Speicherplatz der Festplatte, auf welcher das Workverzeichnis liegt und sendet diese Informationen per E-Mail an Systemadministrator (definiert in Registry).

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.FileTransfer

### Beschreibung:

Dieser Job überträgt eine oder mehrere Dateien von einem Quellpfad in einen Zielpfad (Flags = 0/2), löscht angegebene Dateien (Flags = 4) oder liefert den Digest-Wert einer Datei (Flags = 5).

### Parameter:

Flags (INT): Optionen des Jobs (siehe Tabelle)

FileCount (INT): Anzahl der übergebenen File\_ - Parameter

File\_[0..n] (STRING): siehe Tabelle

Dateiliste: siehe Tabelle

### Rückgabewerte:

Count (INT): immer 0 (nicht mehr genutzt)

[Digest]: Digest-Wert der Datei (nur bei Flags = 5)

### Hinweis:

Bei den jeweiligen Optionen des Job werden nur bestimmte Parameter verwendet. Die folgende Tabelle soll die Parameter-Verwendung darstellen

Eingabeparameter				Rückgabe
Flags	FileCount	File_[0..n]	Dateiliste	Digest
0 = Dateien werden mit WinApi 32-Funktion 'CopyFile' kopiert	Anzahl der übergebenen File_ - Parameter	ohne Eingabe-Dateien: File_0 = Zielpfad+Dateiname File_1 = Quellpfad+Dateiname File_2 = Zielpfad+Dateiname File_3 = Quellpfad+Dateinam usw.  mit Eingabe-Dateien: File_0 = Zielpfad+Dateiname File_1 = Zielpfad+Dateiname usw.	Pfad und Name der Quelldatei(en)	nein
2 = Dateien werden mit WinApi32-Funktionen 'CreateFile', 'ReadFile' und 'WriteFile' kopiert	Anzahl der übergebenen File_ - Parameter	ohne Eingabe-Dateien: File_0 = Zielpfad+Dateiname File_1 = Quellpfad+Dateiname File_2 = Zielpfad+Dateiname File_3 = Quellpfad+Dateinam usw.  mit Eingabe-Dateien: File_0 = Zielpfad+Dateiname File_1 = Zielpfad+Dateiname usw.	Pfad und Name der Quelldatei(en)	nein
4 = Dateien werden gelöscht	Anzahl der übergebenen File_ - Parameter	File_0 = Pfad+Dateiname File_1 = Pfad+Dateiname usw.	nein	nein
5 = Digest-Wert wird berechnet	Anzahl der übergebenen File_ - Parameter	File_0 = Pfad+Dateiname File_1 = Pfad+Dateiname usw.	nein	Berechneter Digest-Wert für Datei(en).

## std. GetTemplates

### Beschreibung:

Dieser Job liefert für den angegebenen Objekttyp eine Liste aller Vorlagen. Wird der Parameter 'dwObjectType' nicht oder mit dem Wert -1 übergeben, werden für alle Objekttypen die Vorlagen zurückgeliefert.

### Parameter:

[dwObjectType] (INT): Typ des Objekts, für den die Vorlagen zurückgeliefert werden sollen

**Rückgabewerte:**

nCount (INT): Anzahl der Vorlagen

ObjectType[1 ... nCount] (INT): Typ des ausgewählten Objektes

TemplateId[1 ... nCount] (INT): Vorlage-ID

Aliase[1 ... nCount] (STRING): Aliasname

Editor[1 ... nCount] (STRING): Editorname

FileName[1 ... nCount] (STRING): Dateiname

Extension[1 ... nCount] (STRING): Endungname

NameSpace[1 ... nCount] (STRING): Namespacename

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.IndexDataChanged

**Beschreibung:**

Dieser Job teilt dem Indexserver mit, dass sich die Indexdaten eines Objektes geändert haben.

**Parameter:**

Flags (INT): 1 = wird in DB-Tab 'objectXXX' das Feld 'osowner' geändert; ansonsten wird Feld 'osowner' nicht verändert

Action (INT): Aktion, die mit dem Objekt ausgeführt wurde (Aktionen sind in der Datei asdll.h aufgelistet)

dwObjectID (INT): ID des Objekts

dwObjectType (INT): Typ des Objekts

Info (STRING): Information, welche in DB-Tab 'osobjhist' geschrieben wird

GUID (STRING): eindeutiger Schlüssel DB-Tab 'osobjhist'

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## std.PackDirectory

**Beschreibung:**

Dieser Job packt den Inhalt des Volltexverzeichnisses in eine CAB-Datei und sendet diese Datei zum Empfänger-Server. Der Empfänger-Server entpackt CAB-Datei und fügt alle Volltexindexdateien in sein Volltextverzeichnis ein.

**Parameter:**

dwPort (INT): IP-Port vom Empfänger-Server

sComString (STRING): IP-Adresse vom Empfänger-Server in Form addr1#port1;addr2#port2; usw. Dabei entspricht 'addr' der Spalte ComString aus der Tabelle Server.

sAction (STRING): Optionen für diesen Job

§ Pack = Inhalt des Volltexverzeichnisses in eine CAB-Datei verpacken

§ Send = CAB-Datei senden

§ DeleteSource = Inhalt des Volltextverzeichnisses wird gelöscht

sRoot (STRING): Verzeichnis, aus dem die Unterzeichnisse und Dateien gepackt werden sollen

dwCabSize (INT): max. Größe der CAB-Datei in KB

sCabDir (STRING): Verzeichnis, in den die CAB-Dateien gespeichert werden sollen

sAddress (STRING): String, in dem alle Server der Zielgruppe aufgelistet sind (Format wie AS.ini-Datei)

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## **std.TransformIndexData**

### **Beschreibung:**

Dieser Job exportiert die Volltextindexdaten gemäß seiner Parameter. Dieser Job wird i.A. aus der axacwexp.dll aufgerufen.

### **Parameter:**

dwObjectType (INT): Objekttyp

sAction (STRING): Optionen für die Aktion

§ DeleteIndex = VolltextIndexDaten werden gelöscht

§ NewIndex = neue Volltextindexdaten werden eingefügt

§ UpdateIndex = gleicht 'NewIndex', aber zusätzlich zu RW wird das Feld '<UPDATE>' eingefügt.

bIgnoreFieldSpecific (BOOL): Volltexteigenschaften für Felder

§ 0 = Volltexteigenschaften werden von jedem Feld beachtet

§ 1 = die Volltexteigenschaften werden von jedem Feld ignoriert

[dwObjectID] (INT): ID des Objektes (Wenn ID = 0 oder nicht in ListParameterIn vorhanden, dann werden alle Objekte mit Typ = dwObjectType betrachtet. Ansonsten wird nur dieses Objekt betrachtet.)

[bWriteFiles] (BOOL): Exportoptionen (Standardwert ist 0)

§ 0 = nur Indexdaten (ohne Dokumentdateien) werden nach Volltext exportiert

§ 1 = die Dokumentdateien werden nach Volltext exportiert

[iAction] (INT): Aktion die Ausgeführt werden soll (definiert in der Datei asdll.h)

## **std.ZipDocument**

### **Beschreibung:**

Dieser Job komprimiert eine oder mehrere Datei(en).

### **Parameter:**

bDeleteSource (INT): 1 = die Quelldateien werden gelöscht, ansonsten 0

Eingabe-Dateiliste: Name und Pfad der Datei(en)

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Volltext-Engine (Namespace vtx)

Die Volltext-Engine dient der Bearbeitung von Volltextanfragen der Clients. Dabei können transparent für den Client unterschiedliche Suchmaschinen eingebunden werden.

Unterstützt werden derzeit folgende Volltextsuchmaschinen:

Name	Kurzname (Registry)
Microsoft SQL Server Volltext	ms
OSFTS	lu
Microsoft Index Server (abgekündigt)	mi
Convera RetrievalWare	rw

Die OSFTS-Engine wird oft auch als *Lucene-Engine* oder auch nur osfts bezeichnet.

Der Microsoft Index Server wird noch unterstützt, aber für Neuinstallationen nicht mehr empfohlen.

Volltext-Engine-Jobs

§ [vtx.CleanupClient](#)

§ [vtx.CloseQuery](#)

§ [vtx.GetDocument](#)

§ [vtx.GetEngineName](#)

§ [vtx.GetSimilarDMSObjects](#)

§ [vtx.GetMaxHits](#)

§ [vtx.IsOntologySearchEnabled](#)

§ [vtx.IsSearchForSimilarDMSObjectsEnabled](#)

§ [vtx.OpenObjectQuery](#)

§ [vtx.OpenWordListQuery](#)

§ Bei jeder Jobbeschreibung wird unter „Engines“ als Kurzname vermerkt, bei welcher Volltext-Engine der Job unterstützt wird. Unterstützt eine Volltext-Engine einen Job nicht, so hat die Ausführung des Jobs keine Auswirkung und es wird gegebenenfalls eine leere Ergebnismenge zurückgeliefert.

### [vtx.CleanupClient](#)

#### **Beschreibung:**

Dieser Job löscht alle Volltextanfragen für die angegebene Client-Anwendung und gibt damit verbundene Ressourcen frei.

#### **Parameter:**

Flags (INT): z. Z. nicht unterstützt

Client (STRING): Client

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Engines:**

rw

## [vtx.CloseQuery](#)

**Beschreibung:**

Dieser Job schließt eine Abfrage, die mittels einer Guid identifiziert ist und gibt dabei Ressourcen frei.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Guid (STRING): Guid der Abfrage. Siehe GUID-Rückgabeparameter bei OpenObjectQuery.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Engines:**

lu, rw

## [vtx.GetDocument](#)

**Beschreibung:**

Dieser Job liefert ein Dokument, das durch Abfrage-Guid und HitId identifiziert wird.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Guid (STRING): Guid der Abfrage. Siehe GUID Rückgabeparameter bei OpenObjectQuery.

HitId (INT): Hit-ID des Dokuments

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Dateiliste: Pfad und Name der angeforderten Datei

**Engines:**

lu, rw

## [vtx.GetEngineName](#)

**Beschreibung:**

Dieser Job liefert den Kurznamen der Volltextengine (z. B. ms: für Microsoft SQL Server Volltext) aus der Registry (Pfad VTX->Engine).

**Parameter:**

Flags (LONG): z. Z. nicht verwendet

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result (STRING): Kurzname der Volltextengine. Siehe Liste der unterstützten Volltext-Engines.

**Engines:**

Alle Engines

## vtx.GetSimilarDMSObjects

**Beschreibung:**

Dieser Job liefert zu einem DMS Objekt eine Liste von anderen Objekten, deren textueller Inhalt dem übergebenen Objekt ähnlich ist.

**Hinweis:**

Diese Suche nach Objekten wird derzeit nur unterstützt, wenn als Volltext-Engine OSFTS verwendet wird, dort als Analyzer 'intrafind' konfiguriert ist und die Lizenz OKM vorhanden ist. Ob die Suche unterstützt wird, kann mit [vtx.IsSearchForSimilarDMSObjectsEnabled](#) ermittelt werden.

**Parameter:**

ObjectID (INT): ID des DMS Objekts, zu den ähnliche Objekte gefunden werden sollen.

MaxHits (INT): optionaler Parameter, der die maximale Anzahl zurückzuliefernder Treffer bestimmt. Wird dieser Parameter nicht übertragen, wird die maximale Trefferanzahl anhand von Registry-Einträgen ermittelt.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result (STRING): besteht aus Objekt-ID, Objekt-Typ, Ranking und Hit-ID (durch Komma getrennt) für jedes gefundene Dokument (durch Semikolon getrennt)

**Engines:**

lu

**Siehe auch:**

[vtx.IsSearchForSimilarDMSObjectsEnabled](#)

## vtx.IsOntologySearchEnabled

**Beschreibung:**

Dieser Job gibt an, ob die Volltextengine die Suche nach ähnlichen Begriffen unterstützt.

**Parameter:**

- keine

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result (BOOL): Wird die Suche nach ähnlichen Begriffen unterstützt?



**Engines:**

Alle Engines. TRUE aber zur Zeit nur bei ,lu'.

**Siehe auch:**

[vtx.OpenObjectQuery](#)

## vtx.IsSearchForSimilarDMSObjectsEnabled

**Beschreibung:**

Dieser Job gibt an, ob die Volltextengine die Suche nach DMS-Dokumenten mit ähnlichem Inhalt unterstützt.

**Parameter:**

- keine

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result (BOOL): Wird die Suche nach Dokumenten mit ähnlichem Inhalt unterstützt?

**Engines:**

Alle Engines. TRUE aber zur Zeit nur bei ,lu'.

**Siehe auch:**

[vtx.GetSimilarDMSObjects](#)

## vtx.GetMaxHits

**Beschreibung:**

Dieser Job liefert die Server-seitig konfigurierte Obergrenze für die max. Trefferzahl bei Volltextsuchen. Beim Aufruf aller Volltextsuch-Jobs sollte sichergestellt werden, dass die dort gegebenenfalls angebare Trefferobergrenze nicht größer ist als das Server-seitig eingestellte Maximum ist.

**Parameter:**

- keine

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result (INT): Server-seitig konfigurierte Obergrenze für die maximale Trefferanzahl, die Clients beim Aufruf von Volltextsuch-Jobs verwenden sollten.

**Engines:**

Alle Engines.

**Siehe auch:**

[vtx.OpenObjectQuery](#), [vtx.GetSimilarDMSObjects](#), [vtx.OpenWordListQuery](#)

## vtx.OpenObjectQuery

### Beschreibung:

Dieser Job stellt eine Anfrage nach einem Object und gibt das Ergebnis zurück.

### Hinweis:

Diese ontologische Suche (Query-Parameter ONTOLOGY=1) wird derzeit nur unterstützt, wenn als Volltext-Engine OSFTS verwendet wird, dort als Analyzer 'intrafind' konfiguriert ist und die Lizenz OKM vorhanden ist. Ob die Suche unterstützt wird, kann mit [vtx.IsOntologySearchEnabled](#) ermittelt werden.

Wenn OSFTS mit 'intrafind' verwendet wird, muss zudem generell die Lizenz LIS vorhanden sein, ansonsten liefert der Job-Aufruf den Fehlercode 0xC1DA0BDA.

### Parameter:

Flags (INT): Flags, die der Volltextengine übergeben werden

Query (STRING): Anfrage im INI-Format. Siehe Beispiel.

MessageLanguageId (INT): Optionaler Parameter für die ID der Sprache, in der eine gegebenenfalls gelieferte Meldung (s. Rückgabewert 'Message') verfasst werden soll (Windows Sprach IDs, s. z. B. <http://msdn.microsoft.com/en-us/library/ms776294.aspx>; z. B. 7=deutsch)

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode, z. B. 0xC1DA0BDA, wenn OSFTS mit 'intrafind' verwendet wird und die Lizenz LIS fehlt.

### Rückgabewerte:

Result (STRING): besteht aus Objekt-ID, Objekt-Typ, Ranking und Hit-ID (durch Komma getrennt) für jedes gefundene Dokument (durch Semikolon getrennt)

AlternativeQuerySuggestion (STRING): Vorschlag für alternative Suchbegriffe. Dieser Rückgabewert wird nur geliefert, wenn als Volltext-Engine OSFTS verwendet wird und wenn von dort ein Alternativvorschlag kommt.

OntologyTerms (STRING): Liste von verwandten Suchbegriffen. Format: <Begriff>,<Verwandtheitsgrad (Prozentangabe)>;... . Z. B. 'Flug,55;Reise,43;Urlaub,12;'. Dieser Rückgabewert wird nur geliefert, wenn in der Anfrage ONTOLOGY=1 gesetzt ist.

Guid (STRING): Guid

Message (STRING): Meldung zum Suchresultat

### Engines:

Alle Engines.

### Beispiel:

Bei Verwendung des Microsoft SQL Servers werden nur die ersten drei Zeilen benötigt.

```
[PAGE00]
#OSTYPE#=262144
FULLTEXT=Pferd
ONTOLOGY=1
#OSACT#=1
FELD0=#OSPOS000#;Autor;feld1;X;50;0;0
FELD1=#OSPOS001#;Quelle;feld2;X;150;0;0
FELD2=#OSPOS002#;Text2;feld3;X;50;0;0
```

```
[ PDMSPParams ]  
EXPANSION_LEVEL_PROPERTY=4  
FUZZY_SPELL_HALF_WORDS=FALSE  
FUZZY_SPELL_THRESHOLD=0  
MAX_FUZZY_SPELL_PROPERTY=15  
MAX_REG_EXPR_PROPERTY=50  
WARN_MAX_REG_EXPR_PROPERTY=FALSE  
WORD_EXPANSION_LIMIT_PROPERTY=20  
LMPI_SET_LANG_PROPERTY=de  
MAX_DOCS_PROPERTY=999  
RwareQueryType=P  
OSQueryType=Object  
OSSelectedTypes=current
```

**Siehe auch:**

[vtx.IsOntologySearchEnabled](#)

## vtx.OpenWordListQuery

**Beschreibung:**

Dieser Job führt eine Anfrage nach einer Wortliste aus. Dieser Job kann derzeit nur bei Verwendung von RetrievalWare als Volltextsoftware ein sinnvolles Ergebnis liefern.

**Parameter:**

Flags (INT): Flags, die der Volltextengine übergeben werden

Query (STRING): Anfrage

Param (STRING): Parameter für die Anfrage

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Result (STRING): besteht aus Objekt-ID, Objekt-Typ, Ranking und Hit-ID (durch Komma getrennt) für jedes gefundene Dokument (durch Semikolon getrennt)

Guid (STRING): Guid

**Engines:**

rw

## Workflow-Engine (Namespace wfm)

Hier werden Jobs für die Bearbeitung und Verwaltung von Workflowprozessen und Modellen bereitgestellt.

Der Aufruf von Workflowjobs aus serverseitigen Workflowevents über den 'running context' ist nicht möglich. Hierbei kommt es zum Absturz des Servers.

### Bereiche

- § [Organisationsstruktur](#)
- § [Workflowmodell](#)
- § [Workflowprozess und Arbeitsschritt](#)
- § [Workflow-Maske, Event und Skript](#)
- § [Administration und Historienverwaltung](#)
  - § [Adminstration](#)
  - § [Historienverwaltung](#)
- § [Sonstige Jobs](#)
- § [Serverinterne Jobs](#)

### Organisationsstruktur

- § [wfm.ConfigUserAbsence](#)
- § [wfm.DeleteOrganisation](#)
- § [wfm.GetAbsentUsers](#)
- § [wfm.GetOrganisationClasses](#)
- § [wfm.GetOrganisationObjects](#)
- § [wfm.GetOrganisations](#)
- § [wfm.GetSubstitutes](#)
- § [wfm.SaveOrganisation](#)
- § [wfm.SetActiveOrganisation](#)
- § [wfm.SetSubstitutes](#)

### wfm.ConfigUserAbsence

#### **Beschreibung:**

Dieser Job definiert einen oder mehrere Benutzer als abwesend/anwesend und benachrichtigt alle Server und enaio® editor-for-workflow.

#### **Parameter:**

OrganisationId (STRING): Organisation der Benutzer

Users (BASE64): Benutzerliste im XML-Format

#### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Beispiel:**

Aufbau von Users

```
<Users>
<User Id=" " Absent="1" />
<User Id=" " Absent="0" />
</Users>
```

**Hinweis:**

Genauere Beschreibung von Users

User: Struktur mit folgenden Inhalten

§ Id (STRING): ID des Benutzers

§ Absent (LONG): Flag

§ 0 = Benutzer ist anwesend

§ 1 = Benutzer ist abwesend

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationObjects](#) , [wfm.GetAbsentUsers](#)

## wfm.DeleteOrganisation

**Beschreibung:**

Dieser Job löscht eine Organisation. Es werden alle Datenbankeinträge, die die Organisation betreffen, aus der Datenbank gelöscht. (u.a Organisationsstruktur und Workflowmodelle)

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[wfm.GetOrganisations](#)

## wfm.GetAbsentUsers

**Beschreibung:**

Dieser Job ermittelt alle Benutzer einer Organisation, die abwesend gemeldet sind.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

ObjectIds (STRING): GUIDs der abwesenden Benutzer-Objekte, durch Komma getrennt

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.ConfigUserAbsence](#)

## wfm.GetOrganisationClasses

### Beschreibung:

Dieser Job liefert Informationen zu den Klassen einer Organisation.

### Parameter:

OrganisationId (STRING): ID der Organisation

RequestType (LONG): Flag spezifiziert die Anfrage, die nachfolgenden Parameter müssen entsprechend gesetzt werden

§ 0 = alle Klassen suchen

§ 1 = Suche erfolgt für ID's im Parameter 'ClassIds'

§ 2 = Suche erfolgt für Namen im Parameter 'ClassName'

ClassIds (STRING): kommaseparierte Liste von Klassen-Ids

ClassName (STRING): Klassen-Name

AttributeId (STRING): wird z. Z. nicht unterstützt

RequestData (INT): spezifiziert die Ergebnisse der Anfrage

§ 1 = nur Index-Daten der Klassen ermitteln (Objektid, Name, Klassenid)

§ 3 = Index-Daten und Klassen-Attribute ermitteln

§ 5 = Index-Daten und Klassen-Beziehungen ermitteln

§ 7 = Index-Daten, Klassen-Attribute und Klassen-Beziehungen ermitteln

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Classes (BASE64): Informationen zu den angeforderten Klassen im XML-Format

### Beispiel:

#### Aufbau von Classes

```
<Classes>
<Class Id="10B9A20E90244BB9B701354C1AB84F8A" Name="Abteilung">
<Attributes/>
<ParentObjects>
<ParentObject Id="894CB679F2ED480A89107BF33A1F" Name="Organisation"/>
</ParentObjects>
<ChildClasses>
<ChildClass Id="12AA95D1D8244E6BB56C70A8D5CEE675" Name="Rolle"/>
</ChildClasses>
</Class>
<Class Id="9AB24246BB9040A29FCD6015CF4F4BD9" Name="Person">
<Attributes>
<Attribute Id="33D4AB4B39" Name="Nachname" AttributeClassId="07F405D">
<AttributeValue><![CDATA[ ]]></AttributeValue>
</Attribute>
</Attributes>
<ParentObjects>
<ParentObject Id="12AA95D1D8244E6BB56C70A8D5CEE675" Name="Rolle"/>
</ParentObjects>
```

```
<ChildClasses/>
</Class>
</Classes>
```

**Hinweis:**

Genauere Beschreibung von Classes

- § Class: Struktur, die Informationen zu einer Organisations-Klasse beinhaltet
  - § Id (STRING): ID der Klasse
  - § Name (STRING): Name der Klasse
- § Attribute: Struktur, die Informationen zu einem Objekt-Attribut beinhaltet
  - § Id (STRING): ID des Attributs
  - § Name (STRING): Name des Attributs
  - § AttributeClassId (STRING): Attribut-KlassenId des Attributs
  - § AttributeValue: CDATA mit Attributwert, ggf. MIME-codiert
- § ParentObject: Struktur, die Informationen zu einem Organisations-Objekt, das im Organisations Baum direkt über dem momentanen Klasse steht, beinhaltet
  - § Id (STRING): ID des Objekts
  - § Name (STRING): Name des Objekts
- § ChildClass: Struktur, die Informationen zu einem Organisations-Objekt, das im Organisations Baum direkt unter dem momentanen Klasse steht, beinhaltet
  - § Id (STRING): ID des Objekts
  - § Name (STRING): Name des Objekts

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationObjects](#)

## wfm.GetOrganisationObjects

**Beschreibung:**

Dieser Job liefert Informationen zu den Objekten in einer Organisation.

**Parameter:**

OrganisationId (STRING): ID der Organisation

RequestType (INT): Flag spezifiziert die Anfrage, die nachfolgenden Parameter müssen entsprechend gesetzt werden

- § 0 = alle Objekte suchen
- § 1 = Suche erfolgt für ID's im Parameter 'ObjectIds'
- § 2 = Suche erfolgt für Namen im Parameter 'ObjectName'
- § 3 = Suche erfolgt für ID's im Parameter 'ClassIds'
- § 4 = Suche erfolgt für Namen im Parameter 'ClassName'
- § 5 = Suche der Vorgänger-Objekte erfolgt für Namen im Parameter 'ObjectName'
- § 6 = Suche der Nachfolger-Objekte erfolgt für Namen im Parameter 'ObjectName'
- § 7 = Suche der Vorgänger-Objekte erfolgt für Namen im Parameter 'ClassName'

§ 8 = Suche der Nachfolger-Objekte erfolgt für Namen im Parameter 'ClassName'

ObjectIds (STRING): kommaseparierte Liste von Objekt-Ids

ObjectName (STRING): Objekt-Name

ClassIds (STRING): kommaseparierte Liste von Klassen-Ids

ClassName (STRING): Klassen-Name

AttributeId (STRING): wird z. Z. nicht unterstützt

AttributeValue (STRING): wird z. Z. nicht unterstützt

RequestData (INT): spezifiziert die Ergebnisse der Anfrage

§ 1 = nur Index-Daten der Objekte ermitteln (Objektid, Name, Klassenid)

§ 3 = Index-Daten und Objekt-Attribute ermitteln

§ 5 = Index-Daten und Vorgänger- und Nachfolger-Objekte ermitteln

§ 7 = Index-Daten, Objekt-Attribute und Vorgänger- und Nachfolger-Objekte ermitteln

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Objects (BASE64): Informationen zu den angeforderten Objekten im XML-Format

### Beispiel:

#### Aufbau von Objects

```
<Objects>
<Object Id="" Name="" ClassId="">
<Attributes>
<Attribute Id="" Name="" AttributeClassId="">
<AttributeValue><![CDATA[]]></AttributeValue>
</Attribute>
<Attribute Id="" Name="" AttributeClassId="">
<AttributeValue><![CDATA[]]></AttributeValue>
</Attribute>
</Attributes>
<ParentObjects>
<ParentObjects Id="" Name="" ClassId="">
</ParentObjects>
<ChildObjects>
<ChildObjects Id="" Name="" ClassId="">
</ChildObjects>
</Object>
</Objects>
```

### Hinweis:

#### Genauere Beschreibung von Objects

§ Object: Struktur, die Informationen zu einem Organisations-Objekt beinhaltet

§ Id (STRING): ID des Objekts

§ Name (STRING): Name des Objekts

§ ClassId (STRING): KlassenId des Objekts

§ Attribute: Struktur, die Informationen zu einem Objekt-Attribut beinhaltet

§ Id (STRING): ID des Attributs



- § Name (STRING): Name des Attributs
- § AttributeClassId (STRING): Attribut-KlassenId des Attributs
- § AttributeValue: CDATA mit Attributwert, ggf. MIME-codiert
- § ParentObject: Struktur, die Informationen zu einem Organisations-Objekt, das im Organisations Baum direkt über dem momentanen Objekt steht, beinhaltet
  - § Id (STRING): ID des Objekts
  - § Name (STRING): Name des Objekts
  - § ClassId (STRING): KlassenId des Objekts
- § ChildObject: Struktur, die Informationen zu einem Organisations-Objekt, das im Organisations Baum direkt unter dem momentanen Objekt steht, beinhaltet
  - § Id (STRING): ID des Objekts
  - § Name (STRING): Name des Objekts
  - § ClassId (STRING): KlassenId des Objekts

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationClasses](#)

## wfm.GetOrganisations

### Beschreibung:

Dieser Job liefert alle definierten Organisationen und gibt an, welche Organisation aktiviert ist. Über den enaio® editor-for-workflow kann immer nur eine Organisation aktiviert werden.

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Organisations (BASE64): Informationen zu allen verfügbaren Organisationen im XML-Format

### Beispiel:

Aufbau von Organisations

```
<Organisations>
<Organisation Id=" " Name=" " Active="0"/>
<Organisation Id=" " Name=" " Active="1"/>
</Organisations>
```

### Hinweis:

Genauere Beschreibung von Organisations

Organisation: Struktur, die Informationen zu einer Organisation beinhaltet

- § Id (STRING): ID der Organisation
- § Name (STRING): Name der Organisation
- § Active (INT): gibt an, ob die Organisation aktiv (1) ist oder nicht (0)

## wfm.GetSubstitutes

### Beschreibung:

Dieser Job ermittelt die Stellvertreter für beliebig viele Benutzer und Rollen.

**Parameter:**

OrganisationId (STRING): ID der Organisation

UserIds (STRING): kommaseparierte Liste von Ids der Benutzer/Rollen, zu denen Stellvertreterinformationen angefordert werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Substitutes (BASE64): angeforderte Stellvertreterinformation im XML-Format

**Beispiel:**

Aufbau von Substitutes

Beispiel:

Aufbau von Substitutes

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Substitutes>
<Object Id="BB4FF62D3FE24DD790DE342585917A36">
<Substitute Id="8CD236F862644DAD904CD8C228EF4F23" Name="LEHMANN" Absent="1"
UserGUID="4FEEDE694EB94E7E9C4A847FC32D10E4" Login="Lehmann" />
</Object>
</Substitutes>
```

**Hinweis:**

Genauere Beschreibung von Substitutes

§ Id (String): Objektid des Benutzers

§ Substitute, Struktur die alle Stellvertreterzuordnungen für bestimmten Benutzer/Rolle enthält

§ Id (STRING): ObjectId des Stellvertreters für Benutzer/Rolle

§ Name (STRING): Loginname des Stellvertreters

§ Absent (INT): 1= Benutzer ist abwesend, 0 = sonst

§ UserGUID (STRING): GUID des Benutzers aus der Benutzertabelle

§ Login (STRING): Login des Benutzers aus der Benutzertabelle

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationObjects](#) , [wfm.SetSubstitutes](#)

## wfm.GetUserSubstitutes

**Beschreibung:**

Dieser Job alle Benutzer, die der anfragende Benutzer vertritt

**Parameter:**

OrganisationId (STRING): ID der Organisation

UserId (STRING): Id des Benutzers, für den die Informationen angefragt werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Objects (BASE64): angeforderte Information im XML-Format

**Beispiel:**

Aufbau von Objects

Beispiel:

Aufbau von Objects

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Objects>
<Object Id="BB4FF62D3FE24DD790DE342585917A36">
<Substitute Id="8CD236F862644DAD904CD8C228EF4F23" Name="LEHMANN" Absent="1"
UserGUID="4FEEDE694EB94E7E9C4A847FC32D10E4" Login="Lehmann"/>
</Object>
</Objects>
```

**Hinweis:**

Genauere Beschreibung von Objects

§ Id (String): Objektid des Benutzers

§ Object, Struktur die alle zu vertretenden Benutzer enthält

§ Id (STRING): ObjectId des Stellvertreters für Benutzer/Rolle

§ Name (STRING): Loginname des Stellvertreters

§ Absent (INT): 1= Benutzer ist abwesend, 0 = sonst

§ UserGUID (STRING): GUID des Benutzers aus der Benutzertabelle

§ Login (STRING): Login des Benutzers aus der Benutzertabelle

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationObjects](#) , [wfm.SetSubstitutes](#)

## wfm.SaveOrganisation

**Beschreibung:**

Dieser Job speichert eine Organisation und benachrichtigt alle betroffenen Clients.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.SetActiveOrganisation

**Beschreibung:**

Dieser Job aktiviert die angebene Organisation.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.SetSubstitutes

**Beschreibung:**

Dieser Job setzt die Stellvertreter für beliebig viele Benutzer oder Rollen.

**Parameter:**

OrganisationId (STRING): ID der Organisation

Substitutes (BASE64): Benutzer-Stellvertreter-Zuordnungen im XML-Format

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Beispiel:**

### Aufbau von Substitutes

```
<Substitutes>
<Substitute Id="12345678901234567890123456789010">
<SubstituteIds>789012345678901A,123456778901B</SubstituteIds>
</Substitute>
<Substitute Id="12345678901234567890123456789011">
<SubstituteIds>123456789012345,123452345678901C</SubstituteIds>
</Substitute>
</Substitutes>
```

**Hinweis:**

Genauere Beschreibung von Substitutes

Substitute: Struktur die alle Stellvertreterzuordnungen zu einem bestimmten Benutzer enthält

§ Id (STRING): ObjectId d. Benutzers/Rolle

§ SubstituteIds (STRING): kommaseparierte Liste der ObjectIds aller Stellvertreter für den Benutzer/Rolle; kann leer sein

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationObjects](#) , [wfm.GetSubstitutes](#)

## Workflowmodell

§ [wfm.ChangeWorkflowState](#)

§ [wfm.CopyWorkflow](#)

§ [wfm.DeleteWorkflow](#)

§ [wfm.GetWorkflow](#)

§ [wfm.GetWorkflowData](#)

§ [wfm.GetWorkflowInfo](#)

§ [wfm.GetWorkflowList](#)

§ [wfm.GetWorkflowListByFamily](#)

§ [wfm.StoreWorkflow](#)

§ [wfm.ValidateWorkflow](#)

## wfm.ChangeWorkflowState

### Beschreibung:

Dieser Job ändert/setzt den Status eines Workflowmodells.

### Parameter:

OrganisationId (STRING): ID der Organisation, der der Workflow zugeordnet ist

WorkflowId (STRING): ID des Workflowmodells

UserId (STRING): ID des Benutzers

State (LONG): neuer Status des Workflowmodells, der gesetzt werden soll

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

State (LONG): neuer Status des Workflow-Modells

### Hinweis:

Status eines Workflowmodells

§ 1 = Das Modell wird verwendet, d.h. vom ihm können neue Prozesse gestartet werden.

§ 2 = Das Modell ist zur Bearbeitung gesperrt.

§ 3 = Das Modell ist noch in Bearbeitung, jedoch nicht gesperrt.

§ 4 = Das Modell ist zum Test freigegeben.

§ 5 = Das Modell ist gelöscht, aber noch in der DB enthalten.

§ 6 = Das Modell ist freigegeben, wird jedoch nicht verwendet. Von diesem Modell können keine neuen Prozesse gestartet werden, laufende werden jedoch noch beendet.

### Siehe auch:

[wfm.ValidateWorkflow](#)

## wfm.CopyWorkflow

### Beschreibung:

Dieser Job erstellt eine Kopie eines Workflowmodells inklusive der Workflow-Masken und Events.

### Parameter:

UserId (STRING): ID der ausführenden Benutzers

SourceOrganisationId (STRING): ID der Organisation, aus der kopiert werden soll

SourceWorkflowId (STRING): ID des Workflowmodells, das kopiert werden soll

TargetOrganisationId (STRING): ID der Organisation, in die kopiert werden soll

TargetFamilyId (STRING): ID der WorkflowFamilie, in die kopiert werden soll

TargetWorkflowName (STRING): Name des neuen Workflows

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

WorkflowId (STRING): ID des neuen Workflowmodells

## wfm.DeleteWorkflow

**Beschreibung:**

Dieser Job löscht ein Workflowmodell und dazugehörige Historien-Einträge aus der Datenbank und benachrichtigt alle anderen Server.

**Parameter:**

OrganisationId (STRING): ID der Organisation des Workflowmodells

WorkflowId (STRING): ID des Workflowmodells

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.GetWorkflow

**Beschreibung:**

Dieser Job liefert ein Workflowmodell vom Server.

**Parameter:**

WorkflowId (STRING): ID des Workflowmodells

ProcessId (STRING): ID des Prozesses (Prozess muss aktiv sein) ; kann optional verwendet werden, er ersetzt dann die Parameter 'FamilyId' und 'WorkflowId'; nur für Acton = 1 gültig

OrganisationId (STRING): ID der Organisation des Workflowmodells

UserId (STRING): ID des DRT-Benutzers

FamilyId (STRING): ID der Workflowfamilie des Workflowmodells

Action (INT): Aktion, die ausgeführt werden soll

§ 1 = Workflowmodell wird zum Lesen angefordert

§ 2 = Workflowmodell soll editiert werden

§ 3 = es wird ein leeres Workflowmodell angefordert

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

NewId (STRING): neue ID des angeforderten Workflow, falls erzeugt (siehe Eingabeparameter 'Action')

State (INT): Status des Modells

§ 1 = Das Modell wird verwendet, d.h. vom ihm können neue Prozesse gestartet werden.

§ 2 = Das Modell ist zur Bearbeitung gesperrt.

§ 3 = Das Modell ist noch in Bearbeitung, jedoch nicht gesperrt.

§ 4 = Das Modell ist zum Test freigegeben.

§ 5 = Das Modell ist gelöscht, aber noch in der DB enthalten.

§ 6 = Das Modell ist freigegeben, wird jedoch nicht verwendet. Von diesem Modell können keine neuen Prozesse gestartet werden, laufende werden jedoch noch beendet.

FamilyId (STRING): ID der Familie des Workflows

CreationTime (INT): Erstellzeit des Workflow

Version (INT): Version des Workflows

Dateiliste: Name und Pfad der Datei, die das XML-Package mit Modellbeschreibung im XML-Format enthält

**Siehe auch:**

[wfm.StoreWorkflow](#)

## wfm.GetWorkflowData

**Beschreibung:**

Dieser Job liefert zu einer Workflowid die Status-Informationen eines Workflowmodells.

**Parameter:**

OrganisationId (STRING): ID der Organisation des Workflowmodells

WorkflowId (STRING): ID des Workflowmodells

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

WorkflowId (STRING): ID des Workflowmodells

WorkflowName (STRING): Name des Workflowmodells

WorkflowState (INT): Status des Workflowmodells

WorkflowCreator (STRING): ID des Erstellers des Workflowmodells

WorkflowCreationTime (INT): Erstellzeitpunkt des Workflowmodells

WorkflowVersion (STRING): Version des Workflowmodells

WorkflowLockId (STRING): ID des Benutzers der das Workflowmodell gesperrt hat

WorkflowLockName (STRING): Name des Benutzers der das Workflowmodell gesperrt hat

WorkflowLockTime (INT): Sperrzeitpunkt des Workflowmodells

WorkflowDescription (STRING): Kurz-Beschreibung des Workflowmodell

WorkflowIconId (INT): IconId des Workflowmodells

**Hinweis:**

Status eines Workflowmodells

§ 1 = Das Modell wird verwendet, d.h. vom ihm können neue Prozesse gestartet werden.

§ 2 = Das Modell ist zur Bearbeitung gesperrt.

§ 3 = Das Modell ist noch in Bearbeitung, jedoch nicht gesperrt.

- § 4 = Das Modell ist zum Test freigegeben.
- § 5 = Das Modell ist gelöscht, aber noch in der DB enthalten.
- § 6 = Das Modell ist freigegeben, wird jedoch nicht verwendet. Von diesem Modell können keine neuen Prozesse gestartet werden, laufende werden jedoch noch beendet.

## wfm.GetWorkflowInfo

### Beschreibung:

Dieser Job liefert zu einem aktiven Workflowmodell (Status = 1) der angegebenen Workflowfamilie die Eingabeparameter (Workflowvariablen).

### Parameter:

OrganisationId (STRING): ID der Organisation des Workflowmodells

FamilyId (STRING): ID der Familie des aktiven Workflowmodells

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

InputParams (BASE64): Eingabeparameter des Modells im XML-Format

### Beispiel:

Aufbau von InputParams

```
<InputParams>
<InputParam Id=" " Name=" "><![CDATA[ ]]></InputParam>
<InputParam Id=" " Name=" "><![CDATA[ ]]></InputParam>
</InputParams>
```

### Hinweis:

Genauere Beschreibung von InputParams

#### § InputParam

- § Id (STRING): ID des Eingabeparameters
- § Name (STRING): Name des Eingabeparameters
- § CDATA: Aufbau des Eingabeparameters

### Siehe auch:

[wfm.GetWorkflowList](#)

## wfm.GetWorkflowList

### Beschreibung:

Dieser Job liefert eine Liste aller startbaren Workflows für den angegebenen Benutzer.

### Parameter:

OrganisationId (STRING): ID der Organisation

UserId (STRING): ID des Benutzers

Flags (INT): Markierungsparameter, z. Z. nur Wert 48 gültig



ClientTypeId (STRING): ID des verwendeten Clienttyps

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Workflows (BASE64) Liste mit Daten zu den angeforderten Workflows im XML-Format

**Beispiel:****Aufbau von Workflows**

```
<Workflows>
<Workflow FamilyId="" ModelName="" Name="" Id="" Description="" IconId="" />
<Workflow FamilyId="" ModelName="" Name="" Id="" Description="" IconId="" />
</Workflows>
```

**Hinweis:**

Genauere Beschreibung von Workflows

**§ Workflow**

- § FamilyId (STRING): ID der Workflowfamilie des Workflows
- § Id (STRING): ID des Workflows
- § Name (STRING): Name des Workflows (Instanzname)
- § Description (STRING): Beschreibung zum Workflowmodell
- § IconId (INT): Iconid des Workflowmodells
- § ModelName (STRING): Name des Workflowmodells

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetOrganisationObjects](#) , [wfm.CreateProcessInstance](#)

## wfm.GetWorkflowListByFamily

**Beschreibung:**

Dieser Job liefert zu einer Workflowfamilie alle enthaltenen Workflowmodelle. Innerhalb einer Workflowfamilie gibt es immer nur ein Modell, welches den Status = 1 besitzt.

**Parameter:**

OrganisationId (STRING): ID der Organisation

FamilyId (STRING): ID der Workflowfamilie

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Workflows (BASE64): Workflowliste im XML-Format

**Beispiel:****Aufbau von Workflows**

```
<Workflows>
<Workflow Id="" Name="" State="" Creator="" CreationTime=""
Version="" LockId="" LockName="" LockTime="" Description=""
```

```

IconId="" />
<Workflow Id="" Name="" State="" Creator="" CreationTime=""
Version="" LockId="" LockName="" LockTime="" Description=""
IconId="" />
</Workflows>

```

**Hinweis:****Genauere Beschreibung von Workflows**

Workflow: Struktur, die Informationen zu einem Workflow beinhaltet

§ Id (STRING): ID des Workflows

§ Name (STRING): Name

§ State (INT): Status des Workflows

§ 1 = Das Modell wird verwendet, d.h. vom ihm können neue Prozesse gestartet werden.

§ 2 = Das Modell ist zur Bearbeitung gesperrt.

§ 3 = Das Modell ist noch in Bearbeitung, jedoch nicht gesperrt.

§ 4 = Das Modell ist zum Test freigegeben.

§ 5 = Das Modell ist gelöscht, aber noch in der DB enthalten.

§ 6 = Das Modell ist freigegeben, wird jedoch nicht verwendet. Von diesem Modell können keine neuen Prozesse gestartet werden, laufende werden jedoch noch beendet.

§ Creator (STRING): Ersteller

§ CreationTime (INT): Erstellungszeit

§ Version: Versionsnummer des Workflows

§ LockId (STRING): ID des Benutzers, der den Workflow gesperrt hat

§ LockName (STRING): Name des Benutzers, der den Workflow gesperrt hat

§ LockTime (INT): Zeitpunkt der Sperrung

§ Description (STRING): Beschreibung zum Workflow

§ IconId (INT): IconId des Workflowmodells

**wfm.StoreWorkflow****Beschreibung:**

Dieser Job ändert/speichert ein Workflowmodell.

**Parameter:**

UserId (STRING): ID des Benutzers

WorkflowId (STRING): ID des Workflowmodells

OrganisationId (STRING): ID der Organisation

FamilyId (STRING): ID der Workflowfamilie

Flags (INT): 1=intern 2=extern (Modell von ausserhalb eingespielt, es wird eine neue ID für das Modell erzeugt)

Eingabe-Datei: Pfad und Name der Datei mit Workflowinformation im XML-Format

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

WorkflowId (STRING): ID des Workflows

Name (STRING): Name des Workflows

State (INT): Status des Workflows

CreatorId (STRING): ID des Erstellers des Workflows

CreationTime (INT): Zeitpunkt des Erstellens des Workflows

Version (STRING): Version des Workflows

LockId (STRING): ID des Benutzers der den Workflow gesperrt hat

LockName (STRING): Name des Benutzers der den Workflow gesperrt hat

LockTime (INT): Zeitpunkt des Sperrens des Workflows

Description (STRING): Kurz-Beschreibung des Workflows

IconId (INT): IconId des Workflowmodells

## [wfm.ValidateWorkflow](#)

### **Beschreibung:**

Dieser Job überprüft ein Workflowmodell auf seine Zulässigkeit.

### **Parameter:**

OrganisationId (STRING): ID der Organisation

Eingabe-Datei: Pfad und Name der Datei mit Beschreibung des Workflowmodells im XML-Format

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

Valid (INT): Flag, das angibt, ob das Modell zulässig ist (1=Ja, 2=Nein)

ErrorCount (INT): Anzahl der im Modell gefundenen Fehler

WarningCount (INT): Anzahl der das Modell betreffenden Warnungen

Errors (BASE64): Informationen über die gefunden Fehler im XML-Format

Warnings (BASE64): Informationen über die Warnungen im XML-Format

### **Beispiel:**

#### Aufbau von Errors

```
<Errors>
<SchemaValidationError>Fehler bei Schemavalidierung!</SchemaValidationError>
<MissingTypeDeclRecordMembers>
<MissingTypeDeclRecordMember TypeDeclId=" " TypeDeclName="TestList"/>
<MissingTypeDeclRecordMember TypeDeclId=" " TypeDeclName="RecFu"/>
</MissingTypeDeclRecordMembers>
<MissingVariableRecordMembers>
<MissingVariableRecordMember VariableId=" " VariableName=" TestList1"/>
</MissingVariableRecordMembers>
<MissingTypeDeclarations>
```

```

<MissingTypeDeclaration Id="12346798134567891345678900001">
<ReferencingDataFields>
<ReferencingDataField Id="1234671345678913456789001" Name="abc"/>
<ReferencingDataField Id="126789134567890000" Name="xyz"/>
</ReferencingDataFields>
<ReferencingTypeDeclarations>
<ReferencingTypeDeclaration Id="123467981345" Name=" TestList"/>
</ReferencingTypeDeclarations>
</MissingTypeDeclaration>
<MissingTypeDeclaration Id="12346798134567891345678900002">
<ReferencingTypeDeclarations>
<ReferencingTypeDeclaration Id="" Name="recBla"/>
</ReferencingTypeDeclarations>
</MissingTypeDeclaration>
</MissingTypeDeclarations>

<MissingActivityParticipants>
<MissingActivityParticipant Id="12346798134567891345678900006">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step1"/>
<ReferencingActivity Id="" Name="step2"/>
</ReferencingActivities>
</MissingActivityParticipant>
<MissingActivityParticipant Id="12346798134567891345678900007">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step1"/>
</ReferencingActivities>
</MissingActivityParticipant>
</MissingActivityParticipants>
<NoParticipantsInStartActivity/>
<MissingToolIds>
<MissingToolId ActivityId="" ActivityName="stepLoop1"/>
<MissingToolId ActivityId="" ActivityName="stepLoop2"/>
</MissingToolIds>
<MissingApplicationMasks>
<MissingApplicationMask ApplicationId="" ApplicationName="Anwendung
Step1" MaskId="12346798134567891345678900008"/>
</MissingApplicationMasks>
<MissingApplicationMaskIds>
<MissingApplicationMaskId ActivityId="" ActivityName="Act Step1" ApplicationId=""
ApplicationName="App Step 1"/>
</MissingApplicationMaskIds>
<MissingActivityApplications>
<MissingActivityApplication ActivityId="" ActivityName="step2"
ApplicationId="12346798134567891345678900010"/>
</MissingActivityApplications>
<InvalidApplicationMaskFieldIds>
<InvalidApplicationMaskFieldId MaskFieldId="123" ApplicationId=""
ApplicationName="AppTest"/>
</InvalidApplicationMaskFieldIds>
<MissingActivityVariables>
<MissingActivityVariable VariableId="12346798134567891345678900012">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step2"/>
</ReferencingActivities>
</MissingActivityVariable>
<MissingActivityVariable VariableId="12346798134567891345678900013">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step2"/>
</ReferencingActivities>
</MissingActivityVariable>
</MissingActivityVariables>
<InvalidParameterListCounts>

```

```

<InvalidParameterListCount ActivityId="" ActivityName="step1"
ApplicationId="" ApplicationName=" App Step1"/>
<InvalidParameterListCount ActivityId="" ActivityName="step3" ApplicationId=""
ApplicationName="App Step3"/>
</InvalidParameterListCounts>
<ParamWithoutVariableActivities>
<ParamWithoutVariableActivity ActivityId="" ActivityName="step1" ApplicationId=""
ApplicationName="App Step1"/>/>
<ParamWithoutVariableActivity ActivityId="" ActivityName="step2 ApplicationId=""
ApplicationName="App Step2"/>/>
</ParamWithoutVariableActivities>
<MissingTakeOverActivities>
<MissingTakeOverActivity ActivityId="12346798134567891345678900014">
<ReferencingActivities>
<ReferencingActivity Id="step1" Name="step1"/>
</ReferencingActivities>
</MissingTakeOverActivity>
<MissingTakeOverActivity ActivityId="12346798134567891345678900015">
<ReferencingActivities>
<ReferencingActivity Id="step2" Name="step2"/>
</ReferencingActivities>
</MissingTakeOverActivity>
</MissingTakeOverActivities>
<MissingTakeOverVariables>
<MissingTakeOverVariable VariableId="12346798134567891345678900016">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step1"/>
</ReferencingActivities>
</MissingTakeOverVariable>
</MissingTakeOverVariables>
<MissingFromActivities>
<MissingFromActivity FromActivityId="12346798134567891345678900017"
TransitionId="12346798134567891345678900018"/>
<MissingFromActivity FromActivityId="12346798134567891345678900019"
TransitionId="12346798134567891345678900020"/>
</MissingFromActivities>
<MissingToActivities>
<MissingToActivity ToActivityId="12346798134567891345678900021"
TransitionId="12346798134567891345678900022"/>
</MissingToActivities>
<RedundantTransitions>
<RedundantTransition FromActivityId="12346798134567891345678900021"
FromActivityName="stepX" ToActivityId="12346798134567891345678900021"
ToActivityName="stepY" TransitionId="12346798134567891345678900021"/>
</RedundantTransitions>
<InvalidFromLoopTransitions>
<InvalidFromLoopTransition
FromActivityId="1234679891345678900021" FromActivityName="stepX"
TransitionId="1234679891345678900021"/>
</InvalidFromLoopTransitions>
<InvalidToLoopTransitions>
<InvalidToLoopTransition ToActivityId="12346798134567891345678900021"
ToActivityName="stepX" TransitionId="12346798134567891345678900021"/>
</InvalidToLoopTransitions>
<InvalidFromOrToLoops>
<InvalidFromOrToLoop LoopActivityId="12346798134567891345678900021"
LoopActivityName="loopX"/>
</InvalidFromOrToLoops>
<CycleActivities>
<CycleActivity Id="12346798134567891345678900021" Name="step58"/>
<CycleActivity Id="12346798134567891345678900021" Name="step42"/>
</CycleActivities>
<InvalidLoopTransitions>
<InvalidLoopTransition TransitionId="12346798134567891345678900021"

```

```

FromActivityId="12346798134567891345678900021" FromActivityName="stepX"
ToActivityId="12346798134567891345678900021" ToActivityName="stepY"/>
</InvalidLoopTransitions>
<InvalidToLoopNumActivities>
<InvalidToLoopNumActivity Id="12367891345678900021" Name="step56"/>
<InvalidToLoopNumActivity Id="81345678913456789001" Name="step57"/>
</InvalidToLoopNumActivities>
<InvalidFromLoopNumActivities>
<InvalidFromLoopNumActivity Id="1234678900021" Name="step59"/>
<InvalidFromLoopNumActivity Id="3467981345678" Name="step60"/>
</InvalidFromLoopNumActivities>
<MissingLoopConditionActivities>
<MissingLoopConditionActivity Id="12891345678900021" Name="step61"/>
</MissingLoopConditionActivities>
<NoTerminationActivities>
<NoTerminationActivity Id="" Name="stepBla"/>
<NoTerminationActivity Id="" Name="stepFoo"/>
</NoTerminationActivities>
<NoLoopTerminationActivities>
<NoLoopTerminationActivity Id="" Name="stepLoop1"/>
<NoLoopTerminationActivity Id="" Name="stepLoop2"/>
</NoLoopTerminationActivities>
<MissingDefActIds>
<MissingDefActId Id="" Name="stepLoop1"/>
<MissingDefActId Id="" Name="stepLoop2"/>
</MissingDefActIds>
<AllClientTypesActivities>
<AllClientTypesActivitiy ActivityId="" ActivityName="XX" ApplicationId=""
ApplicationName="x1"/>
<AllClientTypesActivitiy ActivityId="" ActivityName="YY" ApplicationId=""
ApplicationName="y1"/>
</AllClientTypesActivities>
<AmbiguousActAppClientTypes>
<AmbiguousActAppClientType ActivityId="" ActivityName="ABC" ClientTypeId=""
ClientTypeName="CTX"/>
<AmbiguousActAppClientType ActivityId="" ActivityName="DEF" ClientTypeId=""
ClientTypeName="CTY"/>
</AmbiguousActAppClientTypes>
<ActAppInvalidClientTypes>
<ActAppInvalidClientType ActivityId="" ActivityName="AAB" ApplicationId=""
ApplicationName="v1"/>
<ActAppInvalidClientType ActivityId="" ActivityName="BBX" ApplicationId=""
ApplicationName="w2"/>
</ActAppInvalidClientTypes>
<ActEvtAllClientTypes>
<ActEvtAllClientType ActivityId="" ActivityName="ABC" EventTypeId="100"
EventTypeName="BeforeBlaEvent"/>
<ActEvtAllClientType ActivityId="" ActivityName="DEF" EventTypeId="100"
EventTypeName="AfterFuEvent"/>
</ActEvtAllClientTypes>
<AmbiguousActEvtClientTypes>
<AmbiguousActEvtClientType ActivityId="" ActivityName="ABCDE" EventTypeId="100"
EventTypeName="Before123Event" ClientTypeId="" ClientTypeName="CTX11"/>
<AmbiguousActEvtClientType ActivityId="" ActivityName="DEFGH" EventTypeId="100"
EventTypeName="After456Event" ClientTypeId="" ClientTypeName="CTY22"/>
</AmbiguousActEvtClientTypes>
<AmbiguousGlobalEvtClientTypes>
<AmbiguousGlobalEvtClientType ClientTypeId="" ClientTypeName="CT89"/>
<AmbiguousGlobalEvtClientType ClientTypeId="" ClientTypeName="CT90"/>
</AmbiguousGlobalEvtClientTypes>
<NoProcessStartClientType/>
<InvalidModParamVarIds>
<InvalidModParamVarId VarId="12891345678900021">
<InvalidModParamVarIds/>

```

```

<AdhocActWithoutDefaultSubActs>
<AdhocActWithoutDefaultSubAct Id=" " Name="AdhocActWithoutDefaultSubAct 1"/>
<AdhocActWithoutDefaultSubAct Id=" " Name="AdhocActWithoutDefaultSubAct 2"/>
</AdhocActWithoutDefaultSubActs>
<AdhocActWithUnknownDefaultSubActs>
<AdhocActWithUnknownDefaultSubAct Id=" " Name="AdhocActWithUnknownDefaultSubAct 1"/>
<AdhocActWithUnknownDefaultSubAct Id=" " Name="AdhocActWithUnknownDefaultSubAct 2"/>
</AdhocActWithUnknownDefaultSubActs>
<AdhocActDefaultActIsNotAdhocSubActs>
<AdhocActDefaultActIsNotAdhocSubAct Id=" " Name="AdhocActDefaultActIsNotAdhocSubAct 1"/>
<AdhocActDefaultActIsNotAdhocSubAct Id=" " Name="AdhocActDefaultActIsNotAdhocSubAct 2"/>
</AdhocActDefaultActIsNotAdhocSubActs>
<AdhocActDefaultActIsNotWorkitems>
<AdhocActDefaultActIsNotWorkitem Id=" " Name="AdhocActDefaultActIsNotWorkitem 1"/>
<AdhocActDefaultActIsNotWorkitem Id=" " Name="AdhocActDefaultActIsNotWorkitem 2"/>
</AdhocActDefaultActIsNotWorkitems>
</Errors>

```

### Hinweis:

#### Genauere Beschreibung von Errors

- § SchemaValidationError: Fehlermeldung der Schemavalidierung
- § MissingTypeDeclRecordMember: Struktur die Typdeklaration enthält, die ein Record ohne Member enthält
  - § TypeDeclId (STRING): ID der Typdeklaration
  - § TypeDeclName (STRING): Name der Typdeklaration
- § MissingVariableRecordMember: Struktur die Variable enthält, die ein Record ohne Member enthält
  - § VariableId (STRING): ID der Variablen
  - § VariableName (STRING): Name der Variablen
- § MissingTypeDeclarations
  - § MissingTypeDeclaration: Struktur die verwendete, aber nicht definierte Typdeklaration enthält
    - § Id (STRING): ID mit der auf nicht vorhandene Typdeklaration referenziert wird
- § ReferencingDataFields
  - § ReferencingDataField: Struktur, die ein DataField (Workflowvariable) enthält, das eine nicht definierte Typdeklaration benutzt
    - § Id (STRING): ID des DataFields
    - § Name (STRING): Name des DataFields
- § ReferencingTypeDeclarations
  - § ReferencingTypeDeclaration: Struktur, die eine Typdeklaration enthält, die eine nicht definierte Typdeklaration benutzt
    - § Id (STRING): ID der Typdeklaration
    - § Name (STRING): Name der Typdeklaration
- § MissingActivityParticipants
  - § MissingActivityParticipant: Struktur, die einen Activity-Teilnehmer enthält, der nicht Teilnehmer des Workflows ist
    - § Id (STRING): ID mit der auf nicht vorhandenen Workflow-Teilnehmer referenziert wird

## § ReferencingActivities

§ ReferencingActivity: Struktur, die eine Aktivität enthält die auf die übergeordnete Struktur referenziert

§ Id (STRING): ID der Aktivität

§ Name (STRING): Name der Aktivität

§ NoParticipantsInStartActivity: Tag ist vorhanden, falls der StartActivity keine Teilnehmer zugeordnet wurden

## § MissingToolIds

§ MissingToolId: Struktur, die eine Aktivität enthält, der die Tool-ID fehlt

§ ActivityId (STRING): ID der Aktivität

§ ActivityName (STRING): Name der Aktivität

## § MissingApplicationMasks

§ MissingApplicationMask: Struktur, die eine Applikation enthält, die auf eine nicht-existente Maske referenziert

§ ApplicationId (STRING): ID der Applikation

§ ApplicationName (STRING): Name der Applikation

§ MaskId (STRING): ID mit der auf nicht-existente Maske referenziert wird

## § MissingApplicationMaskIds

§ MissingApplicationMaskId: Struktur, die eine Aktivität und eine zugeordnete Anwendung enthält, der keine Maske zugeordnet ist

§ ActivityId (STRING): ID der Aktivität

§ ActivityName (STRING): Name der Aktivität

§ ApplicationId (STRING): ID der Anwendung

§ ApplicationName (STRING): Name der Anwendung

## § MissingActivityApplications

§ MissingActivityApplication: Struktur, die eine Aktivität enthält, die auf eine nicht-existente Applikation referenziert

§ ActivityId (STRING): ID der Aktivität

§ ActivityName (STRING): Name der Aktivität

§ ApplicationId (STRING): ID mit der auf nicht-existente Applikation referenziert wird

## § InvalidApplicationMaskFieldId: Struktur enthält ungültige Referenz auf Maskenfelder

§ MaskFieldId (STRING): ID des nicht existenten Maskenfelds

§ ApplicationId (STRING): ID der Anwendung

§ ApplicationName (STRING): Name der Anwendung

## § MissingActivityVariables

§ MissingActivityVariable: Struktur, die eine AktivitätenVariable enthält, die nicht im Workflow als DataField (Workflow-Variable) existiert

§ VariableId (STRING): ID mit der auf nicht-existente WorkflowVariable referenziert wird

§ ReferencingActivityApplications



- § ReferencingActivityApplication: Struktur die eine Aktivität-Anwendungszuordnung enthält, für die das Problem besteht
- § ActivityId (STRING): ID der Aktivität
- § ActivityName (STRING): Name der Aktivität
- § ApplicationId (STRING): ID der Anwendung
- § ApplicationName (STRING): Name der Anwendung
- § InvalidParameterListCounts
  - § InvalidParameterListCount: Struktur, die eine Aktivität mit zugehöriger Applikation enthält, bei denen die Anzahl der Parameter nicht übereinstimmt
  - § ActivityId (STRING): ID der Aktivität
  - § ActivityName (STRING): Name der Aktivität
  - § ApplicationId (STRING): ID der Applikation
  - § ApplicationName (STRING): Name der Anwendung
- § ParamWithoutVariableActivities
  - § ParamWithoutVariableActivity: Struktur die eine Aktivität enthält, die Anwendungsparameter ohne Variablenzuordnung enthält
  - § ActivityId (STRING): ID der Aktivität
  - § ActivityName (STRING): Name der Aktivität
  - § ApplicationId (STRING): ID der Applikation
  - § ApplicationName (STRING): Name der Anwendung
- § MissingTakeOverActivities
  - § MissingTakeOverActivity: Struktur, die eine Aktivität enthält, aus der Variablen übernommen werden sollen, wobei diese Aktivität nicht im Workflow existiert
  - § ActivityId (STRING): ID mit der auf nicht-existente Aktivität referenziert wird
- § MissingTakeOverVariables
  - § MissingTakeOverVariable: Struktur, die eine Variable enthält, die in eine Aktivität übernommen werden soll, wobei diese Variable (DataField) nicht im Workflow existiert
  - § VariableId (STRING): ID mit der auf nicht-existente Variable referenziert wird
- § MissingFromActivities
  - § MissingFromActivity: Struktur, die eine Transition enthält, die eine From-Aktivität benutzt, die nicht Workflow existiert
  - § FromActivityId (STRING): ID mit der auf nicht-existente Aktivität referenziert wird
  - § TransitionId (STRING): ID der Transition
- § MissingToActivities
  - § MissingToActivity: Struktur, die eine Transition enthält, die eine To-Aktivität benutzt, die nicht Workflow existiert
  - § ToActivityId (STRING): ID mit der auf nicht-existente Aktivität referenziert wird
  - § TransitionId (STRING): ID der Transition
- § RedundantTransitions

- § RedundantTransition: Struktur, die eine Transition enthält, die 'zuviel' ist, da diese bereits mit anderer ID existiert
  - § FromActivityId (STRING): ID der From-Aktivität
  - § FromActivityName (STRING): Name der From-Aktivität
  - § ToActivityId (STRING): ID der To-Aktivität
  - § ToActivityName (STRING): Name der To-Aktivität
  - § TransitionId (STRING): ID der Transition
- § InvalidFromLoopTransitions
  - § InvalidFromLoopTransition: Struktur, die eine Transition vom Typ 'FROM LOOP' enthält, wobei die From-Aktivität keine Schleifen-Aktivität ist
    - § FromActivityId (STRING): ID der From-Aktivität
    - § FromActivityName (STRING): Name der From-Aktivität
    - § TransitionId (STRING): ID der Transition
- § InvalidToLoopTransitions
  - § InvalidToLoopTransition: Struktur, die eine Transition vom Typ 'TO LOOP' enthält, wobei die To-Aktivität keine Schleifen-Aktivität ist
    - § ToActivityId (STRING): ID der To-Aktivität
    - § ToActivityName (STRING): Name der To-Aktivität
    - § TransitionId (STRING): ID der Transition
- § InvalidFromOrToLoops
  - § InvalidFromOrToLoop: Struktur enthält Schleifen-Aktivität deren Kombination aus FROM- und TO-LOOP-Transitionen unzulässig ist
    - § LoopActivityId (STRING): ID der Schleifen-Aktivität
    - § LoopActivityName: (STRING) Name der Schleifen-Aktivität
- § CycleActivities
  - § CycleActivity: Struktur, die eine Aktivität enthält, in denen in der Workflow-Graph-Struktur ein verbotener Zyklus zusammenläuft
    - § Id (STRING): ID der Aktivität
    - § Name (STRING): Name der Aktivität
- § InvalidLoopTransitions
  - § InvalidLoopTransition: Struktur, die eine Transition enthält, die aus einer Schleife in eine andere führt, oder anders ausgedrückt: die eine Aktivität enthält, die in mehreren Teilgraphen (Schleifen) liegt
    - § FromActivityId (STRING): ID der From-Aktivität
    - § FromActivityName (STRING): Name der From-Aktivität
    - § ToActivityId (STRING): ID der To-Aktivität
    - § ToActivityName (STRING): Name der To-Aktivität
    - § TransitionId (STRING): ID der Transition
- § InvalidToLoopNumActivities

- § InvalidToLoopNumActivity: Struktur, die eine Schleifen-Aktivität enthält, die nicht genau eine Transition vom Typ 'TO LOOP' hat
  - § Id (STRING): ID der Schleifen-Aktivität
  - § Name (STRING): Name der Schleifen-Aktivität
- § InvalidFromLoopNumActivities
  - § InvalidFromLoopNumActivity: Struktur, die eine Schleifen-Aktivität enthält, die nicht genau eine Transition vom Typ 'FROM LOOP' hat
    - § Id (STRING): ID der Schleifen-Aktivität
    - § Name (STRING): Name der Schleifen-Aktivität
- § MissingLoopConditionActivities
  - § MissingLoopConditionActivity: Struktur, die eine Schleifen-Aktivität enthält, für die keine Bedingung definiert ist
    - § Id (STRING): ID der Schleifen-Aktivität
    - § Name (STRING): Name der Schleifen-Aktivität
- § NoTerminationActivities
  - § NoTerminationActivity: Struktur, die eine Aktivität enthält von der aus die EndAktivität nicht erreicht werden kann
    - § Id (STRING): ID der Aktivität
    - § Name (STRING): Name der Aktivität
- § NoLoopTerminationActivities
  - § NoLoopTerminationActivity: Struktur, die eine Aktivität enthält die sich innerhalb einer Schleife befindet, von der aber kein Weg abgeht, der über eine TOLOOP Transition zurück zur Schleifenaktivität führt
    - § Id (STRING): ID der Aktivität
    - § Name (STRING): Name der Aktivität
- § MissingDefActIds
  - § MissingDefActId: Struktur, die eine Aktivität enthält, der keine Default-Aktivität zur Variablenübernahme zugeordnet wurde
    - § Id (STRING): ID der Aktivität
    - § Name (STRING): Name der Aktivität
- § AllClientTypesActivities
  - § AllClientTypesActivity: Struktur, die eine Aktivität enthält, für die die Client-Anwendungszuordnung nicht eindeutig ist (Es existiert eine Zuordnung für alle Clients)
    - § ActivityId (STRING): ID der Aktivität
    - § ActivityName (STRING): Name der Aktivität
    - § ApplicationId (STRING): ID der Anwendung
    - § ApplicationName (STRING): Name der Anwendung
- § AmbiguousActAppClientTypes

- § **AmbiguousActAppClientType**: Struktur, die eine Aktivität enthält, für die die Client-Anwendungszuordnung nicht eindeutig ist
  - § **ActivityId** (STRING): ID der Aktivität
  - § **ActivityName** (STRING): Name der Aktivität
  - § **ClientId** (STRING): ID des ClientType
  - § **ClientTypeName** (STRING): Name des ClientType
- § **ActAppInvalidClientTypes**
  - § **ActAppInvalidClientType**: Struktur, die eine Aktivität enthält, die eine Anwendungszuordnung für einen unzulässigen Client enthält
    - § **ActivityId** (STRING): ID der Aktivität
    - § **ActivityName** (STRING): Name der Aktivität
    - § **ApplicationId** (STRING): ID der Anwendung
    - § **ApplicationName** (STRING): Name der Anwendung
- § **ActEvtAllClientTypes**
  - § **ActEvtAllClientType**: Struktur, die eine Aktivität enthält, für die die Client-Event-Zuordnung nicht eindeutig ist (Es existiert eine Zuordnung für alle Clients)
    - § **ActivityId** (STRING): ID der Aktivität
    - § **ActivityName** (STRING): Name der Aktivität
    - § **EventTypeId** (STRING): ID des EventType
    - § **EventTypeName** (STRING): Name des EventType
- § **AmbiguousActEvtClientTypes**
  - § **AmbiguousActEvtClientType**: Struktur, die eine Aktivität enthält, für die die Client-Event-Zuordnung nicht eindeutig ist
    - § **ActivityId** (STRING): ID der Aktivität
    - § **ActivityName** (STRING): Name der Aktivität
    - § **EventTypeId** (STRING): ID des ClientType
    - § **EventTypeName** (STRING): Name des ClientType
    - § **ClientId** (STRING): ID des ClientType
    - § **ClientTypeName** (STRING): Name des ClientType
- § **AmbiguousGlobalEvtClientTypes**
  - § **AmbiguousGlobalEvtClientType**: Struktur, die einen Clienttyp enthält, für den die Zuordnung zum globalen Client-Event nicht eindeutig ist
    - § **ClientId** (STRING): ID des ClientType
    - § **ClientTypeName** (STRING): Name des ClientType
- § **AdhocActWithoutDefaultSubActs**
- § **AdhocActWithoutDefaultSubAct**: Struktur die Adhoc-Aktivitäten enthält, denen keine Default-Aktivität zu geordnet wurde
- § **Id** (STRING): ID der Adhoc-Aktivität
  - § **Name** (STRING): Name der Adhoc-Aktivität

## § AdhocActWithUnknownDefaultSubActs

§ AdhocActWithUnknownDefaultSubAct: Struktur, die Adhoc-Aktivitäten enthält, deren Default-Aktivität nicht im Modell existiert

§ Id (STRING): ID der Adhoc-Aktivität

§ Name (STRING): Name der Adhoc-Aktivität

## § AdhocActDefaultActIsNotAdhocSubActs

§ AdhocActDefaultActIsNotAdhocSubAct: Struktur, die Adhoc-Aktivitäten enthält, deren Default-Aktivität keine zulässige Lauflistenaktivität für die jeweilige Adhoc-Aktivität ist

§ Id (STRING): ID der Adhoc-Aktivität

§ Name (STRING): Name der Adhoc-Aktivität

## § AdhocActDefaultActIsNotWorkitems

§ AdhocActDefaultActIsNotWorkitem: Struktur, die Adhoc-Aktivitäten enthält, deren Default-Aktivität kein Arbeitsschritt ist

§ Id (STRING): ID der Adhoc-Aktivität

§ Name (STRING): Name der Adhoc-Aktivität

**Beispiel:****Aufbau von Warnings**

```
<Warnings>
<MissingWFParticipants>
<MissingWFParticipant Id="1234670003" Name="Dilbert" IsResponsible="1"/>
<MissingWFParticipant Id="1234679814" Name="Dogbert" IsResponsible="0"
IsFileResponsible="0">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step1"/>
<ReferencingActivity Id="" Name="step2"/>
</ReferencingActivities>
</MissingWFParticipant>
<MissingWFParticipant Id="813456780005" Name="Catbert" IsResponsible="1">
<ReferencingActivities>
<ReferencingActivity Id="" Name="step1"/>
</ReferencingActivities>
</MissingWFParticipant>
</MissingWFParticipants>
<WFParticipantsWithoutASUser>
<WFParticipantWithoutASUser Id="1234670003" Name="Dilbert"/>
<WFParticipantWithoutASUser Id="1234679814" Name="Dogbert">
</WFParticipantsWithoutASUser>
<InvalidResponsibleIds>
<InvalidResponsibleId Id="1234567..."/>
<InvalidResponsibleId Id="2345678..."/>
</InvalidResponsibleIds>
<UnconnectedActivities>
<UnconnectedActivity Id="12346798134567891345678900021" Name="step7"/>
<UnconnectedActivity Id="12346798134567891345678900021" Name="step9"/>
</UnconnectedActivities>
<InvalidDefActIds>
<InvalidDefActId Id="" Name="stepLoop3">
<ValidDefaultActivities>
<ValidDefaultActivity Id="" Name="bla"/>
</ValidDefaultActivities>
</InvalidDefActId>
</InvalidDefActIds>
<NoOutVarsInLoopActConditions>
```

```

<NoOutVarsInLoopActCondition Id="12346798134545678900021" Name="loopX"/>
<NoOutVarsInLoopActCondition Id="12346798134567891345671" Name="loopY"/>
</NoOutVarsInLoopActConditions>
<NoOutVarsInLoopActs>
<NoOutVarsInLoopAct Id="12346798134567891345678900021" Name="loopXA"/>
<NoOutVarsInLoopAct Id="12346798134567891345678900021" Name="loopYB"/>
</NoOutVarsInLoopActs>
<UnknownActAppClientTypes>
<UnknownActAppClientType ActivityId="" ActivityName="AAA"
ClientTypeId="1234679813456789134567890000A"/>
<UnknownActAppClientType ActivityId="" ActivityName="BBB"
ClientTypeId="1234679813456789134567890000B"/>
</UnknownActAppClientTypes>
<UnknownActEvtClientTypes>
<UnknownActEvtClientType ActivityId="" ActivityName="ABCDE" EventTypeId="100"
EventTypeName="Before123Event" ClientTypeId="ABC.." />
<UnknownActEvtClientType ActivityId="" ActivityName="DEFGH" EventTypeId="100"
EventTypeName="After456Event" ClientTypeId="XYZ.." />
</UnknownActEvtClientTypes>
<UnknownGlobalEvtClientTypes>
<UnknownGlobalEvtClientType ClientTypeId="123456..." />
<UnknownGlobalEvtClientType ClientTypeId="1234567..." />
</UnknownGlobalEvtClientTypes>
<AppToolsWithoutClientType>
<AppToolWithoutClientType ActivityId="" ActivityName="Aktivität 42"
ApplicationId="" ApplicationName="Applikation 43"/>
<AppToolWithoutClientType ActivityId="" ActivityName="Aktivität 44"
ApplicationId="" ApplicationName="Applikation 45"/>
<AppToolWithoutClientType ActivityId="" ActivityName="Aktivität 46"
ApplicationId="" ApplicationName="Applikation 47"/>
</AppToolsWithoutClientType>
<NoOrInvalidFileResponsibleId/>
<MasksWithoutFields>
<MaskWithoutFields MaskId="1234567" MaskName="Maske1"/>
<MasksWithoutFields/>
</Warnings>

```

## Hinweis:

### Genauere Beschreibung von Warnings

#### § MissingWFParticipants

- § MissingWFParticipant: Struktur, die einen Teilnehmer des Workflows enthält, der nicht in der Organisation existiert
  - § Id (STRING): ID des Teilnehmers
  - § Name (STRING): Name des Teilnehmers
  - § IsResponsible (INT): gibt an, ob der Teilnehmer Prozess-Verantwortlicher ist (ja = 1, nein = 0)
  - § IsFileResponsible (INT): gibt an, ob der Teilnehmer Akten-Verantwortlicher ist (ja = 1, nein = 0)

#### § ReferencingActivities

- § ReferencingActivity: Struktur, die eine Aktivität enthält, der Benutzer zugeordnet sind, die nicht in der Organisation existieren
  - § Id (STRING): ID der Aktivität
  - § Name (STRING): Name der Aktivität

#### § WFParticipantsWithoutASUser

- § WFParticipantWithoutASUser: Struktur, die einen Benutzer enthält, der keinem oder einem nicht existierenden AS-Benutzer zugeordnet ist
  - § Id (STRING): ID des Benutzers
  - § Name (STRING): Name des Benutzers
- § InvalidResponsibleIds
  - § InvalidResponsibleId: Struktur, die einen Prozessverantwortlichen des Workflows enthält, der nicht in der Teilnehmerliste existiert
    - § Id (STRING): ID des Prozessverantwortlichen
- § UnconnectedActivities
  - § UnconnectedActivity: Struktur, die eine Aktivität enthält, die ausgehend von der StartActivity nicht erreicht werden kann
    - § Id (STRING): ID der Aktivität
    - § Name (STRING): Name der Aktivität
- § InvalidDefActIds: Struktur, die Aktivitäten enthält, denen eine ungültige Default-Aktivität zur Variablenübernahme zugeordnet wurde
  - § Id (STRING): ID der Aktivität
  - § Name (STRING): Name der Aktivität
- § ValidDefaultActivities: Struktur die Aktivitäten enthält, die gültige Default-Aktivitäten für die übergeordnete InvalidDefActId darstellen
  - § Id (STRING): ID der Aktivität
  - § Name (STRING): Name der Aktivität
- § NoOutVarsInLoopActConditions: Struktur mit Schleifenaktivitäten, für die keine Variablenübernahme definiert wurde
  - § Id (STRING): ID der Aktivität
  - § Name (STRING): Name der Aktivität
- § UnknownActAppClientTypes: Struktur mit Aktivitäten, für die eine Anwendungszuordnung für einen unbekannten Client existiert
  - § ActivityId (STRING): ID der Aktivität
  - § ActivityName (STRING): Name der Aktivität
  - § ClientTypeId (STRING): ID des ClientTyps
- § UnknownActEvtClientTypes: Struktur mit Aktivitäten, für die ein Event-zuordnung für einen unbekannten Client existiert
  - § ActivityId (STRING): ID der Aktivität
  - § ActivityName (STRING): Name der Aktivität
  - § EventTypeId (STRING): ID des EventTyps
  - § EventTypeName (STRING): Name des EventTyps
  - § ClientTypeId (STRING): ID des ClientTyps
- § UnknownGlobalEvtClientTypes: Struktur mit unbekannten Clienttypen, die für einglobales Client-Event angelegt wurde

- § ClientTypeId (STRING): ID des ClientTyps
- § AppToolsWithoutClientType: Struktur mit Aktivitäten, für die ein Anwendungszuordnung ohne Clienttyp-Angabe existiert
  - § ActivityId (STRING): ID der Aktivität
  - § ActivityName (STRING): Name der Aktivität
  - § ApplicationId (STRING): ID der Anwendung
  - § ApplicationName (STRING): Name der Anwendung
- § MasksWithoutFields: Struktur mit Masken, für die keine Maskenfelder existieren
  - § MaskId (STRING): ID der Maske
  - § MaskName (STRING): Name der Maske

## Workflowprozess und Arbeitsschritt

- § [wfm.CancelWorkItem](#)
- § [wfm.CompleteWorkItem](#)
- § [wfm.CreateProcessInstance](#)
- § [wfm.GetActivityPerformers](#)
- § [wfm.GetProcessList](#)
- § [wfm.GetProcessListByObject](#)
- § [wfm.GetProcessProtocol](#)
- § [wfm.GetProcessResponsibles](#)
- § [wfm.GetRunningActivities](#)
- § [wfm.GetWorkItem](#)
- § [wfm.GetWorkItemList](#)
- § [wfm.GetWorkItemParams](#)
- § [wfm.SetActivityPerformers](#)
- § [wfm.SetProcessResponsibles](#)
- § [wfm.StartProcess](#)
- § [wfm.StartWorkItem](#)

### wfm.CancelWorkItem

#### **Beschreibung:**

Dieser Job entpersonalisiert einen Arbeitsschritt. Danach ist der Arbeitsschritt wieder für alle Teilnehmer sichtbar und kann erneut personalisiert werden.

#### **Parameter:**

- UserId (STRING): ID des bisherigen Benutzers
- WorkItemId (STRING): Instanz-ID der Aktivität
- OrganisationId (STRING): ID der Organisation
- ClientTypeId (String): ID des verwendeten Clienttyps



**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**wfm.CompleteWorkItem****Beschreibung:**

Dieser Job überträgt die Bearbeitungsdaten (Variablen, Akte) eines Arbeitsschritts an den Server und leitet den Arbeitsschritt in Abhängigkeit des Parameters 'ActionType' weiter.

**Parameter:**

UserId (STRING): ID des Benutzers

WorkItemId (STRING): Instanz der Aktivität

Parameters (BASE64): Liste der Workflowvariablen im XML-Format

ActionType (STRING): Flag, das angibt, was mit der Aktivität passieren soll

§ SEND\_BUTTON: Arbeitsschritt weiterleiten

§ STOREONLY: Änderungen werden gespeichert, aber Arbeitsschritt wird nicht weitergeleitet

SendTo (STRING): wird nicht mehr unterstützt -> Leerstring übergeben

File (BASE64): beinhaltet Dokumente der Workflowakte im XML-Format

DocsDeleted (STRING): kommaseparierte Liste von Dokumentenids, die aus der WF-Akte gelöscht werden sollen

ClientTypeId (String): ID des verwendeten Clienttyps

RoutingList (BASE64): Laufliste. Dieser Parameter ist optional.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Beispiel:****Aufbau von Parameters**

```
<Parameters>
<Parameter Name="WF_BEARBEITER_1" DataField=
"9FC5D03089E843F7B2D64F1CC2421418"><![CDATA[Schulze]]></Parameter>
</Parameters>
```

**Hinweis:****Genauere Beschreibung von Parameters****§ Parameter: Workflowvariable**

§ DataField (STRING): ID des Parameters

§ Name (STRING): Name des Parameters

§ CDATA: Daten, die den Inhalt des Parameters angeben

**Beispiel:****Aufbau von File**

```
<File>
<Docs>
  <Doc Id="45" Type="23" Location="1" Workspace="0" New="1" Deleteable="1"
```

```
Moveable="2" UseActiveVariant="1"/>
</Docs>
</File>
```

**Hinweis:****Genauere Beschreibung von File****§ Docs:** Liste von Parametern (Doc) mit der folgenden Struktur

- § **Id (STRING):** ID des Dokuments
- § **Type (LONG):** Typ des Dokuments
- § **Location (INT):** gibt an, ob das Dokument in der SDREL(Location='1', SDREL ist die Datenbanktabelle Stamm-Dokumenten-Relation) oder Systemablage(Location='2') liegt
- § **Workspace (INT):** gibt an, ob sich das Objekt im Infobereich (0) oder im Arbeitsbereich (1) der befindet
- § **New (INT):** gibt an, ob das Objekt neu der Akte eingefügt wurde ( New ='1')
- § **Deleteable (INT):** gibt an, ob das Dokument aus der Akte gelöscht werden darf (0 = nein, 1= ja)
- § **Moveable (INT):** gibt an, ob das Dokument in der Akte verschoben werden darf (0 = nein, 1= ja)
- § **UseActiveVariant (INT):** gibt an, ob für diese Dokument die aktive Variante verwendet werden soll (0 = nein, 1= ja)

**Beispiel:****Aufbau von RoutingList**

```
<RoutingList Id="3294B433BFF6454D9C861B86B5A8AD5D"
ProcessId="BA16C21BB96D46D099E72070BCB644CC"
ActivityId="3294B433BFF6454D9C861B86B5A8AD5D" Expandable="1">
<Entries>
<Entry Nr="203" Expandable="1">
<Item Id="99825B18A8334987935684FDA3D6A40D"
ActivityId="6EE4490A48164A0FA6DC34A80099AF66" ActivityName="Rechnung erstellen"
ModelActivityName="Rechnung erstellen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectsIds>
</Item>
</Entry>
<Entry Nr="253" Expandable="1">
<Item Id="E15594D692C14FDA9AFDE8FA0A43F6E4"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen BL"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectsIds>
</Item>
<Item Id="C6DA9503CD874D69A9B703D0E06A52E8"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen GF"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectsIds>
</Item>
</Entry>
</Entries>
</RoutingList>
```

**Hinweis:****Genauere Beschreibung von RoutingList**

- § RoutingList: Laufliste mit folgender Struktur (oder Untermengen davon)
- § Id (String): ID der Laufliste. Dieser Wert wird vom Server gesetzt und darf nicht verändert werden.
- § ProcessId (String): Prozessid
- § ActivityId (String): Aktivitätsid
- § Expandable (Int): 0: Laufliste kann nicht erweitert werden, 1: Laufliste kann erweitert werden.
  - § Entries: Die Struktur fasst Einträge der Laufliste zusammen. Ein Eintrag besteht aus mehreren Elementen, die parallel ausgeführt werden können.
  - § Entry: Beschreibt einen Eintrag in der Laufliste.
  - § Nr (Int): Dient der relativen Sortierung der Einträge innerhalb der Laufliste. Die absoluten Werte haben für den Client keine Bedeutung.
  - § Expandable (Int): 0: Eintrag kann nicht erweitert werden, 1: Eintrag kann erweitert werden.
  - § Item: Beschreibt ein Element der Laufliste. Hierbei handelt es sich um eine Aktivität, eine ausführende Person und ggfs einen Termin.
  - § Id (STRING): Dient der Identifizierung. Diese ID darf nicht verändert werden und muss bei allen Jobs identisch mitgeschickt werden. Wurde ein Item durch den Client erstellt, muss dieser hier eine ID angeben.
  - § ActivityId (String): ID der Aktivität im Workflowmodell
  - § ActivityName (String): Name der Aktivität (muss nicht unbedingt mit dem Namen im Workflowmodell übereinstimmen).
  - § ActivityModelName (String): Name der Aktivität im Workflowmodell
  - § TimerId(String): ID einer Mahnfrist
  - § TimerDuration(Int): Dauer der Frist
  - § TimerDurationType(Int): 0: keine Frist, 1: relativ, 2: absolut
  - § Changeable(Int): 0: keine Änderung möglich, 1: Das Element darf vom Client verändert werden.
  - § Deleteable(Int): 0: Löschen nicht erlaubt, 1: Element darf gelöscht werden
  - § Remark (String): Hinweis zur Bearbeitung (Text)
  - § ObjectsIds (String): Liste von GUIDS der Bearbeiter (Rollen oder Personen), durch Komma getrennt

## wfm.CreateProcessInstance

### Beschreibung:

Dieser Job erzeugt einen Prozess der angegebenen Workflow-Familie, wenn der angegebene Benutzer das Recht auf Ausführung des Workflows besitzt. Die erzeugte Prozess-Instanz kann dann mit Hilfe des Jobs [wfm.StartProcess](#) gestartet werden.

### Parameter:

UserId (STRING): ID des Benutzers

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID der Workflow-Familie

ClientTypeId (STRING): ID des verwendeten Clienttyps

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

ProcessId (STRING): ID des neu erzeugten Prozesses

**Siehe auch:**

[wfm.GetWorkflowList](#) , [wfm.StartProcess](#)

## wfm.GetActivityPerformers

**Beschreibung:**

Dieser Job liefert aller Benutzer/Rollen, die als Teilnehmer der angegebenen Aktivität zugeordnet sind.

**Parameter:**

RActivityId (STRING): Instanz-ID der Aktivität

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

ObjectIds (BASE64): Liste der Rollen/Benutzer im XML-Format

OrganisationId (STRING): ID der Organisation, zu der die Rollen/Benutzer gehören

**Beispiel:**

Aufbau der ObjectIds

```
<Objects>
<Object Id="E95EF24F6C6AE1A40F" Absent="0" Substitute="0" Flag="1"/>
<Object Id="8FDD6BCB06CE478699" Absent="0" Substitute="0" Flag="1"/>
<Object Id="A7286EA0463F382057" Absent="0" Substitute="0" Flag="1"/>
</Objects>
```

**Hinweis:**

Genauere Beschreibung von ObjectIds

Object: Struktur mit folgenden Inhalten

§ Id (STRING): ID des Objektes

§ Absent (LONG): Flag gibt an, ob die Person abwesend = 1 oder anwesend = 0 ist

§ Substitute (LONG): Flag gibt an, ob die Person die Aktivität in Stellvertretung = 1 sieht

§ Flag (LONG):

§ 1 = Benutzer ist dem Arbeitsschritt direkt (nicht über Rolle) zugeordnet

§ 2 = Benutzer sieht den Arbeitsschritt in seinem Eingangskorb

§ 4 = Arbeitsschritt ist von diesem Benutzer personalisiert

**Siehe auch:**

[wfm.SetActivityPerformers](#)

## wfm.GetProcessFile

### Beschreibung:

Dieser Job liefert die Akte zu einem Prozess.

### Parameter:

ProcessId (String): Prozessid

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

File (BASE64): Die Akte zum Prozess im XML-Format

### Beispiel:

Aufbau von File

```
<File>
<Docs>
  <Doc Id="45" Type="23" Location="1" Workspace="0" New="1" Deleteable="1"
Moveable="2" UseActiveVariant="0" OriginalId="42" Display="1" />
</Docs>
</File>
```

### Hinweis:

Genauere Beschreibung von Workspace

§ Docs: Liste von Parametern (Doc) mit der folgenden Struktur

- § Id (STRING): ID des Dokuments
- § Type (LONG): Typ des Dokuments
- § Location (INT): gibt an, ob das Dokument in der SDREL(Location='1', SDREL ist die Datenbanktabelle Stamm-Dokumenten-Relation) oder Systemablage(Location='2') liegt
- § Workspace (INT): gibt an, ob sich das Objekt im Infobereich (0) oder im Arbeitsbereich (1) der befindet
- § New (INT): gibt an, ob das Objekt neu der Akte eingefügt wurde ( New ='1')
- § Deleteable (INT): gibt an, ob das Dokument aus der Akte gelöscht werden darf (0 = nein, 1= ja)
- § Moveable (INT): gibt an, ob das Dokument in der Akte verschoben werden darf (0 = nein, 1= ja)
- § UseActiveVariant (INT): gibt an, ob für das Objekt die aktive Variante verwendet werden soll (0 = nein, 1= ja)
- § OriginalId (INT): gibt an, welches Dokument ursprünglich in die Akte gezogen wurde (welche Id dieses Dokument hatte)
- § Display (INT): gibt an, ob dieses Dokument in der Vorschau angezeigt werden soll (0 = nein, 1= ja)

## wfm.GetProcessList

### Beschreibung:

Dieser Job liefert eine Liste von Prozessen, wobei der Status spezifiziert werden kann.

**Parameter:**

Flags (INT): zusammengesetztes Flag, gibt die möglichen Status der abzufragenden Prozesse an

- § 1 = CSPROCESS\_INIT
- § 2 = CSPROCESS\_RUNNING
- § 4 = CSPROCESS\_SUSPENDED
- § 8 = CSPROCESS\_ACTIVE
- § 16 = CSPROCESS\_TERMINATED
- § 32 = CSPROCESS\_COMPLETED

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (BASE64): Informationen zu den angeforderten Prozessen im XML-Format

**Beispiel:**

Aufbau von Processes

```
<Processes>
<Process>
<Id>9E813BD6D4054B6ABD9385A804FEA398</Id>
<Name>test 231</Name>
<Subject>Test (€42)</Subject>
<State>2</State>
<CreatorId>8FDD6BCB06CE467FAE8885E81F078699</CreatorId>
<CreationTime>1077888972</CreationTime>
</Process>
</Processes>
```

**Hinweis:**

Genauere Beschreibung von Process

- § Id (STRING): ID des Prozesses
- § Name (STRING): Name des Prozesses
- § State (INT): Status des Prozesses
- § CreatorId (STRING): ID des Erstellers des Prozesses
- § CreationTime (INT): Erstellzeitpunkt des Prozesses

**wfm.GetProcessListByObject****Beschreibung:**

Dieser Job liefert eine Liste von Prozessen die ein bestimmtes Objekt in der Akte haben.

**Parameter:**

- § OrganisationId (STRING): ID der Organisation
- § ObjectId (INT): ID des Objekts
- § UserId (STRING, optional): Id der Benutzers aus der Organisationsstruktur
- § AllProcesses (INT, optional, default =0): Wenn 1, dann werden auch beendete Prozesse ermittelt. Andernfalls werden nur laufende Processes zurückgeliefert.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (BASE64): Informationen zu den angeforderten Prozessen im XML-Format

**Beispiel:****Aufbau von Processes**

```
<Processes>
<Process
Id ="9E813BD6D4054B6ABD9385A804FEA398"
Name="test 231"
Subject ="Test (€42)
State="2"
CreatorId="8FDD6BCB06CE467FAE8885E81F078699"
CreationTime="1077888972"
      ProcessResponsible="0"
</Process>
</Processes>
```

**Hinweis:****Genauere Beschreibung von Process**

- § Id (STRING): ID des Prozesses
- § Name (STRING): Name des Prozesses
- § State (INT): Status des Prozesses
- § CreatorId (STRING): ID des Erstellers des Prozesses
- § CreationTime (INT): Erstellzeitpunkt des Prozesses
- § ProcessResponsible (INT): 1 = Der anfragende Benutzer ist Prozessverantwortlicher für diesen Prozess; 0 = sonst

## wfm.GetProcessProtocol

**Beschreibung:**

Dieser Job liefert die Protokoll-Datei zu einem Prozess.

**Parameter:**

ProcessId (STRING): ID des Prozesses

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Dateiliste: Name und Pfad der Protokoll-Datei

## wfm.GetProcessResponsibles

**Beschreibung:**

Dieser Job liefert die Verantwortlichen für den angegebenen Prozess.

**Parameter:**

ProcessId (STRING): ID des Prozesses

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

ProcessId (STRING): ID der Organisation des Prozesses

Responsibles (STRING): kommaseparierte ID-Liste der Prozessverantwortlichen

**Siehe auch:**

[wfm.GetProcessResponsibles](#)

## wfm.GetRunningActivities

**Beschreibung:**

Dieser Job liefert alle zu bearbeitenden Aktivitäten für einen bestimmten Benutzer.

**Parameter:**

OrganisationId (STRING): ID der Organisation

UserId (STRING): ID des enaio®-Benutzers

ClientTypeId (String): ID des verwendeten Clienttyps

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

RunningActivities (BASE64): Liste mit Daten aller laufenden Aktivitäten des Benutzers im XML-Format

**Beispiel:****Aufbau von RunningActivities**

```
<RunningActivities>
<RunningActivity>
<Activity Id="" RActivityId="" Name="" State="" ClosureTime="" OverTime=""
ReminderTime="" CanCancel =""/>
<User Name=""/>
<Process Id="" Name="" WorkflowId="" CreationTime="" IconId="" ObjectId=""/>
<Columns>
<Column DisplayName="" Value="" Position="">
<Column DisplayName="" Value="" Position="">
</Columns>
</RunningActivity>
<RunningActivity>
<Activity Id="" RActivityId="" Name="" State="" ClosureTime="" OverTime=""
ReminderTime="" CanCancel =""/>
<User Name=""/>
<Process Id="" Name="" WorkflowId="" Subject="" CreationTime="" IconId=""/>
<Columns/>
</RunningActivity>
</RunningActivities>
```

**Hinweis:**

Genauere Beschreibung des Rückgabewertes RunningActivities



- § Activity: Struktur beschreibt eine laufende Aktivität
  - § Id (STRING): ID der Aktivität im Modell
  - § RActivityId (STRING): Instanz-ID der Aktivität
  - § Name (STRING): Name der Aktivität
  - § State (INT): Status der Aktivität
  - § ClosureTime (INT): Sperrfrist gibt an, wie lange die Aktivität noch gesperrt ist
  - § OverTime (INT): Flag, das angibt, ob die Aktivität schon hätte erledigt (1) werden müssen
  - § ReminderTime (INT): Mahnfrist gibt an, bis wann die Aktivität erledigt sein soll
  - § CanCancel (INT): nich mehr unterstützt -> immer 0
- § User:
  - § Name (STRING): Name des personalisierenden Users
- § Process: beschreibt den zugehörigen Prozess
  - § Id (STRING): ID des Prozesses
  - § Name (STRING): Name des Prozesses
  - § WorkflowId (STRING): ID des Workflows
  - § Subject (STRING): Betreff des Prozesses
  - § CreationTime (INT): Erstellzeitpunkt des Prozesses
  - § IconId (INT): Iconid des Workflowmodells
  - § ObjectId(String): Id des Dokuments, das von den Clients in der Vorschau angezeigt werden soll.
- § Columns: Liste von Elementen des Typs 'Column'
  - § Column: Dient zur Anzeige von Workflowvariablen
  - § DisplayName (STRING): Unter diesem Namen soll die Variable angezeigt werden
  - § Value: Wert der Variablen
  - § Position (INT): bestimmt die Reihenfolge der Elemente

## wfm.GetWorkItemList

### Beschreibung:

Dieser Job liefert eine Liste aller Arbeitsschritte für die der angebene Benutzer als Teilnehmer konfiguriert ist und der Arbeitsschritt nicht schon von einem anderen Teilnehmer personalisiert ist.

### Parameter:

OrganisationId (STRING): ID der Organisation

UserId (STRING): ID des Benutzers

ClientTypeId (String): ID des verwendeten Clienttyps

Flags (INT): Mit Flags können angefragten Arbeitsschritte eingegrenzt werden.

§ 1 = INITIATED

§ 2 = RUNNING

§ 4 = SUSPENDED

§ 16 = TERMINATED

§ 32 = COMPLETED

§ 64 = INUSE

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

WorkItems (BASE64): Liste mit angeforderten Arbeitsschritten im XML-Format

### Beispiel:

#### Aufbau von WorkItems

```
<WorkItems>
<WorkItem Id="" State="" Personalized="" ProcessId="" ProcessName=""
ActivityId="" ActivityName="" WarningTime="" OverTime=""
CreationTime="" WorkflowId="" Substitute="" IconId="" WorkflowType="2"
WorkflowVersion="5" ObjecId="32">
<Columns>
<Column DisplayName="" Value="" Position="">
<Column DisplayName="" Value="" Position="">
</Columns>
</WorkItem>
<WorkItem Id="" State="" Personalized="" ProcessId="" ProcessName=""
ProcessSubject="" ActivityId="" ActivityName="" WarningTime=""
OverTime="" CreationTime="" WorkflowId="" Substitute="" IconId="" WorkflowType="1"
WorkflowVersion="42" ObjecId="52">
<Columns>
<Column DisplayName="" Value="" Position="">
<Column DisplayName="" Value="" Position="">
</Columns>
</WorkItem>
</WorkItems>
```

### Hinweis:

#### Genauere Beschreibung von WorkItems

§ WorkItem: beschreibt einen Arbeitsschritt

§ Id (STRING): ID des Arbeitsschritts

§ State (INT): Status des Arbeitsschritts

§ 1 = INITIATED

§ 2 = RUNNING

§ 4 = SUSPENDED

§ 16 = TERMINATED

§ 32 = COMPLETED

§ 64 = INUSE

§ Personalized (STRING): Name der Nutzers, der diesen Schritt personalisiert hat

§ ProcessId (STRING): Prozessid

§ ProcessName (STRING): Prozessname

§ ProcessSubject (STRING): Prozesssubject

- § ActivityId (STRING): Instanz-ID der Aktivität
- § ActivityName (STRING): Name der Aktivität
- § WarningTime (INT): Mahnfrist
- § OverTime (INT): Flag, das angibt, ob der Schritt schon hätte erledigt (1) werden müssen
- § CreationTime (INT): Erstellungszeit der Aktivität
- § WorkflowId (STRING): Workflowid
- § IconId (INT): Iconid des Workflowmodells
- § Substitute (INT): 1 = Benutzer bekommt den Arbeitsschritt als Stellvertreter, ansonsten 0
- § WorkflowType (INT): 1 = ProductionWorkflow, 2 = Adhoc Workflow
- § WorkflowVersion (INT): liefert die Versionsnummer des Workflowmodells
- § ObjectId (String): Die Id des Dokuments, das von den Clients in der Vorschau angezeigt werden soll.
- § Columns: Liste von Elementen des Typs 'Column'
  - § Column: Dient zur Anzeige von Workflowvariablen
  - § DisplayName (STRING): Unter diesem Namen soll die Variable angezeigt werden
  - § Value: Wert der Variablen
  - § Position (INT): bestimmt die Reihenfolge der Elemente

**Siehe auch:**

[wfm.GetWorkItem](#) , [wfm.StartWorkItem](#) , [wfm.GetWorkItemParams](#)

## wfm.GetWorkItemParams

### Beschreibung:

Dieser Job ermittelt alle Parameter eines Arbeitsschritts für den Benutzer, der den Arbeitsschritt personalisiert hat. Es werden alle Workflowvariablen, Parameter zur Eingabemaske, Inhalt der Workflowakte und zusätzliche Parameter (z. B. muss zum Weiterleiten eine Passwordeingabe erfolgen) zurückgeliefert. Dieser Job wurde durch [wfm.GetWorkItem](#) ersetzt.

### Parameter:

WorkItemId (STRING): Instanz-ID der Aktivität

UserId (STRING): ID des Benutzers

ClientTypeId (String): ID des verwendeten Clienttyps

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Parameters (BASE64): Liste mit Parametern zur Datenmaske im XML-Format

ExtendedAttributes (BASE64): Liste mit Daten zu den Parametern ('Attribute') im XML-Format

File (BASE64): Liste mit Dokumenten des Info-/Arbeitsbereiches der WF-Akte im XML-Format

RoutingList (Base64): Laufliste. Dieser Parameter ist optional.

### Beispiel:

## Aufbau von Parameters

```
<Parameters>
<Parameter FormField="" DataField="" Name="" Mode="" Selection=""
InfoText="" ListType="" ><![CDATA[ ]]></Parameter>
<Parameter FormField="" DataField="" Name="" Mode="" Selection=""
InfoText="" ListType="" ><![CDATA[ ]]></Parameter>
</Parameters>
```

### Hinweis:

#### Genauere Beschreibung von Parameters

- § Parameters: Liste der formalen Parameter mit folgender Struktur (oder Untermengen davon)
  - § FormField (STRING): ID des Feldes auf einem Formularblatt dem die Workflowvariable zugeordnet ist, wenn keine Zurordnung besteht Name der Workflowvariable
  - § DataField (STRING): ID der Workflowvariable
  - § Name (STRING): Name der Workflowvariable
  - § Mode (INT): Modus der Workflowvariable
    - § 1 = EingabeParameter
    - § 2 = AusgabeParameter
    - § 3 = Ein/Ausgabeparameter
  - § Selection (STRING): Auswahltyp bei Workflowvariablen in Listenform (single oder multi:x)
  - § InfoText (STRING): Informationstext Infotext bei Workflowvariablen in Listenform
  - § ListType (STRING): Typ der Liste
    - § ProcessList
    - § UserList
    - § UserDefList
  - § CDATA: Aufbau und Daten der Workflowvariable

### Beispiel:

#### Aufbau von ExtendedAttributes

```
<ExtendedAttributes>
<ExtendedAttribute Name="MASKID" Value="" />
<ExtendedAttribute Name="SEND_BUTTON" Value="0" />
<ExtendedAttribute Name="SENDTO_BUTTON" Value="0" />
<ExtendedAttribute Name="END_BUTTON" Value="0" />
<ExtendedAttribute Name="SIGN_ACTIVITY" Value="" />
<ExtendedAttribute Name="CHECK_PASSWORD" Value="" />
</ExtendedAttributes>
```

### Hinweis:

#### Genauere Beschreibung von ExtendedAttributes

- § ExtendedAttributes: Liste von Parametern ('Attribute') mit folgender Struktur
  - § Name (STRING): Name des Attributes
    - § MASKID: GUID der Workflowmaske
    - § SEND\_BUTTON:
    - § END\_BUTTON:

- § SIGN\_ACTIVITY: 1 = Digitale Signatur erforderlich, ansonsten 0
- § CHECK\_PASSWORD: 1 = zum Weiterleiten muss Passwort eingegeben werden, ansonsten 0
- § Value: Wert des Attributes

### Beispiel:

#### Aufbau von RoutingList

```
<RoutingList Id="3294B433BFF6454D9C861B86B5A8AD5D"
ProcessId="BA16C21BB96D46D099E72070BCB644CC"
ActivityId="3294B433BFF6454D9C861B86B5A8AD5D" Expandable="1">
<Entries>
<Entry Nr="203" Expandable="1">
<Item Id="99825B18A8334987935684FDA3D6A40D"
ActivityId="6EE4490A48164A0FA6DC34A80099AF66" ActivityName="Rechnung erstellen"
ModelActivityName="Rechnung erstellen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
</Entry>
<Entry Nr="253" Expandable="1">
<Item Id="E15594D692C14FDA9AFDE8FA0A43F6E4"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen BL"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
<Item Id="C6DA9503CD874D69A9B703D0E06A52E8"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen GF"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
</Entry>
</Entries>
</RoutingList>
```

- § RoutingList: Laufliste mit folgender Struktur (oder Untermengen davon)
- § Id (String): ID der Laufliste. Dieser Wert wird vom Server gesetzt und darf nicht verändert werden.
- § ProcessId (String): Prozessid
- § ActivityId (String): Aktivitätsid
- § Expandable (Int): 0: Laufliste kann nicht erweitert werden, 1: Laufliste kann erweitert werden.
  - § Entries: Die Struktur fasst Einträge der Laufliste zusammen. Ein Eintrag besteht aus mehreren Elementen, die parallel ausgeführt werden können.
  - § Entry: Beschreibt einen Eintrag in der Laufliste.
  - § Nr (Int): Dient der relativen Sortierung der Einträge innerhalb der Laufliste. Die absoluten Werte haben für den Client keine Bedeutung.
  - § Expandable (Int): 0: Eintrag kann nicht erweitert werden, 1: Eintrag kann erweitert werden.
  - § Item: Beschreibt ein Element der Laufliste. Hierbei handelt es sich um eine Aktivität, eine ausführende Person und ggfs einen Termin.
  - § Id (STRING): Dient der Identifizierung. Diese ID darf nicht verändert werden und muss bei allen Jobs identisch mitgeschickt werden. Wurde ein Item durch den Client erstellt, muss dieser hier eine ID angeben.

- § ActivityId (String): ID der Aktivität im Workflowmodell
- § ActivityName (String): Name der Aktivität (muss nicht unbedingt mit dem Namen im Workflowmodell übereinstimmen).
- § ActivityModelName (String): Name der Aktivität im Workflowmodell
- § TimerId(String): ID einer Mahnfrist
- § TimerDuration(Int): Dauer der Frist
- § TimerDurationType(Int): 0: keine Frist, 1: relativ, 2: absolut
- § Changeable(Int): 0: keine Änderung möglich, 1: Das Element darf vom Client verändert werden.
- § Deleteable(Int): 0: Löschen nicht erlaubt, 1: Element darf gelöscht werden
- § Remark (String): Hinweis zur Bearbeitung (Text)
- § ObjectsIds (String): Liste von GUIDS der Bearbeiter (Rollen oder Personen), durch Komma getrennt

### Beispiel:

#### Aufbau von File

```
<File>
<Docs>
  <Doc Id="45" Type="23" Location="1" Workspace="0" New="1" Deleteable="1"
  Moveable="2" UseActiveVariant="0" OriginalId="42" Display="1" />
</Docs>
</File>
```

### Hinweis:

#### Genauere Beschreibung von Workspace

- § Docs: Liste von Parametern (Doc) mit der folgenden Struktur
  - § Id (STRING): ID des Dokuments
  - § Type (LONG): Typ des Dokuments
  - § Location (INT): gibt an, ob das Dokument in der SDREL(Location='1', SDREL ist die Datenbanktabelle Stamm-Dokumenten-Relation) oder Systemablage(Location='2') liegt
  - § Workspace (INT): gibt an, ob sich das Objekt im Infobereich (0) oder im Arbeitsbereich (1) der befindet
  - § New (INT): gibt an, ob das Objekt neu der Akte eingefügt wurde ( New ='1')
  - § Deleteable (INT): gibt an, ob das Dokument aus der Akte gelöscht werden darf (0 = nein, 1= ja)
  - § Moveable (INT): gibt an, ob das Dokument in der Akte verschoben werden darf (0 = nein, 1= ja)
  - § UseActiveVariant (INT): gibt an, ob für das Objekt die aktive Variante verwendet werden soll (0 = nein, 1= ja)
  - § OriginalId (INT): gibt an, welches Dokument ursprünglich in die Akte gezogen wurde (welche Id dieses Dokument hatte)
  - § Display (INT): gibt an, ob dieses Dokument in der Vorschau angezeigt werden soll (0 = nein, 1= ja)
  - §

## wfm.SetActivityPerformers

### Beschreibung:

Dieser Job setzt die Teilnehmer einer beliebigen Aktivität. Teilnehmer können Benutzer oder Rollen sein. Es ist zu beachten, dass alte Einstellungen überschrieben werden.

### Parameter:

OrganisationId (STRING): ID der Organisation, zu der die Rollen/Benutzer gehören

UserId (STRING): ID des ausführenden Benutzers

RActivityId (STRING): Instanz-ID der Aktivität

ObjectIds (STRING): GUIDs der Rollen/Benutzer, durch Komma getrennt

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Siehe auch:

[wfm.GetActivityPerformers](#)

## wfm.SetProcessResponsibles

### Beschreibung:

Dieser Job setzt die Verantwortlichen eines Prozesses.

### Parameter:

ProcessId (STRING): ID des Prozesses

OrganisationId (STRING): ID der Organisation

Responsibles (STRING): komma-getrennte Liste mit Ids der Prozessverantwortlichen

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.StartProcess

### Beschreibung:

Dieser Job startet einen Workflow-Prozess. Es wird überprüft, ob der angegebene Benutzer berechtigt ist den Prozess zu starten. Die Startaktivität des Prozesses wird ausgeführt. Um diesen Job nutzen zu können, muss zuvor eine Prozess-Instanz mit Hilfe des Jobs [wfm.CreateProcessInstance](#) erzeugt werden. Dem Prozess übergebene Dokumente werden immer in den Arbeitsbereich der Workflowakte gestellt.

### Parameter:

UserId (STRING): ID des Benutzers

ProcessId (STRING): ID des Prozesses

Workspace (BASE64): enthält Dokumente im XML-Format

DataFields (BASE64): enthält Aufbau und Werte der Eingabevariablen im XML-Format

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Beispiel:****Aufbau von Workspace:**

```
<Workspace>
<Docs>
<Doc Id ="" Type="" Location="" Moveable="" Deleteable="" Workspace=""/>
<Doc Id ="" Type="" Location="" Moveable="" Deleteable="" Workspace=""/>
</Docs>
</Workspace>
```

**Hinweis:****Genauere Beschreibung von Workspace****§ Doc Struktur, die Informationen zu einem Dokument kapselt**

- § Id (INT): ID des Dokuments
- § Type (INT): Typ des Dokuments
- § Location (INT): gibt an, ob das Dokument in der SDREL(Location='1', SDREL ist die Datenbanktabelle Stamm-Dokumenten-Relation) oder Systemablage(Location='2') liegt
- § Moveable (INT): gibt an, ob das Dokument vom Info- in den Arbeitsbereich (und umgekehrt) verschoben werden darf (Moveable=1), ansonsten 0
- § Deleteable (INT): gibt an, ob das Dokument aus der Akte gelöscht werden darf (Deleteable=1), ansonsten 0
- § Workspace (INT): gibt an, ob sich das Objekt im Infobereich (0) oder im Arbeitsbereich (1) der befinden soll

**Beispiel:****Aufbau von DataFields**

```
<DataFields>
<DataField Id="iSkonto">
<![CDATA[<WFVar><String>0</String></WFVar>]]>
</DataField>
<DataField Id="iSkontofaehig">
<![CDATA[<WFVar><String>0</String></WFVar>]]>
</DataField>
<DataField Id="lPositionen">
<![CDATA[
<List TypeId="920C3899284B424EACBF881EE3A714C0">
<ListItem Id="00000000000000000000000000000001" Selection="0">
<Record>
<Member Name="iPosition"><STRING>1</STRING></Member>
<Member Name="strBezeichnung"><STRING>Tisch</STRING></Member>
</Record>
</ListItem>
</List>
]]>
</DataField>
</DataFields>
```

**Hinweis:****Genauere Beschreibung von DataFields****§ DataField: Workflowvariable**

- § Id (STRING):: Name der Workflowvariable
- § CDATA: Aufbau der Workflowvariable



**Siehe auch:**

[wfm.GetOrganisationObjects](#) , [wfm.CreateProcessInstance](#) ,

## wfm.StartWorkItem

**Beschreibung:**

Dieser Job startet einen Arbeitsschritt. Der Arbeitsschritt wird für den angegebenen Benutzer personalisiert. Dieser Job wurde durch [wfm.GetWorkItem](#) ersetzt.

**Parameter:**

UserId (STRING): ID des Benutzers

WorkItemId (STRING): Instanz-ID der Aktivität

ClientTypeId (String): ID des verwendeten Clienttyps

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[wfm.GetWorkItemList](#) , [wfm.CancelWorkItem](#) , [wfm.CompleteWorkItem](#)

## wfm.GetWorkItem

**Beschreibung:**

Dieser Job startet einen Arbeitsschritt. Der Arbeitsschritt wird für den angegebenen Benutzer personalisiert. Es werden zusätzlich alle benötigten Daten (Maske, Akte u. Workflowvariablen) für den Client zurückgeliefert.

**Parameter:**

UserId (STRING): ID des Benutzers

WorkItemId (STRING): Instanz-ID der Aktivität

ClientTypeId (String): ID des verwendeten Clienttyps

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

ExtendedAttributes (BASE64): Liste mit Daten zu den Parametern ('Attribute') im XML-Format  
Format

File (BASE64): Liste mit Dokumenten des Info-/Arbeitsbereiches der WF-Akte im XML-Format

Masks (BASE64): Maskendaten im XML-Format

Parameters (BASE64): Liste mit Parametern zur Datenmaske im XML-

RoutingList (Base64): Laufliste. Dieser Parameter ist optional.

**Siehe auch:**

[wfm.GetWorkItemList](#) , [wfm.CancelWorkItem](#) , [wfm.CompleteWorkItem](#)

**Beispiel:**

## Aufbau von ExtendedAttributes

```
<ExtendedAttributes>
<ExtendedAttribute Name="MASKID" Value="" />
<ExtendedAttribute Name="SEND_BUTTON" Value="0" />
<ExtendedAttribute Name="SENDTO_BUTTON" Value="0" />
<ExtendedAttribute Name="END_BUTTON" Value="0" />
<ExtendedAttribute Name="SIGN_ACTIVITY" Value="" />
<ExtendedAttribute Name="CHECK_PASSWORD" Value="" />
</ExtendedAttributes>
```

### Hinweis:

#### Genauere Beschreibung von ExtendedAttributes

§ ExtendedAttributes: Liste von Parametern ('Attribute') mit folgender Struktur

§ Name (STRING): Name des Attributes

§ MASKID: GUID der Workflowmaske

§ SEND\_BUTTON:

§ END\_BUTTON:

§ SIGN\_ACTIVITY: 1 = Digitale Signatur erforderlich, ansonsten 0

§ CHECK\_PASSWORD: 1 = zum Weiterleiten muss Passwort eingegeben werden, ansonsten 0

§ Value: Wert des Attributes

### Beispiel:

#### Aufbau von File

```
<File>
<Docs>
<Doc Id="" Type="" Rights="" Location="" Workspace="" Deleteable="0" Moveable="1"
UseActiveVariant="" OriginalId="" Display="" />
<Doc Id="" Type="" Rights="" Location="" Workspace="" Deleteable="1" Moveable="1"
UseActiveVariant="" OriginalId="" Display="" />
</Docs>
</File>
```

### Hinweis:

#### Genauere Beschreibung von File

§ File: kapselt die Parameter Workspace und Infospace

§ Docs: ist eine Liste von Dokumenten-Parametern ('Doc') mit dieser Struktur (oder einer Untermenge davon)

§ Id (INT): ID des Dokuments aus dem DMS

§ Type (INT): Typ des Dokuments

§ Rights (INT): Zugriffsrechte

0 = accessDenied

1 = accessView

2 = accessEdit

4 = accessDelete

8 = accessEditDataSheet

15 = accessAll

§ Location (INT): Standort des Dokuments (1: SDREL, 2: WF-Ablage (Dokument hat noch keinen Standort))

- § Workspace (INT): gibt an, ob sich das Objekt im Infobereich (0) oder im Arbeitsbereich (1) der befindet
- § Deleteable (INT): gibt an, ob das Dokument aus der Akte gelöscht werden darf (0 = nein, 1= ja)
- § Moveable (INT): gibt an, ob das Dokument in der Akte verschoben werden darf (0 = nein, 1= ja)
- § UseActiveVariant (INT): gibt an, ob für das Objekt die aktive Variante verwendet werden soll (0 = nein, 1= ja)
- § OriginalId (INT): gibt an, welches Dokument ursprünglich in die Akte gezogen wurde (welche Id dieses Dokument hatte)
- § Display (INT): gibt an, ob dieses Dokument in der Vorschau angezeigt werden soll (0 = nein, 1= ja)

### Beispiel:

#### Aufbau von Masks

```
<Masks>
<Mask Id="" Name="" Flags="" FrameWidth="" FrameHeight="">
<MaskField Id="" Name="" InternalName="" FieldName="" TabOrder=""
DataType="" InpLen="" Init="" Flags="" Flags1="" Flags2=""
InpLeft="" InpTop="" InpRight="" InpBottom="" FieldLeft=""
FieldTop="" FieldRight="" FieldBottom="" ToolTip="" ValuesId="">
<MaskFieldVal><![CDATA[ ]]></MaskFieldVal>
</MaskField>

<!-- Aufbau für Listcontrols -->
<MaskField Id="" Name="" InternalName="" TabOrder=""
DataType="" InpLen="" Init="" Flags="" Flags1="" Flags2=""
InpLeft="" InpTop="" InpRight="" InpBottom="" FieldLeft=""
FieldTop="" FieldRight="" FieldBottom="" ToolTip="" ValuesId="">
<MaskListCtrls>
<MaskListCtrl ColPos="" Name="" Type="" Length=""
ColWidth="" Color="" TextAlign="" ValuesId=""/>
<MaskListCtrlVal><![CDATA[ ]]></MaskListCtrlVal>
</MaskListCtrls>
</MaskField>

<!-- Aufbau für Pagecontrols -->
<MaskField Id="" Name="" InternalName="" TabOrder=""
DataType="" InpLen="" Init="" Flags="" Flags1="" Flags2=""
InpLeft="" InpTop="" InpRight="" InpBottom="" FieldLeft=""
FieldTop="" FieldRight="" FieldBottom="" ToolTip="" ValuesId="">
<Page Id="" Name="" Number="" IconId=""/>
<MaskFields>
<MaskField Id="" Name="" InternalName=""
TabOrder="" DataType="" InpLen="" Init="" Flags="" Flags1=""
Flags2="" InpLeft="" InpTop="" InpRight="" InpBottom=""
FieldLeft="" FieldTop="" FieldRight="" FieldBottom=""
ToolTip="" ValuesId=""/>
</MaskFields>
</Page>
</MaskField>
</Mask>
</Masks>
```

### Hinweis:

## Genauere Beschreibung von Masks

§ Masks: Liste der Masken, die Elemente dieser Liste sind vom Typ 'Mask'

§ Mask Struktur, die auch eine Liste von Maskenfeldern vom Typ 'MaskField' enthält

§ Id (STRING): ID der Maske

§ Name (STRING): Name der Maske

§ Flags (INT): Flags

§ FrameWidth (INT): Breite der Maske

§ FrameHeight (INT): Höhe der Maske

§ MaskField: Struktur, die Informationen zu einem Maskenfeld beinhaltet; außerdem beinhaltet sie entweder den Wert des Maskenfeldes ('MaskfieldVal') oder eine Liste Maskenfeld-Controls('MaskListCtrls'):

§ Id (STRING): ID des Maskenfeldes

§ Name (STRING): Name

§ InternalName (STRING): interner Name

§ TabOrder (INT): Tabulatorreihenfolge

§ DataType (INT?): Datentyp

§ InpLen (INT): Länge der Eingabe

§ Init (STRING): Initialisierungswert

§ Flags (INT): Flags

§ Flags1 (INT): weitere Flags

§ Flags2 (INT): weitere Flags

§ InpLeft (INT): X-Position des Eingabefeldes

§ InpTop (INT): Y-Position des Eingabefeldes

§ InpRight (INT): Breite der Eingabefeldes

§ InpBottom (INT): Höhe der Eingabefeldes

§ FieldLeft (INT): X-Position der Feldbezeichnung

§ FieldTop (INT): Y-Position der Feldbezeichnung

§ FieldRight (INT): Breite der Feldbezeichnung in Pixel

§ FieldBottom (INT): Höhe der Feldbezeichnung in Pixel

§ ToolTip (INT): Tooltip

§ ValuesId (INT): Verweis auf Listenfelder

§ MaskFieldVal: Wert des Maskenfeldes als CDATA

§ MaskListCtrl: Struktur, die die Informationen und Daten zu einem Maskenfeld-Control beinhaltet

§ ColPos (INT): Position der Spalte

§ Name (STRING): Name

§ Type (STRING): Typ

§ Length (INT): Länge

§ ColWidth (INT): Spaltenbreite

- § Color (INT): Farbe
- § TextAlign (INT): Bündigkeit des Textes
- § ValuesId (STRING): Verweis auf Listenfelder
- § MaskListCtrlVal: Wert des Maskenfeld-Controls als CDATA
- § Page: Struktur, die die Informationen zu einem Page-Control enthält (enthält dann wieder MaskFields)
- § Id (STRING): ID des Pagecontrols
- § Name (STRING): Name des Pagecontrols
- § Number (INT): gibt die Position ('Seitenzahl') einer Page an
- § IconId (INT): ID des Icons (aus DB-Tab. Osicons), welches auf dem Pagecontrol angezeigt werden soll

### Beispiel:

#### Aufbau von Parameters

```
<Parameters>
<Parameter FormField="" DataField="" Name="" Mode="" Selection=""
InfoText="" ListType="" ><![CDATA[ ]]></Parameter>
<Parameter FormField="" DataField="" Name="" Mode="" Selection=""
InfoText="" ListType="" ><![CDATA[ ]]></Parameter>
</Parameters>
```

### Hinweis:

#### Genauere Beschreibung von Parameters

- § Parameters: Liste der formalen Parameter mit folgender Struktur (oder Untermengen davon)
  - § FormField (STRING): ID des Feldes auf einem Formularblatt dem die Workflowvariable zugeordnet ist, wenn keine Zurordnung besteht Name der Workflowvariable
  - § DataField (STRING): ID der Workflowvariable
  - § Name (STRING): Name der Workflowvariable
  - § Mode (INT): Modus der Workflowvariable
    - § 1 = EingabeParameter
    - § 2 = AusgabeParameter
    - § 3 = Ein/Ausgabeparameter
  - § Selection (STRING): Auswahltyp bei Workflowvariablen in Listenform (single oder multi:x)
  - § InfoText (STRING): Informationstext Infotext bei Workflowvariablen in Listenform
  - § ListType (STRING): Typ der Liste
    - § ProcessList
    - § UserList
    - § UserDefList
  - § CDATA: Aufbau und Daten der Workflowvariable

### Beispiel:

#### Aufbau von RoutingList

```

<RoutingList Id="3294B433BFF6454D9C861B86B5A8AD5D"
ProcessId="BA16C21BB96D46D099E72070BCB644CC"
ActivityId="3294B433BFF6454D9C861B86B5A8AD5D" Expandable="1">
<Entries>
<Entry Nr="203" Expandable="1">
<Item Id="99825B18A8334987935684FDA3D6A40D"
ActivityId="6EE4490A48164A0FA6DC34A80099AF66" ActivityName="Rechnung erstellen"
ModelActivityName="Rechnung erstellen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectsIds>
</Item>
</Entry>
<Entry Nr="253" Expandable="1">
<Item Id="E15594D692C14FDA9AFDE8FA0A43F6E4"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen BL"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectsIds>
</Item>
<Item Id="C6DA9503CD874D69A9B703D0E06A52E8"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen GF"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectsIds>
</Item>
</Entry>
</Entries>
</RoutingList>

```

### Achtung, wird noch erweitert!!!

- § RoutingList: Laufliste mit folgender Struktur (oder Untermengen davon)
- § Id (String): ID der Laufliste. Dieser Wert wird vom Server gesetzt und darf nicht verändert werden.
- § ProcessId (String): Prozessid
- § ActivityId (String): Aktivitätsid
- § Expandable (Int): 0: Laufliste kann nicht erweitert werden, 1: Laufliste kann erweitert werden.
  - § Entries: Die Struktur fasst Einträge der Laufliste zusammen. Ein Eintrag besteht aus mehreren Elementen, die parallel ausgeführt werden können.
  - § Entry: Beschreibt einen Eintrag in der Laufliste.
  - § Nr (Int): Dient der relativen Sortierung der Einträge innerhalb der Laufliste. Die absoluten Werte haben für den Client keine Bedeutung.
  - § Expandable (Int): 0: Eintrag kann nicht erweitert werden, 1: Eintrag kann erweitert werden.
  - § Item: Beschreibt ein Element der Laufliste. Hierbei handelt es sich um eine Aktivität, eine ausführende Person und ggfs einen Termin.
  - § Id (STRING): Dient der Identifizierung. Diese ID darf nicht verändert werden und muss bei allen Jobs identisch mitgeschickt werden. Wurde ein Item durch den Client erstellt, muss dieser hier eine ID angeben.
  - § ActivityId (String): ID der Aktivität im Workflowmodell
  - § ActivityName (String): Name der Aktivität (muss nicht unbedingt mit dem Namen im Workflowmodell übereinstimmen).
  - § ActivityModelName (String): Name der Aktivität im Workflowmodell
  - § TimerId (String): ID einer Mahnfrist

- § TimerDuration(Int): Dauer der Frist
- § TimerDurationType(Int): 0: keine Frist, 1: relativ, 2: absolut
- § Changeable(Int): 0: keine Änderung möglich, 1: Das Element darf vom Client verändert werden.
- § Deleteable(Int): 0: Löschen nicht erlaubt, 1: Element darf gelöscht werden
- § Remark (String): Hinweis zur Bearbeitung (Text)
- § ObjectsIds (String): Liste von GUIDS der Bearbeiter (Rollen oder Personen), durch Komma getrennt

## Workflow-Maske, Event und Skript

- § [wfm.DeleteEvent](#)
- § [wfm.DeleteMasks](#)
- § [wfm.DeleteScript](#)
- § [wfm.GetEvents](#)
- § [wfm.GetEventTypes](#)
- § [wfm.GetGlobalScripts](#)
- § [wfm.LoadMasks](#)
- § [wfm.LoadScript](#)
- § [wfm.SaveEvent](#)
- § [wfm.SaveMasks](#)
- § [wfm.SaveScript](#)
- § [wfm.SetEventScriptRelation](#)

### wfm.DeleteEvent

#### Beschreibung:

Dieser Job löscht ein oder mehrere Workflow-Events eines bestimmten Workflowmodells.

#### Parameter:

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflowmodells

EventIds (STRING): kommaseparierte GUID-Liste von zu löschenden Events

ClientTypeId (String): ID des verwendeten Clienttyps

#### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

#### Siehe auch:

[wfm.GetEvents](#)

### wfm.DeleteMasks

#### Beschreibung:

Dieser Job löscht ein oder mehrere Workflow-Masken eines bestimmten Workflowmodells.

**Parameter:**

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflowmodells

MaskIds (STRING): kommaseparierte GUID-Liste der zu löschenden Masken

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflowmodells

MaskIds (STRING): kommaseparierte GUID-Liste der Masken, die nicht gelöscht werden konnten

## wfm.DeleteScript

**Beschreibung:**

Dieser Job löscht ein zum Workflow-Event hinterlegtes Skript.

**Parameter:**

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflows

ScriptId (STRING): ID des Skripts

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.GetEvents

**Beschreibung:**

Dieser Job liefert zu einer Aktivität oder zum gesamten Workflowmodell die Liste aller eingerichteten Events.

**Parameter:**

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflowmodells

ActivityId (STRING): ID der Aktivität. Wird der Parameter leer gelassen, werden alle Events des Workflowmodells geliefert.

ClientTypeId (String): ID des verwendeten Clienttyp. Wird der Parameter leer gelassen, werden die Events aller definierten Clienttypen geliefert.

EventTypeGroups (INT): kombinierbares Flag, gibt an welche Events angefordert werden

§ 1 – alle Clientevents werden zurückgeliefert

§ 2 – das globale Clientscript wird zurückgeliefert

§ 4 – alle Serverevents werden zurückgeliefert



§ 8 – das globale Clientscript wird zurückgeliefert

§ 15 – alle Events werden zurückgeliefert

Code (INT): bei Code=0 wird kein Skript-Code übertragen, bei Code=1 wird Skript-Code übertragen

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

Events (BASE64): enthält angeforderte Eventliste im XML-Format

### **Beispiel:**

#### Aufbau von Events

```
<Events>
<Event>
<Id></Id>
<EventType></EventType>
  <ActivityId></ActivityId>
<Params></Params>
<Description></Description>
  <ClientId></ClientId>
<Script Id=" " Name=" " Time=" " Description=" " ScriptLanguage="1">Skript
</Script>
</Event>
</Events>
```

### **Hinweis:**

Genauere Beschreibung von Events:

§ Event: enthält weitere Elemente und die Struktur 'Script'

§ Id (STRING): ID des Events

§ EventType (LONG): Typ des Events

§ 1 = BeforeForward

§ 2 = AfterForward

§ 3 = BeforeForwardTo

§ 4 = ButtonClick

§ 5 = BeforeOpen

§ 6 = AfterSignature

§ 7 = BeforeCancel

§ 8 = SimulateMaskEdit

§ 10000 = StartActivity

§ 10001 = EndActivity

§ 10003 = PersonalizeWorkItem

§ 10004 = GetWorkItemParams

§ 1000000 = Globales Serverskript

§ 1000001 = Globales Clientskript

§ Params: für Eventtyp 'ButtonClick' steht hier die ID des Buttons

- § Description (STRING): wird z. Z. nicht unterstützt
- § ActivityId (STRING): ID der Aktivität, für die das Event erstellt wurde
- § ClientTypeId (STRING): ID des Clienttyps, für den das Event erstellt wurde
- § Script: Struktur, welche das Skript enthält:
  - § Id (STRING): ID des Skripts
  - § Name (STRING): Name des Skripts
  - § Time (LONG): Erstellungszeit des Skripts
  - § Description (STRING): wird z. Z. nicht unterstützt
  - § ScriptLanguage (LONG): Skriptsprache (1 = VB-Script, 2 = J-Script)
  - § CDATA: Daten (enthält den Code des Skripts)

**Siehe auch:**

[wfm.SaveEvent](#) , [wfm.DeleteEvent](#)

## wfm.GetEventTypes

### Beschreibung:

Dieser Job liefert alle verfügbaren Eventtypen.

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

EventTypes (BASE64): enthält alle Eventtypen im XML-Format

### Beispiel:

Aufbau von EventTypes

```
<EventTypes>
<EventType Id="1" Name="BeforeForward"/>
<EventType Id="2" Name="AfterForward"/>
<EventType Id="3" Name="BeforeForwardTo"/>
<EventType Id="4" Name="ButtonClick"/>
<EventType Id="5" Name="BeforeOpen"/>
<EventType Id="6" Name="AfterSignature"/>
<EventType Id="7" Name="BeforeCancel"/>
<EventType Id="8" Name="SimulateMaskEdit"/>
<EventType Id="10000" Name="StartActivity"/>
<EventType Id="10001" Name="EndActivity"/>
<EventType Id="10002" Name="BeforeStartSubProc"/>
<EventType Id="10003" Name="PersonalizeWorkItem"/>
<EventType Id="10004" Name="GetWorkItemParams"/>
      <EventType Id="10005" Name="CancelWorkItem"/>
</EventTypes>
```

### Hinweis:

Genauere Beschreibung von EventTypes

§ EventType: Struktur, die einen Eventtyp charakterisiert

- § Id (STRING): ID des Eventtyps
- § Name (STRING): Name des Eventtyps

## wfm.GetGlobalScripts

### Beschreibung:

Dieser Job liefert zu einem Workflowmodell die globalen Skripte.

### Parameter:

WorkflowId (STRING): ID des Workflowmodells

OrganisationId (STRING): ID der Organisation

Code (LONG): 0 = kein Skriptcode wird zurückgeliefert, 1 = Skriptcode wird zurückgeliefert

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Scripts (BASE64): enthält die globalen Skripte im XML-Format

### Beispiel:

Aufbau von Scripts

```
<Scripts>
<Script Id = " " Type="2"><![CDATA[...]]></Script>
<Script Id = " " Type="3"><![CDATA[...]]></Script>
</Scripts>
```

### Hinweis:

Genauere Beschreibung von Script

§ Script: Struktur, die ein globales Script charakterisiert

§ Id (STRING): Skriptid

§ Type (LONG): Gibt an, ob es sich um ein Serverskript (2) oder ein Clientskript(3) handelt.

## wfm.LoadMasks

### Beschreibung:

Dieser Job liefert die angegebenen Masken oder alle Masken zu einem Workflowmodell mit Unterstruktur (Felder, ListCtrlCols, Kataloge).

### Parameter:

OrganisationId (STRING): ID der Organisation, in der der Workflow mit den Masken liegt

WorkflowId (STRING): ID des Workflow mit den Masken

MaskIds (STRING): Ids der angeforderten Masken (kommasepariert); leer = es werden alle Masken des Workflowmodells geladen

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

OrganisationId (STRING): ID der Organisation in der der Workflow mit den Masken liegt

WorkflowId (STRING): ID des Workflow mit den Masken

Masks (BASE64): Maskendaten im XML-Format

**Beispiel:****Aufbau von Masks**

```

<Masks>
<Mask Id="" Name="" Flags="" FrameWidth="" FrameHeight="">
<MaskField Id="" Name="" InternalName="" FieldName="" TabOrder=""
DataType="" InpLen="" Init="" Flags="" Flags1="" Flags2=""
InpLeft="" InpTop="" InpRight="" InpBottom="" FieldLeft=""
FieldTop="" FieldRight="" FieldBottom="" ToolTip="" ValuesId="">
<MaskFieldVal><![CDATA[ ]]></MaskFieldVal>
</MaskField>

<!-- Aufbau für Listcontrols -->
<MaskField Id="" Name="" InternalName="" TabOrder=""
DataType="" InpLen="" Init="" Flags="" Flags1="" Flags2=""
InpLeft="" InpTop="" InpRight="" InpBottom="" FieldLeft=""
FieldTop="" FieldRight="" FieldBottom="" ToolTip="" ValuesId="">
<MaskListCtrls>
<MaskListCtrl ColPos="" Name="" Type="" Length=""
ColWidth="" Color="" TextAlign="" ValuesId=""/>
<MaskListCtrlVal><![CDATA[ ]]></MaskListCtrlVal>
</MaskListCtrls>
</MaskField>

<!-- Aufbau für Pagecontrols -->
<MaskField Id="" Name="" InternalName="" TabOrder=""
DataType="" InpLen="" Init="" Flags="" Flags1="" Flags2=""
InpLeft="" InpTop="" InpRight="" InpBottom="" FieldLeft=""
FieldTop="" FieldRight="" FieldBottom="" ToolTip="" ValuesId="">
<Page Id="" Name="" Number="" IconId=""/>
<MaskFields>
<MaskField Id="" Name="" InternalName=""
TabOrder="" DataType="" InpLen="" Init="" Flags="" Flags1=""
Flags2="" InpLeft="" InpTop="" InpRight="" InpBottom=""
FieldLeft="" FieldTop="" FieldRight="" FieldBottom=""
ToolTip="" ValuesId=""/>
</MaskFields>
</Page>
</MaskField>
</Mask>
</Masks>

```

**Hinweis:****Genauere Beschreibung von Masks**

§ Masks: Liste der Masken, die Elemente dieser Liste sind vom Typ 'Mask'

§ Mask Struktur, die auch eine Liste von Maskenfeldern vom Typ 'MaskField' enthält

§ Id (STRING): ID der Maske

§ Name (STRING): Name der Maske

§ Flags (INT): Flags

§ FrameWidth (INT): Breite der Maske

§ FrameHeight (INT): Höhe der Maske

- § MaskField: Struktur, die Informationen zu einem Maskenfeld beinhaltet; außerdem beinhaltet sie entweder den Wert des Maskenfeldes ('MaskfieldVal') oder eine Liste Maskenfeld-Controls('MaskListCtrls'):
- § Id (STRING): ID des Maskenfeldes
- § Name (STRING): Name
- § InternalName (STRING): interner Name
- § TabOrder (INT): Tabulatorreihenfolge
- § DataType (INT?): Datentyp
- § InpLen (INT): Länge der Eingabe
- § Init (STRING): Initialisierungswert
- § Flags (INT): Flags
- § Flags1 (INT): weitere Flags
- § Flags2 (INT): weitere Flags
- § InpLeft (INT): X-Position des Eingabefeldes
- § InpTop (INT): Y-Position des Eingabefeldes
- § InpRight (INT): Breite der Eingabefeldes
- § InpBottom (INT): Höhe der Eingabefeldes
- § FieldLeft (INT): X-Position der Feldbezeichnung
- § FieldTop (INT): Y-Position der Feldbezeichnung
- § FieldRight (INT): Breite der Feldbezeichnung in Pixel
- § FieldBottom (INT): Höhe der Feldbezeichnung in Pixel
- § ToolTip (INT): Tooltip
- § ValuesId (INT): Verweis auf Listenfelder
- § MaskFieldVal: Wert des Maskenfeldes als CDATA
- § MaskListCtrl: Struktur, die die Informationen und Daten zu einem Maskenfeld-Control beinhaltet
- § ColPos (INT): Position der Spalte
- § Name (STRING): Name
- § Type (STRING): Typ
- § Length (INT): Länge
- § ColWidth (INT): Spaltenbreite
- § Color (INT): Farbe
- § TextAlign (INT): Bündigkeit des Textes
- § ValuesId (STRING): Verweis auf Listenfelder
- § MaskListCtrlVal: Wert des Maskenfeld-Controls als CDATA
- § Page: Struktur, die die Informationen einem Page-Control (enthält dann wieder MaskFields)
- § Id (STRING): ID des Pagecontrols
- § Name (STRING): Name des Pagecontrols
- § Number (INT): gibt die Position ('Seitenzahl') einer Page an

§ IconId (INT): ID des Icons (aus DB-Tab. Osicons), welches auf dem Pagecontrol angezeigt werden soll

**Siehe auch:**

[wfm.SaveMasks](#)

## wfm.LoadScript

### **Beschreibung:**

Dieser Job liefert ein Skript aus der Datenbank.

### **Parameter:**

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflows

ScriptId (STRING): ID des Skripts

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

ScriptCode (STRING): der angeforderte Skriptcode

**Siehe auch:**

[wfm.SaveScript](#)

## wfm.SaveEvent

### **Beschreibung:**

Dieser Job richtet zu einer Aktivität ein Event ein. Wenn der Parameter EventId gesetzt ist, werden für ein bestehendes Event die Werte Params und Description neu gesetzt.

### **Parameter:**

EventId (STRING): ID eines Events (muss leer sein, wenn es sich um einen neuen Event handelt)

WorkflowId (STRING): ID eines Workflowmodells

ActivityId (STRING): ID einer Aktivität

EventType (INT): Typ eines Events

Params (STRING): für Eventtyp 'ButtonClick' wird hier die ID des Buttons übergeben

Description (STRING): Beschreibung zum Event

OrganisationId (STRING): Organisationsid

ClientTypeId (String): ID des verwendeten Clienttyps

### **Rückgabe:**

(INT): 0: Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

EventId (STRING): ID eines Events

### **Hinweis:**

## Eventtypen

- § 1 = BeforeForward
- § 2 = AfterForward
- § 3 = BeforeForwardTo
- § 4 = ButtonClick
- § 5 = BeforeOpen
- § 6 = AfterSignature
- § 7 = BeforeCancel
- § 8 = SimulateMaskEdit
- § 10000 = StartActivity
- § 10001 = EndActivity
- § 10002 = BeforeStartSubProc
- § 10003 = PersonalizeWorkItem
- § 10004 = GetWorkItemParams

### Siehe auch:

[wfm.GetEvents](#) , [wfm.DeleteEvent](#) , [wfm.SaveScript](#) , [wfm.SetEventScriptRelation](#)

## wfm.SaveMasks

### Beschreibung:

Dieser Job speichert Änderungen ein oder mehrerer Masken mit Unterstruktur (Felder, ListCtrlCols, Kataloge).

### Parameter:

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflowmodells

Masks (BASE64): Struktur, die die zu speichernden Masken enthält (XML-Format)

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflowmodells

MaskIds (STRING): kommaseparierte ID-Liste von Masken die nicht gespeichert werden konnten

### Beispiel:

#### Aufbau von Masks

```
<Masks>
<Mask Id="" ModState="" />
<!--für 0=MODSTATE_UNMODIFIED, 3=MODSTATE_DELETED -->
<!--oder-->
<Mask Id="" ModState="" Name="" Flags="" FrameWidth="" FrameHeight="">
<!--für 1=MODSTATE_CHANGED, 2=MODSTATE_NEW-->
<MaskField Id="" ModState="" />
```

```

<!--für 0=MODSTATE_UNMODIFIED, 3=MODSTATE_DELETED-->
<MaskField Id="" ModState="" Name="" InternalName="" FieldName=""
TabOrder="" DataType="" InpLen="" Init="" Flags="" Flags1=""
Flags2="" InpLeft="" InpTop="" InpRight="" InpBottom=""
FieldLeft="" FieldTop="" FieldRight="" FieldBottom="" ToolTip=""
ValuesId="">
<!--für 1=MODSTATE_CHANGED, 2=MODSTATE_NEW-->
<MaskFieldVal Id="" ModState="">
<![CDATA[  ]]>
</MaskFieldVal>
</MaskField>
</Mask>
</Masks>

```

## wfm.SaveScript

### Beschreibung:

Dieser Job speichert ein Skript oder legt ein neues Skript an (Action = 2).

### Parameter:

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflows

Id (STRING): ID des Skripts

Name (STRING): Name des Skripts

Description (STRING): Kurz-Beschreibung zum Skripts

Action (INT): Aktion, die ausgeführt werden soll

§ 1 = das alte Script wird überschrieben

§ 2 = es wird ein neues Skript erstellt

ScriptCode (STRING): Skript-Code

Type (INT): Die Art des Skripts

§ 1 = Eventskript

§ 2 = globales Serverskript

§ 3 = gloables Clientskript

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

ScriptId (STRING): ID des Skripts

### Siehe auch:

[wfm.SaveEvent](#) , [wfm.SetEventScriptRelation](#)

## wfm.SetEventScriptRelation

### Beschreibung:

Dieser Job verknüpft ein Event mit einem Skript.

### Parameter:



OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflows

EventId (STRING): ID des Events

ScriptId (STRING): ID des Skripts

Action (INT): Aktion die mit den angegebenen Parametern ausgeführt werden soll:

§ 1 = Verknüpfung setzen

§ 2 = Verknüpfung löschen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[wfm.SaveScript](#) , [wfm.SaveEvent](#)

## Administration und Historienverwaltung

### Administration

§ [wfm.AdminDeleteStatisticReportConfigs](#)

§ [wfm.AdminDeleteStatisticReports](#)

§ [wfm.AdminDeleteProcesses](#)

§ [wfm.AdminGetActivityVariables](#)

§ [wfm.AdminGetLockInfo](#)

§ [wfm.AdminGetProcessActivities](#)

§ [wfm.AdminGetProcessList](#)

§ [wfm.AdminGetProcessListByRole](#)

§ [wfm.AdminGetProcessListByUser](#)

§ [wfm.AdminGetProcessLocks](#)

§ [wfm.AdminGetProcessReport](#)

§ [Wfm.AdminGetStatisticReportConfigs](#)

§ [wfm.AdminGetStatisticReportData](#)

§ [wfm.AdminGetStatisticReports](#)

§ [wfm.AdminGetRoleProcesses](#)

§ [wfm.AdminGetUserProcesses](#)

§ [wfm.AdminGetWorkerqueue](#)

§ [wfm.AdminGetWorkflowList](#)

§ [wfm.AdminReleaseLock](#)

§ [wfm.AdminRequestStatisticReport](#)

§ [wfm.AdminResumeActivity](#)

§ [wfm.AdminResumeProcess](#)

§ [wfm.AdminRollbackProcess](#)

- § [wfm.AdminSaveActivityVariables](#)
- § [wfm.AdminSaveStatisticReportConfig](#)
- § [wfm.AdminSuspendActivity](#)
- § [wfm.AdminSuspendProcess](#)
- § [wfm.AdminTerminateActivity](#)
- § [wfm.AdminTerminateProcess](#)

## wfm.AdminDeleteStatisticReportConfigs

### **Beschreibung:**

Dieser Job löscht alle angegebenen Reportkonfigurationen.

### **Parameter:**

OrganisationId (STRING): ID der Organisation in der sich die Reportkonfigurationen befinden

ConfigIds (STRING): kommaseparierte Liste von Reportkonfiguration-GUID's, die gelöscht werden sollen

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

keine

### § **Siehe auch:**

§ [Wfm.AdminGetStatisticReportConfigs](#) , [wfm.AdminSaveStatisticReportConfig](#)

## wfm.AdminDeleteStatisticReports

### **Beschreibung:**

Dieser Job löscht alle angegebenen Statistikreports.

### **Parameter:**

ReportIds (STRING): kommaseparierte Liste von Report-GUID's, die gelöscht werden sollen

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

keine

## wfm.AdminDeleteProcesses

### **Beschreibung:**

Dieser Job löscht alle angegebenen Workflow-Prozesse. Sind in der Workflowakte Dokumente enthalten, die während des Workflow-Prozesses angelegt wurden und sich noch nicht in enaio® befinden, werden diese in die Ablage des enaio® clients des Jobausführenden eingefügt. Bevor ein Prozess gelöscht werden kann, muss der Prozess über den Job [wfm.AdminSuspendProcess](#) angehalten werden.

**Parameter:**

OrganisationId (STRING): GUID der Organisation, zu der die Prozesse gehören

Processes (STRING): kommaseparierte Liste von Prozess-GUID's, die gelöscht werden sollen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (STRING): kommaseparierte Liste von Prozess-GUID's, die nicht gelöscht werden konnten

**Siehe auch:**

[wfm.AdminGetProcessList](#) , [wfm.GetOrganisations](#) ,  
[wfm.AdminSuspendProcess](#)

## wfm.AdminGetActivityVariables

**Beschreibung:**

Dieser Job liefert die ID's, Namen, Aufbau und Werte aller Workflowvariablen zu einer Aktivität im XML-Format.

**Parameter:**

RActivityId (STRING): Instanz-ID einer Aktivität

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

DataFields (BASE64): Liste mit Informationen zu Variablen der Aktivität im XML-Format

**Beispiel:****Aufbau von DataFields**

```
<DataFields>
<DataField Id="72FA13878B7C4744BB33C58B5AAFF5F0" Name="wfProtocol">
<![CDATA[
<WFVar>
<List TypeId="75FCEF515EDE4CF5A1BA8DE94FD26023"></List>
<Types>
<Type Id="75FCEF515EDE4CF5A1BA8DE94FD26023">
<Record>
<Member Name="Datum"><STRING/></Member>
<Member Name="Uhrzeit"><STRING/></Member>
<Member Name="Aktivität"><STRING/></Member>
<Member Name="Benutzer"><STRING/></Member>
<Member Name="Protokoll"><STRING/></Member>
</Record>
</Type>
</Types>
</WFVar>
]]>
</DataField>
<DataField Id="4170579B168642B0E8A172BC73459" Name="Testvariable">
<![CDATA[
<WFVar>
<String>Hier steht ein String.</String>
<Types></Types>
```

```
</WFVar>
]]>
</DataField>
</DataFields>
```

**Hinweis:**

Genauere Beschreibung von DataFields

## § DataField

- § Id (STRING): GUID der Variablen
- § Name (STRING): Name der Variablen
- § CDATA: Aufbau und Werte der Variablen

**Siehe auch:**

[wfm.AdminGetProcessActivities](#) , [wfm.AdminSaveActivityVariables](#)

## wfm.AdminGetProcessActivities

**Beschreibung:**

Dieser Job liefert zu einem Prozess alle Aktivitäten.

**Parameter:**

ProcessId (STRING): GUID des Prozesses

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Activities (BASE64): enthält angeforderte Angaben im XML-Format

**Beispiel:****Aufbau von Activities**

```
<Activities>
<Activity>
<Name></Name>
<Id></Id>
<RActivityId></RActivityId>
<CreationTime></CreationTime>
<Owner></Owner>
<OwnerId></OwnerId>
<AccessTime></AccessTime>
<EndTime></EndTime>
<ReminderTime></ReminderTime>
<ReminderState></ReminderState>
<State></State>
<ClosureTime></ClosureTime>
<WorkItem></WorkItem>
    <LoopCount></LoopCount>
    <ExecutionPoints></ExecutionPoints>
<RLoopId></RLoopId>
    <ActivityNr></ActivityNr>
</Activity>
</Activities>
```

**Hinweis:**

## Genauere Beschreibung von Activities

### § Activity: Struktur, die alle Informationen zu einer Aktivität enthält

- § Name (STRING): Name der Aktivität
- § Id (STRING): GUID der Aktivität (aus dem Workflowmodell)
- § RActivityId (STRING): Instanz-ID der Aktivität
- § CreationTime (LONG): Erstellungszeit der Aktivität am Server
- § Owner (STRING): Besitzer, hat die Aktivität personalisiert
- § OwnerId (STRING): ID des Besitzers
- § AccessTime (LONG): Zeitpunkt des letzten Zugriffs auf die Aktivität
- § EndTime (LONG): Zeitpunkt des Endes der Aktivität
- § ReminderTime (LONG): Mahnfrist
- § ReminderState (LONG): Status der Mahnung (1 = Mahnfrist überschritten, ansonsten 0)
- § State (LONG): Status der Aktivität
  - § 0x1 = Die Aktivität wurde initialisiert.
  - § 0x2 = Die Aktivität wurde gestartet (z. B. wurden Variablen angelegt).
  - § 0x4= Das StartActivity-Event wurde ausgeführt
  - § 0x8= Das EndActivity-Event wurde ausgeführt.
  - § 0x10=Nur bei Schleifen: Die Schleifenbedingung wurde geprüft.
  - § 0x20=Nur bei Schleifen: Der Schleifenkörper wird abgearbeitet.
  - § 0x40=Nur bei Arbeitsschritten: Der Arbeitsschritt liegt in den Eingangskörben bereit.
  - § 0x80=Nur bei Arbeitsschritten: Der Arbeitsschritt ist personalisiert.
  - § 0x100= Warten auf den Ablauf eine Sperrfrist.
  - § 0x400=Die wurde Aktivität ausgeführt, z. B. wurde ein Arbeitsschritt weitergeleitet oder eine Schleifenaktivität komplett abgearbeitet.
  - § 0x800=Nachfolgende Aktivitäten wurden berechnet und eventuell auch schon erstellt.
  - § 0x1000=Die Aktivität ist beendet, aber es wurden noch keine Folgeaktivitäten angestoßen.
  - § 0x2000 = Die Aktivität wurde durch einen Benutzer angehalten.
  - § 0x4000=Die Aktivität ist beendet.
  - § 0x8000=Nur bei Multi-Instanz Aktivitäten: Die Aktivität wurde erstellt.
  - § 0x10000=Nur bei Ad-hoc-aktivitäten: Die Ad-hoc-aktivität wurde erstellt.
  - § 0x20000=Aktivität abgebrochen
  - § 0x40000= Nur bei Ad-hoc-aktivitäten: Die Ad-hoc-aktivität wird ausgeführt.
  - § 0x10000000= Die Aktivität wurde durch das System aufgrund eines Fehlers angehalten.
- § ClosureTime (LONG): Sperrfrist zur Aktivität
- § WorkItem (LONG): 1 -> Aktivität in Eingangskörben sichtbar, ansonsten 0
- § LoopCount (LONG): wenn es sich um eine Schleifenaktivität handelt wird hier die Anzahl der Schleifendurchläufe angegeben, ansonsten 0
- § ExecutionPoints (LONG): Aufsatzpunkte für Job [wfm.AdminRollbackProcess](#)

- § 100 = Aktivität ist erstellt
- § 200 = Aktivität ist beendet
- § RLoopId (STRING): InstanzId der umgebenden Schleife, wenn es keine gibt ist der Parameter leer
- § ActivityNr (LONG): gibt an, an welcher Position die Aktivität im Prozess erstellt wurde

**Siehe auch:**

[wfm.AdminGetProcessList](#) , [wfm.AdminGetActivityVariables](#)

**wfm.AdminGetProcessList****Beschreibung:**

Dieser Job liefert zu einem Workflowmodell eine Liste aller laufenden Prozesse.

**Parameter:**

OrganisationId (STRING): ID der Organisation

WorkflowId (STRING): ID des Workflow-Modells

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (BASE64): Liste aller laufenden Prozesse im XML-Format

**Beispiel:****Aufbau von Processes**

```
<Processes>
<Process Id="" Name="" Subject="" State="" SuspendedActivity="">
<Creation UserId="" UserName="" Time="" />
<LastActivity ExecTime="" Name="" Id="" UserId="" UserName="" />
</Process>
<Process Id="" Name="" Subject="" State="" SuspendedActivity="">
<Creation UserId="" UserName="" Time="" />
<LastActivity ExecTime="" Name="" Id="" UserId="" UserName="" />
</Process>
</Processes>
```

**Hinweis:****Genauere Beschreibung von Processes**

- § Id (STRING) = ID eines Prozesses
- § Name (STRING): Name eines Prozesses
- § Subject (STRING): Name des Prozesses
- § State (LONG): Zustand eines Prozesses
  - § 1 = INIT (Prozess ist initialisiert)
  - § 2 = RUNNING (Prozess läuft)
  - § 4 = SUSPENDED (Prozess wurde angehalten -> wird noch nicht unterstützt)
  - § 8 = ACTIVE (Prozess läuft u. mindestens eine Aktivität ist personalisiert)
  - § 16 = TERMINATED (Prozess wurde abgebrochen -> wird noch nicht unterstützt)

- § 32 = COMPLETED (Prozess erfolgreich abgeschlossen)
- § 64 = SYSSUSPENDED (Prozess wurde durch Engine angehalten z. B. Aufgrund eines Fehlers im Eventskript)
- § SuspendedActivity (LONG): 1 – mindestens eine Aktivität des Prozesses ist angehalten, sonst 0
- § Creation: Struktur, die Informationen zur Erzeugung des jeweiligen Prozesses kapselt
  - § UserId (STRING): Benutzer-ID des Erstellers
  - § UserName (STRING): Benutzername des Erstellers
  - § Time (LONG): Erstellungszeit
- § LastActivity: Struktur, die Informationen zur letzten Benutzung des Prozesses kapselt
  - § ExecTime (LONG): Letzte Ausführungszeit
  - § Name (STRING): Name der zuletzt ausgeführten Aktivität
  - § Id (STRING): ID der zuletzt ausgeführten Aktivität
  - § UserId (STRING): ID des Ausführers der letzten Aktivität
  - § UserName (STRING): Name des Ausführers der letzten Aktivität

**Siehe auch:**

[wfm.AdminGetWorkflowList](#) , [wfm.GetOrganisations](#) , [wfm.AdminDeleteProcesses](#)

## wfm.AdminGetProcessListByRole

### Beschreibung:

Dieser Job liefert zu einer Rollenid alle Prozesse, zu denen sich aktuell ein Arbeitsschritt im Eingangskorb befindet.

### Parameter:

OrganisationId (STRING): ID der Organisation

RoleId (STRING): ID der Rolle

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Processes (BASE64): Liste aller Arbeitsschritte im Eingangskorb im XML-Format

### Beispiel:

#### Aufbau von Processes

```
<Processes>
<Process Id="" Name="" Subject="" State="">
<Creation UserId="" UserName="" Time="" />
<Activity CreationTime="" Name="" Id="" Owner="" OwnerId=""
ReminderTime="" OwnerTime="" WICreationTime="" State="" />
</Process>
</Processes>
```

### Hinweis:

Genauere Beschreibung von Processes

- § Id (STRING): ID des Prozesses

- § Name (STRING): Name des Prozesses
- § Subject (STRING): Betreff des Prozesses
- § State (LONG): Zustand des Prozesses
  - § 1 = INIT (Prozess ist initialisiert)
  - § 2 = RUNNING (Prozess läuft)
  - § 4 = SUSPENDED (Prozess wurde angehalten -> wird noch nicht unterstützt)
  - § 8 = ACTIVE (Prozess läuft u. mindestens eine Aktivität ist personalisiert)
  - § 16 = TERMINATED (Prozess wurde abgebrochen -> wird noch nicht unterstützt)
  - § 32 = COMPLETED (Prozess erfolgreich abgeschlossen)
  - § 64 = SYSSUSPENDED (Prozess wurde durch Engine angehalten z. B. Aufgrund eines Fehlers im Eventskript)
- § Creation: Struktur, die Informationen zur Erzeugung des jeweiligen Prozesses kapselt
  - § UserId (STRING): Benutzer-ID des Erstellers
  - § UserName (STRING): Benutzername des Erstellers
  - § Time (LONG): Erstellungszeit des Prozesses
- § Activity: Struktur, die Informationen zur Aktivität des Prozesses kapselt
  - § CreationTime (LONG): Erstellungszeit
  - § Name (STRING): Name der Aktivität
  - § Id (STRING): ID der Aktivität
  - § ReminderTime (LONG): Mahnfrist (wenn 0, dann gibt es keine Mahnfrist)
  - § Owner (STRING): Name des Benutzers, der den Arbeitsschritt personalisiert hat
  - § OwnerId (STRING): ID des Benutzers, der den Arbeitsschritt personalisiert hat
  - § OwnerTime (LONG): Zeitpunkt, an dem der Arbeitsschritt personalisiert wurde
  - § WICreationTime (LONG): Zeitpunkt, an dem der Arbeitsschritt im Eingangskorb erstellt wurde
  - § State (LONG): Status der Aktivität
    - § 0x1 = Die Aktivität wurde initialisiert.
    - § 0x2 = Die Aktivität wurde gestartet (z. B. wurden Variablen angelegt).
    - § 0x4= Das StartActivity-Event wurde ausgeführt
    - § 0x8= Das EndActivity-Event wurde ausgeführt.
    - § 0x10=Nur bei Schleifen: Die Schleifenbedingung wurde geprüft.
    - § 0x20=Nur bei Schleifen: Der Schleifenkörper wird abgearbeitet.
    - § 0x40=Nur bei Arbeitsschritten: Der Arbeitsschritt liegt in den Eingangskörben bereit.
    - § 0x80=Nur bei Arbeitsschritten: Der Arbeitsschritt ist personalisiert.
    - § 0x100= Warten auf den Ablauf eine Sperrfrist.
    - § 0x400=Die wurde Aktivität ausgeführt, z. B. wurde ein Arbeitsschritt weitergeleitet oder eine Schleifenaktivität komplett abgearbeitet.



- § 0x800=Nachfolgende Aktivitäten wurden berechnet und eventuell auch schon erstellt.
- § 0x1000=Die Aktivität ist beendet, aber es wurden noch keine Folgeaktivitäten angestoßen.
- § 0x2000 = Die Aktivität wurde durch einen Benutzer angehalten.
- § 0x4000=Die Aktivität ist beendet.
- § 0x8000=Nur bei Multi-Instanz Aktivitäten: Die Aktivität wurde erstellt.
- § 0x10000=Nur bei Ad-hoc-aktivitäten: Die Ad-hoc-aktivität wurde erstellt.
- § 0x20000=Aktivität abgebrochen
- § 0x40000= Nur bei Ad-hoc-aktivitäten: Die Ad-hoc-aktivität wird ausgeführt.
- § 0x10000000= Die Aktivität wurde durch das System aufgrund eines Fehlers angehalten.

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.AdminGetRoleProcesses](#) , [wfm.AdminGetProcessActivities](#) , [wfm.AdminDeleteProcesses](#)

## wfm.AdminGetProcessListByUser

**Beschreibung:**

Dieser Job liefert zu einer Benutzer-ID alle Prozesse, zu denen sich aktuell ein Arbeitsschritt im Eingangskorb befindet.

**Parameter:**

OrganisationId (STRING): ID der Organisation

UserId (STRING): ID des Benutzers

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (BASE64): enthält angeforderte Angaben im XML-Format

**Beispiel:**

Aufbau von Processes

```
<Processes>
<Process Id="" Name="" Subject="" State="">
<Creation UserId="" UserName="" Time="" />
<Activity CreationTime="" Name="" Id="" ReminderTime=""
OwnerTime="" WICreationTime="" State="" />
</Process>
</Processes>
```

**Hinweis:**

Genauere Beschreibung von Processes

- § Id (STRING): ID des Prozesses
- § Name (STRING): Name des Prozesses
- § Subject (STRING): Betreff des Prozesses
- § State (LONG): Zustand des Prozesses
  - § 1 = INIT (Prozess ist initialisiert)

- § 2 = RUNNING (Prozess läuft)
  - § 4 = SUSPENDED (Prozess wurde angehalten -> wird noch nicht unterstützt)
  - § 8 = ACTIVE (Prozess läuft u. mindestens eine Aktivität ist personalisiert)
  - § 16 = TERMINATED (Prozess wurde abgebrochen -> wird noch nicht unterstützt)
  - § 32 = COMPLETED (Prozess erfolgreich abgeschlossen)
  - § 64 = SYSSUSPENDED (Prozess wurde durch Engine angehalten z. B. Aufgrund eines Fehlers im Eventskript)
- § Creation: Struktur, die Informationen zur Erzeugung des jeweiligen Prozesses kapselt
- § UserId (STRING): Benutzer-ID des Erstellers
  - § UserName (STRING): Benutzername des Erstellers
  - § Time (LONG): Erstellungszeit des Prozesses
- § Activity: Struktur, die Informationen zur Aktivität des Prozesses kapselt
- § CreationTime (LONG): Erstellungszeit
  - § Name (STRING): Name der Aktivität
  - § Id (STRING): ID der Aktivität
  - § ReminderTime (LONG): Mahnfrist (wenn 0, dann gibt es keine Mahnfrist)
  - § Owner (STRING): Name des Benutzers, der den Arbeitsschritt personalisiert hat
  - § OwnerId (STRING): ID des Benutzers, der den Arbeitsschritt personalisiert hat
  - § OwnerTime (LONG): Zeitpunkt, an dem der Arbeitsschritt personalisiert wurde
  - § WICreationTime (LONG): Zeitpunkt, an dem der Arbeitsschritt im Eingangskorb erstellt wurde
  - § State (LONG): Status der Aktivität
    - § 0x1 = Die Aktivität wurde initialisiert.
    - § 0x2 = Die Aktivität wurde gestartet (z. B. wurden Variablen angelegt).
    - § 0x4= Das StartActivity-Event wurde ausgeführt
    - § 0x8= Das EndActivity-Event wurde ausgeführt.
    - § 0x10=Nur bei Schleifen: Die Schleifenbedingung wurde geprüft.
    - § 0x20=Nur bei Schleifen: Der Schleifenkörper wird abgearbeitet.
    - § 0x40=Nur bei Arbeitsschritten: Der Arbeitsschritt liegt in den Eingangskörben bereit.
    - § 0x80=Nur bei Arbeitsschritten: Der Arbeitsschritt ist personalisiert.
    - § 0x100= Warten auf den Ablauf eine Sperrfrist.
    - § 0x400=Die wurde Aktivität ausgeführt, z. B. wurde ein Arbeitsschritt weitergeleitet oder eine Schleifenaktivität komplett abgearbeitet.
    - § 0x800=Nachfolgende Aktivitäten wurden berechnet und eventuell auch schon erstellt.
    - § 0x1000=Die Aktivität ist beendet, aber es wurden noch keine Folgeaktivitäten angestoßen.
    - § 0x2000 = Die Aktivität wurde durch einen Benutzer angehalten.
    - § 0x4000=Die Aktivität ist beendet.

- § 0x8000=Nur bei Multi-Instanz Aktivitäten: Die Aktivität wurde erstellt.
- § 0x10000=Nur bei Ad-hoc-aktivitäten: Die Ad-hoc-aktivität wurde erstellt.
- § 0x20000=Aktivität abgebrochen
- § 0x40000= Nur bei Ad-hoc-aktivitäten: Die Ad-hoc-aktivität wird ausgeführt.
- § 0x10000000= Die Aktivität wurde durch das System aufgrund eines Fehlers angehalten.

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.AdminGetUserProcesses](#) , [wfm.AdminGetProcessActivities](#) ,  
[wfm.AdminDeleteProcesses](#)

## wfm.AdminGetRoleProcesses

**Beschreibung:**

Dieser Job liefert alle Rollen einer bestimmten Organisation, zu denen Arbeitsschritte im Eingangskorb vorhanden sind.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Roles (BASE64): Liste aller Rollen im XML-Format

**Beispiel:**

Aufbau von Roles

```
<Roles>
<Role Id=" " Name=" " ProcessCount=" " />
<Role Id=" " Name=" " ProcessCount=" " />
</Roles>
```

**Hinweis:**

Genauere Beschreibung von Roles

§ Role: Struktur, die Informationen über Rollen und Prozessanzahl zusammenbringt

§ Id (STRING): ID der Rolle

§ Name (STRING): Name der Rolle

§ ProcessCount (LONG): Anzahl der Prozesse, zu denen Arbeitsschritte im Eingangskorb vorhanden sind

**Siehe auch:**

[wfm.GetOrganisations](#)

## wfm.AdminGetUserProcesses

**Beschreibung:**

Dieser Job liefert alle Benutzer einer bestimmten Organisation, zu denen Arbeitsschritte im Eingangskorb vorhanden sind.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Users (BASE64): Liste aller Benutzer im XML-Format

**Beispiel:**

Aufbau von Users

```
<Users>
<User Id=" " Name=" " ProcessCount=" " />
<User Id=" " Name=" " ProcessCount=" " />
</Users>
```

**Hinweis:**

Genauere Beschreibung von Users

§ User: Struktur, die Informationen von einem Benutzer und der Prozessanzahl zusammenbringt

§ Id (STRING): ID des Benutzers

§ Name (STRING): Name des Benutzers

§ ProcessCount (LONG): Anzahl der Prozesse des Benutzers, zu denen Arbeitsschritte im Eingangskorb vorhanden sind

**Siehe auch:**

[wfm.GetOrganisations](#)

## wfm.AdminGetWorkflowList

**Beschreibung:**

Dieser Job liefert alle in Benutzung befindlichen Workflowmodelle (Status ACTIVE/INUSE) und die Anzahl der laufenden Prozesse einer Organisation.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Workflows (BASE64): enthält angeforderte Angaben im XML-Format

**Beispiel:**

Aufbau von Workflows

```
<Workflows>
<Workflow Id=" " Name=" " ProcessCount=" " />
<Workflow Id=" " Name=" " ProcessCount=" " />
</Workflows>
```

**Hinweis:**

## Genauere Beschreibung von Workflows

§ Workflow: Struktur die Informationen zu einem Workflowmodell kapselt

§ Id (STRING): ID des Workflowmodells

§ Name (STRING): Name des Workflowmodells

§ ProcessCount (LONG): Anzahl der laufenden Prozesse zu diesem Workflowmodell

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.AdminGetProcessList](#)

## wfm.AdminRequestStatisticReport

### Beschreibung:

Dieser Job fordert die Erzeugung eines neuen Reports zu einer gegebenen Statistikreportkonfiguration an.

### Parameter:

OrganisationId (STRING): ID der Organisation

UserId (STRING): ID des Benutzers, der die Aktion ausführt

ConfigId (STRING): ID der Statistikreportkonfiguration

CompileTime (INT): Zeitpunkt zu dem der Report frühestens generiert werden soll

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

ReportId (STRING): ID des Reports

CreatorId (STRING): ID des Benutzers der den Report angefordert hat

CreatorName (STRING): Name des Personenobjekts des Benutzers

CreationTime (INT): Zeitstempel (wann wurde der Report erstellt), Bei State = 0 oder 1 ist dies der Zeitpunkt zu dem der Report angefragt wurde, sonst ist dies der Zeitpunkt bei dem mit der Erstellung des Reports begonnen wurde

§ State (INT): Status des Reports

0 – Report ist angefragt

§ 1 – Report momentan wird erzeugt

§ 2 – Report ist fertig erstellt

## wfm.AdminResumeActivity

### Beschreibung:

Dieser Job gibt eine angehaltene Aktivität wieder zur Bearbeitung frei.

### Parameter:

RActivityId (STRING): InstanzId der Aktivität

UserId (STRING): ID des Benutzers, der die Aktion ausführt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

State (INT): [Status der Aktivität](#) nach dem Fortsetzen

**Siehe auch:**

[wfm.AdminSuspendActivity](#)

## wfm.AdminResumeProcess

**Beschreibung:**

Dieser Job gibt eine angehaltenen Prozess wieder zur Bearbeitung frei. Sind aus diesem Prozess auch einzelne Aktivitäten angehalten werden diese auch fortgesetzt.

**Parameter:**

ProcessId (STRING): ID des Prozesses

UserId (STRING): ID des Benutzers, der die Aktion ausführt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

State (INT): [Status des Prozesses](#) nach dem Fortsetzen

**Siehe auch:**

[wfm.AdminSuspendProcess](#)

## wfm.AdminRollbackProcess

**Beschreibung:**

Dieser Job setzt einen Prozess auf die angebenene Aktivität zurück und liefert alle Aktivitäten des Prozesses, die dadurch gelöscht werden. Über den Parameter 'ExecutionPoint' wird der Zurücksetzpunkt an der Aktivität selbst bestimmt. Man kann auf die Aktivität zurücksetzen, bevor sie erstellt wird oder beendet ist. Bevor ein Prozess zurückgesetzt werden kann, muss der Prozess über den Job [wfm.AdminSuspendProcess](#) angehalten werden.

**Parameter:**

RActivityId (STRING): InstanzId der Aktivität zu der der Prozess zurückgesetzt werden soll

ExecutionPoint (INT): Startpunkt an der Aktivität

§ 100 = Aktivität ist erstellt

§ 200 = Aktivität ist beendet

DoRollback (INT): 0 = Zurücksetzen wird nicht durchgeführt; 1 = Prozess wird zurückgesetzt  
(RunningActivities wird bei beiden Optionen gefüllt)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

RunningActivities (BASE64): enthält alle Aktivitäten die nach dem Zurücksetzen gelöscht wurden (DoRollback = 1) bzw. würden (DoRollback = 0)

**Beispiel:****Aufbau von RunningActivities**

```
<RunningActivities>
<RunningActivity Id=" " RActivityId=" " Name=" " State=" " CreationTime=" "/>
<RunningActivity Id=" " RActivityId=" " Name=" " State=" " CreationTime=" "/>
</RunningActivities>
```

**Hinweis:****Genauere Beschreibung von RunningActivities****§ RunningActivity**

- § Id (STRING): ID der Aktivität aus dem Workflowmodell
- § RactivityId (STRING): InstanzId der Aktivität
- § Name (STRING): Name der Aktivität
- § State (LONG): Status der Aktivität
- § CreationTime (LONG): Erstellzeitpunkt der Aktivität (Zeitstempel)

**Siehe auch:**

[wfm.AdminSuspendProcess](#)

**wfm.AdminSaveActivityVariables****Beschreibung:**

Dieser Job speichert Workflowvariablen zu einer angegebenen Instanz-ID einer Aktivität in der Datenbank.

**Parameter:**

UserId (STRING): ID des Benutzers, der die Änderung durchführt

RActivityId (STRING): Instanz-ID einer Aktivität

DataFields (BASE64): Liste mit Informationen zu Variablen im XML-Format

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Beispiel:****Aufbau von DataFields**

```
<DataFields>
<DataField Id=" " ><![CDATA[]]></DataField>
<DataField Id=" " ><![CDATA[]]></DataField>
</DataFields>
```

**Hinweis:****Genauere Beschreibung von DataFields****§ DataField: Struktur die Informationen zu einer Workflowvariable kapselt**

- § Id (STRING): ID der Variablen

§ CData: Wert der Variablen

**Siehe auch:**

[wfm.AdminGetProcessActivities](#) , [wfm.AdminGetActivityVariables](#)

## wfm.AdminSaveReportConfig

**Beschreibung:**

Dieser Job speichert ('insert' oder 'update') eine Reportkonfiguration.

**Parameter:**

UserId (STRING): ID des Benutzers, der die Änderung durchführt

ConfigId (STRING): ID einer Konfiguration. Wenn diese dem Server noch nicht bekannt ist, wird eine neue Konfiguration angelegt, falls die ID schon existiert, wird die entsprechende Konfiguration verändert. Bei diesem Update werden nur ConfigName, FamilyIds und ConfigData geändert.

ConfigName (STRING): Name der Konfiguration

OrganisationId (STRING): ID der Organisation in der sich die Konfiguration befindet

ConfigType (LONG): Typ der Konfiguration

0 – Statistik Prozesse

1 – Prozessdetails

ConfigData (STRING): XML-Beschreibung der Konfiguration

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

SaveType (INT): Gibt an, wie gespeichert wurde

0 – Neu Einfügen / Insert

1 – Vorhandene ändern / Update

§ Falls SaveType = 0 gibt es weitere Ausgabeparameter:

§ CreationTime (INT): Zeitpunkt der Erzeugung

CreatorId (STRING): ID des Benutzers, der die Änderung durchgeführt hat

**Siehe auch:**

[wfm.AdminDeleteStatisticReportConfigs](#) , [Wfm.AdminGetStatisticReportConfigs](#)

## wfm.AdminSuspendActivity

**Beschreibung:**

Dieser Job hält eine Aktivität an. Die Aktivität kann nicht bearbeitet werden.

**Parameter:**

RActivityId (STRING): InstanzId der Aktivität

UserId (STRING): ID des Benutzers, der die Aktion ausführt

**Rückgabe:**



(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

State (LONG): Status der Aktivität vor dem Anhalten

- § 0x1 = Die Aktivität wurde initialisiert.
- § 0x2 = Die Aktivität wurde gestartet (z. B. wurden Variablen angelegt).
- § 0x4 = Das StartActivity-Event wurde ausgeführt
- § 0x8 = Das EndActivity-Event wurde ausgeführt.
- § 0x10 = Nur bei Schleifen: Die Schleifenbedingung wurde geprüft.
- § 0x20 = Nur bei Schleifen: Der Schleifenkörper wird abgearbeitet.
- § 0x40 = Nur bei Arbeitsschritten: Der Arbeitsschritt liegt in den Eingangskörben bereit.
- § 0x80 = Nur bei Arbeitsschritten: Der Arbeitsschritt ist personalisiert.
- § 0x100 = Warten auf den Ablauf einer Sperrfrist.
- § 0x400 = Die Aktivität wurde ausgeführt, z. B. wurde ein Arbeitsschritt weitergeleitet oder eine Schleifenaktivität komplett abgearbeitet.
- § 0x800 = Nachfolgende Aktivitäten wurden berechnet und eventuell auch schon erstellt.
- § 0x1000 = Die Aktivität ist beendet, aber es wurden noch keine Folgeaktivitäten angestoßen.
- § 0x2000 = Die Aktivität wurde durch einen Benutzer angehalten.
- § 0x4000 = Die Aktivität ist beendet.
- § 0x8000 = Nur bei Multi-Instanz Aktivitäten: Die Aktivität wurde erstellt.
- § 0x10000 = Nur bei Ad-hoc-Aktivitäten: Die Ad-hoc-Aktivität wurde erstellt.
- § 0x20000 = Aktivität abgebrochen
- § 0x40000 = Nur bei Ad-hoc-Aktivitäten: Die Ad-hoc-Aktivität wird ausgeführt.
- § 0x10000000 = Die Aktivität wurde durch das System aufgrund eines Fehlers angehalten.

**Siehe auch:**

[wfm.AdminResumeActivity](#)

[wfm.AdminSuspendProcess](#)

### **Beschreibung:**

Dieser Job hält einen Prozess an. Es können keine Aktivitäten des Prozesses mehr bearbeitet werden.

### **Parameter:**

ProcessId (STRING): ID des Prozesses

UserId (STRING): ID des Benutzers, der die Aktion ausführt

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### **Rückgabewerte:**

State (INT): Status des Prozesses, vor dem Anhalten

- § 0 – neutraler Wert
- § 1 – Prozess ist initialisiert
- § 2 – Prozess läuft
- § 3 – Prozess wurde angehalten
- § 4 – Prozess ist aktiv
- § 5 – Prozess ist terminiert
- § 6 – Prozess ist beendet
- § 7 – Prozess wurde vom System angehalten (Fehler ist aufgetreten)

**Siehe auch:**

[wfm.AdminResumeProcess](#)

## wfm.AdminTerminateActivity

### **Beschreibung:**

Dieser Job bricht eine Aktivität ab. Bisher können nur Multi-Instanzen abgebrochen werden. Handelt es sich hierbei um die letzte Instanz einer Multi-Instanz-Aktivität, so wird der Prozess fortgesetzt.

### **Parameter:**

RActivityId (STRING): ID der Aktivität

UserId (STRING): ID des Benutzers, der die Aktion ausführt

### **Rückgabe:**

keiner

## wfm.AdminTerminateProcess

### **Beschreibung:**

Dieser Job bricht einen Prozess ab. Der Prozess wird aus den Laufzeitabellen des Workflows entfernt, bleibt in der Historie jedoch erhalten und ist als abgebrochen gekennzeichnet.

### **Parameter:**

ProcessId (STRING): ID des Prozesses

UserId (STRING): ID des Benutzers, der die Aktion ausführt

### **Rückgabe:**

keiner

## wfm.AdminGetLockInfo

### **Beschreibung:**

Dieser Job liefert Informationen zu Workflow-Datenbanktabellen die vom System gelockt werden.

### **Parameter:**

OrganisationsId (STRING): ID der Organisation

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

LockItems (BASE64): Informationen zu Workflow-Datenbanktabellen

**Beispiel:****Aufbau von LockItems**

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<LockItems>
<LockItem Id="" Name="" State="" LockTime="" ServerId="" ThreadId="" />
<LockItem Id="" Name="" State="" LockTime="" ServerId="" ThreadId="" />
</LockItems>
```

**Hinweis:**

Genauere Beschreibung von LockItems

- § Id (LONG): ID der Datenbank-Tabelle
- § Name (STRING): Name der Datenbank-Tabelle
- § State (LONG): 1- Datenbank-Tabelle ist gelockt, ansonsten 0
- § LockTime (LONG): Zeitstempel (wann wurde die Tabelle gelockt)
- § ServerId (LONG): ID des Servers der Tabelle gelockt hat
- § ThreadId (LONG): ID des Threads, der die Tabelle gelockt hat

**Siehe auch:**

[wfm.AdminReleaseLock](#)

## wfm.AdminGetWorkerqueue

**Beschreibung:**

Dieser Job liefert Informationen zu Elementen, die sich aktuell in der Workerqueue befinden.

**Parameter:**

OrganisationsId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

WorkerqueueItems (BASE64): Informationen zu Workerqueue-Elementen im XML-Format

**Beispiel:****Aufbau von WorkerqueueItems**

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<WorkerqueueItems>
<WorkerqueueItem Id="" ActivityName="" ProcessName="" State=""
TargetState="" CreationTime="" LockState="" LockTime="" ServerId="" />
<WorkerqueueItem Id="" ActivityName="" ProcessName="" State=""
TargetState="" CreationTime="" LockState="" LockTime="" ServerId="" />
</WorkerqueueItems>
```

**Hinweis:**

Genauere Beschreibung von WorkerqueueItems

- § Id (STRING): Instanz-ID der Aktivität in der Workerqueue

- § ActivityName (STRING): Name der Aktivität
- § ProcessName (STRING): Name des Prozesses, zu dem die Aktivität gehört
- § State (LONG): aktueller Status der Aktivität
- § TargetState (LONG): dieser Status soll nach Bearbeitung durch die Workerqueue erreicht werden
- § CreationTime (LONG): Zeitstempel (wann wurde Element in die Workerqueue aufgenommen )
- § LockState (LONG): 1- Workerqueue-Element ist gelockt, ansonsten 0
- § LockTime (LONG): Zeitstempel (wann wurde Workerqueue-Element gelockt)
- § ServerId (LONG): ID des Servers der Workerqueue-Element gelockt hat

**Siehe auch:**

[wfm.AdminReleaseLock](#)

## wfm.AdminGetProcessLocks

### Beschreibung:

Dieser Job liefert Informationen zu gelockten Workflow-Prozessen.

### Parameter:

OrganisationsId (STRING): ID der Organisation

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Processes (BASE64): Informationen zu gelockten Workflow-Prozessen im XML-Format

### Beispiel:

Aufbau von Processes

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<Processes>
<Process Id="" Name="" State="" LockTime="" ServerId="" ThreadId="" />
<Process Id="" Name="" State="" LockTime="" ServerId="" ThreadId="" />
</LockItems>
```

### Hinweis:

Genauere Beschreibung von Processes

- § Id (STRING): ID des Prozesses
- § Name (STRING): Name des Prozesses
- § LockTime (LONG): Zeitstempel (wann wurde der Prozess gelockt)
- § ServerId (LONG): ID des Servers, der den Prozess gelockt hat
- § ThreadId (LONG): ID des Threads, der den Prozess gelockt hat

**Siehe auch:**

[wfm.AdminReleaseLock](#)

## wfm.AdminReleaseLock

### Beschreibung:

Dieser Job gibt ein Datenbanklock des angegebenen Typs wieder frei.

**Parameter:**

Locktype (LONG): gibt den Locktyp an

- § 1 - gelockte Prozesse werden freigegeben (LockId muss dann entsprechende ProzessID enthalten)
- § 2 - gelocktes Workerqueue-Element wird freigegeben (LockId muss dann entsprechende WorkerqueueItemId enthalten)
- § 3 - gelockte Datenbanktabelle wird freigegeben (LockId muss dann entsprechende TabellenId enthalten)

LockId (STRING): siehe Locktype

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[wfm.AdminGetWorkerqueue](#) , [wfm.AdminGetProcessLocks](#) ,  
[wfm.AdminGetLockInfo](#)

## wfm.AdminGetProcessReport

**Beschreibung:**

Dieser Job liefert Informationen zu den angegebenen Workflowprozessen. Nicht zu verwechseln mit den statistischen Prozessreports!

**Parameter:**

Processes (STRING): kommaseparierte Liste von ProzessIDs

File (LONG): 1 = Informationen zur WF-Akte werden mitgeliefert, ansonsten 0

GlobalDataFields (LONG): 1 = Informationen zu globalen Variablen werden mitgeliefert, ansonsten 0

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (BASE64): Informationen zu Workflow-Prozessen im XML-Format

**Beispiel:****Aufbau von Processes**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<Processes>
  <Process>
    <Id>B216B6ACC46B4A33AFB9D6852D928A33</Id>
    <Name>Globale Variablen Test - Paralleler Ablauf 2</Name>
    <CreationTime>1143128816</CreationTime>
    <EndTime>0</EndTime>
    <UserName>Peter Mustermann</UserName>
    <DataFields>
      <DataField>
        <Id>79FBB5E2F38B434DBF6C7507614CF740</Id>
        <Name>TextVar02</Name>
        <Value><WFVar><String>Wert 1</String><Types></Types></WFVar></Value>
      </DataField>
    </DataFields>
  </Process>
</Processes>
```

```

<Docs>
<Doc>
<Id>194</Id>
<Type>196608</Type>
</Doc>
</Docs>
</Process>
</Processes>

```

**Hinweis:**

Genauere Beschreibung von Processes

- § Id (STRING): ID des Prozesses
- § Name (STRING): Name des Prozesses
- § CreationTime (LONG): Zeitstempel (wann wurde der Prozess erstellt)
- § EndTime (LONG): Zeitstempel (wann wurde der Prozess beendet)
- § UserName (STRING): Name des Prozesserstellers
- § DataFields: Informationen zu globalen Variablen
  - § Id (STRING): ID der Variable
  - § Name (STRING): Name der Variable
  - § Value: Aufbau und Wert der Variable
- § Docs: Objekte der Workflow-Akte
  - § Id (STRING): ID des ECM-Objekts
  - § Type (LONG): Objekttyp

## wfm.AdminGetStatisticReportConfigs

**Beschreibung:**

Dieser Job liefert Konfigurationen für Prozessreports (Statistiken etc.).

**Parameter:**

OrganisationId (STRING): ID der Organisation in der die angefragten Reportkonfigurationen liegen

CreatorId (STRING): ID des Erstellers einer Konfiguration. Darf leer sein. Falls die ID gesetzt ist, werden nur Konfigurationen mit dieser CreatorId zurückgeliefert.

UserId (STRING): ID des aufrufenden Benutzers. Es werden nur Konfigurationen zurückgeliefert, für die der Benutzer Rechte besitzt.

ConfigTypes (STRING): Kommaseparierte Liste von Konfigurationstypen. Darf leer sein. Falls der Parameter gesetzt ist, werden nur Konfigurationen zurückgeliefert, die einen der angegebenen Typen haben.

0 – Statistik Prozesse

1 – Prozessdetails

FamilyIds (STRING): Kommaseparierte Liste mit Ids von Workflowfamilien. Darf leer sein. Falls der Parameter nicht leer ist, werden nur Konfigurationen zurückgeliefert, die mindestens einer der angegebenen Familien zugeordnet sind.

RespectRights (LONG): Flag gibt an, ob nur Reportkonfigurationen zurückgeliefert werden, deren Reports der Benutzer anfordern und ansehen darf

0 – Benutzer erhält Reportkonfiguration unabhängig von Rechten

1 – Benutzer erhält Reportkonfiguration den Rechten entsprechend

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

ReportConfigs (BASE64): Informationen zu den angefragten Reportkonfigurationen im XML-Format

### Beispiel:

#### Aufbau von ReportConfigs

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ReportConfigs>
<ReportConfig ConfigId="" ConfigName="" CreationTime="" CreatorId="" CreatorName=""
ConfigType="">
<ConfigData><![CDATA[]]></ConfigData>
</ReportConfig>
<ReportConfig ConfigId="" ConfigName="" CreationTime="" CreatorId="" CreatorName=""
ConfigType="">
<ConfigData><![CDATA[]]></ConfigData>
</ReportConfig>
</ReportConfigs>
```

### Hinweis:

#### Genauere Beschreibung von ReportConfig

§ ConfigId (BSTR): ID der Konfiguration

§ ConfigName (STRING): Name der Konfiguration

§ CreationTime (LONG): Zeitstempel (wann wurde die Konfiguration erstellt)

§ CreatorId (STRING): Benutzer-ID des Erstellers der Konfiguration

§ CreatorName (STRING): Name des dem Benutzer entsprechenden Personenobjekts

§ ConfigType (LONG): Typ der Konfiguration

0 – Statistik Prozesse

1 – Prozessdetails

§ ConfigData CDATA Sektion (XML): Konfiguration als XML Beschreibung

§ **Siehe auch:**

§ [wfm.AdminDeleteStatisticReportConfigs](#) , [wfm.AdminSaveStatisticReportConfig](#)

## wfm.AdminGetStatisticReportData

### Beschreibung:

Dieser Job liefert die Daten zu einem bereits fertig generierten Statistikreport.

### Parameter:

ReportId (STRING): ID des Reports

UserId (STRING): ID des aufrufenden Benutzers. Es werden nur Daten geliefert, wenn der Benutzer die entsprechenden Rechte besitzt.

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Dateiliste: Name und Pfad der XML Datei mit dem Daten des Statistikreports

## wfm.AdminGetStatisticReports

**Beschreibung:**

Liefert die vorhandenen Reports zu einer Statistikreportkonfiguration.

**Parameter:**

OrganisationId (STRING): ID der Organisation

ConfigId (STRING): ID der Statistikreportkonfiguration

UserId (STRING): ID des aufrufenden Benutzers. Es werden nur Daten geliefert, wenn der Benutzer die entsprechenden Rechte besitzt.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Reports (BASE64): Informationen zu den angefragten Reports im XML-Format

**Beispiel:****Aufbau von Reports**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<Reports>
<Report ReportId="" CreationTime="" CreatorId="" CreatorName="" State="" />
<Report ReportId="" CreationTime="" CreatorId="" CreatorName="" State="" />
</Reports>
```

**Hinweis:****Genauere Beschreibung von Report**

§ ReportId (STRING): ID des Reports

§ CreationTime (LONG): Zeitstempel (wann wurde der Report erstellt), Bei State = 0 oder 1 ist dies der Zeitpunkt zu dem der Report angefragt wurde, sonst ist dies der Zeitpunkt bei dem mit der Erstellung des Reports begonnen wurde

§ CreatorId (STRING): BenutzerID des Erstellers des Reports

§ CreatorName (STRING): Name des dem Benutzer entsprechenden Personenobjekts

§ State (LONG): Status des Reports

0 – Report ist angefragt

§ 1 – Report momentan wird erzeugt

§ 2 – Report ist fertig erstellt

## Historienverwaltung

§ [wfm.GetHistActivitiesByProcess](#)

§ [wfm.GetHistEntries](#)

§ [wfm.GetHistProcessList](#)



- § [wfm.GetHistTimerEntries](#)
- § [wfm.GetHistTimersByProcess](#)
- § [wfm.GetHistVariablesByHistEntry](#)
- § [wfm.GetHistWorkflowList](#)
- § [wfm.GetHistWorkItemRelActivitiesByProcess](#)
- § [wfm.GetHistWorkItemRelEntriesByActivity](#)
- § [wfm.GetHistWorkItemRelUsersByProcess](#)
- § [wfm.GetHistWorkItemRelEntriesByUser](#)

## wfm.GetHistActivitiesByProcess

### Beschreibung:

Dieser Job ermittelt zu einem historischen Prozess alle Aktivitäten.

### Parameter:

ProcessId (STRING): historische ID des Prozesses

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

Activities (BASE64): enthält die angeforderten Aktivitäten im XML-Format

### Beispiel:

#### Aufbau von Activities

```
<Activities>
<Activity RActivityId="3A25AFD5CBD9B246" Name="Schritt1" EntryNr="10"/>
<Activity RActivityId="51BBA8B5AAA1D471" Name="StartActivity" EntryNr="5"/>
<Activity RActivityId="1A8E439C131AADF2B" Name="EndActivity" EntryNr="21"/>
</Activities>
```

### Hinweis:

#### Genauere Beschreibung von Activity

§ Activity: Struktur, die eine Aktivitäteninstanz charakterisiert

- § RActivityId (STRING): InstanzID der Aktivität
- § Name (STRING): Name der Aktivität
- § EntryNr (LONG): spiegelt den zeitlichen Ablauf wieder

### Siehe auch:

[wfm.GetHistProcessList](#) , [wfm.GetHistEntries](#)

## wfm.GetHistEntries

### Beschreibung:

Dieser Job ermittelt zu einer historischen Aktivität oder einem historischen Prozess alle durchgeführten Aktionen. Beim Aufruf des Jobs ist darauf zu achten, dass nur einer der beiden Parameter gesetzt ist.

**Parameter:**

ProcessId (STRING): hist. ID des Prozesses

RActivityId (STRING): hist. Instanz-ID der Aktivität

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

HistoryEntries (BASE64): enthält die angeforderten Aktionen im XML-Format

**Beispiel:****Aufbau von HistoryEntries**

```
<HistoryEntries>
<HistoryEntry HistId=".." EntryNr=".." ProcessId=".." RActivityId=".."
HistType=".." Time=".." OrganisationId=".." UserId=".." UserName=".."
ServerId=".." ServerName=".." />
<HistoryEntry HistId=".." EntryNr=".." ProcessId=".." RActivityId=".."
HistType=".." Time=".." OrganisationId=".." UserId=".." UserName=".."
ServerId=".." ServerName=".." />
</HistoryEntries>
```

**Hinweis:****Genauere Beschreibung von HistoryEntries**

- § HistId (STRING): ID des hist. Eintrags
- § EntryNr (LONG): Eintragsnummer, spiegelt den zeitlichen Ablauf wieder
- § ProcessId (STRING): ID des Prozesses
- § RActivityId (STRING): Instanz-ID der Aktivität (nur bei Aufruf über Parameter RActivityId gesetzt)
- § HistType (LONG): Typ des hist. Eintrags
  - § 1 = PREPAREPROCESS
  - § 2 = STARTPROCESS
  - § 3 = ENDPROCESS
  - § 4 = PREPAREACTIVITY
  - § 5 = STARTACTIVITY
  - § 6 = ENDACTIVITY
  - § 7 = COPYACTIVITYVARIABLES
  - § 8 = HALTACTIVITY
  - § 9 = REACTIVATEACTIVITY
  - § 10 = STARTWORKITEM
  - § 11 = PERSONALIZED
  - § 12 = DEPERSONALIZED
  - § 13 = SAVEWORKITEM
  - § 14 = ENDWORKITEM
  - § 15 = CREATETIMER

- § 16 = CANCELTIMER
- § 17 = REMIND
- § 18 = DELAYED
- § 19 = STARTSCRIPT
- § 20 = ENDSRIPT
- § 21 = SYSSUSPEND
- § 22 = SETACTIVITYPERFORMER
- § Time (LONG)
- § OrganisationId (STRING)
- § UserId (STRING)
- § UserName (STRING)
- § ServerId (STRING)
- § ServerName (STRING)

**Siehe auch:**

[wfm.GetHistActivitiesByProcess](#) , [wfm.GetHistProcessList](#) , [wfm.GetHistVariablesByHistEntry](#)

## wfm.GetHistProcessList

**Beschreibung:**

Dieser Job ermittelt zu einem historischen Workflowmodell alle Prozesse.

**Parameter:**

OrganisationId (STRING): ID der Organisation

HistWorkflowId (STRING): historische ID des Modells

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Processes (BASE64): enthält die angeforderten Prozesse im XML-Format

**Beispiel:****Aufbau von Processes**

```
<Processes>
<Process Id=".." Name=".." FinalSubject=".." UserName=".." CreationTime=".."
EndTime=".." />
<Process Id=".." Name=".." FinalSubject=".." UserName=".." CreationTime=".."
EndTime=".." />
</Processes>
```

**Hinweis:**

Genauere Beschreibung von Processes

- § Id (STRING): ID des Prozesses
- § Name (STRING): Name des Prozesses
- § FinalSubject (STRING): Letzter Betreff des Prozesses

§ UserName (STRING): Name des Prozesserstellers

§ CreationTime (STRING): Startzeit des Prozesses

§ EndTime (STRING): Endzeit des Prozesses

**Siehe auch:**

[wfm.GetHistWorkflowList](#) , [wfm.GetHistActivitiesByProcess](#) , [wfm.GetHistEntries](#) , [wfm.GetHistTimersByProcess](#)

## wfm.GetHistTimerEntries

**Beschreibung:**

Dieser Job ermittelt zu einer Mahn-/Sperrfrist alle Aktionen (historische Einträge).

**Parameter:**

TimerId (STRING): ID des Timers

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

TimerEntries (BASE64): enthält die angeforderten Aktionen im XML-Format

**Beispiel:**

Aufbau von TimerEntries:

```
<TimerEntries>
<TimerEntry HistId=".." EntryNr=".." ProcessId=".." RActivityId=".."
HistType=".." Time=".." OrganisationId=".." UserId=".." UserName=".."
ServerId=".." ServerName=".." />
</TimerEntries>
```

**Hinweis:**

Genauere Beschreibung von TimerEntries

§ HistId (STRING): hist. ID

§ EntryNr (LONG): Nummer des Eintrags (spiegelt den zeitlichen Ablauf wieder)

§ ProcessId (STRING): ID des Prozesses

§ RActivityId (STRING): Instanz-ID der Aktivität

§ HistType(LONG): Typ des hist. Eintrags

§ 15 = CREATETIMER

§ 16 = CANCELTIMER

§ 17 = REMIND

§ 18 = DELAYED

§ Time (LONG): Zeitpunkt der Eintragserstellung

§ OrganisationId (STRING): ID der Organisation

§ UserId (STRING): ID des Benutzers

§ UserName (STRING): Name des Benutzers, der Aktivität personalisiert hat

§ ServerId (STRING): ID des Server

§ ServerName (STRING): Name des Servers

**Siehe auch:**

[wfm.GetHistTimersByProcess](#)

## wfm.GetHistTimersByProcess

**Beschreibung:**

Dieser Job ermittelt zu einem historischen Prozess alle Mahn-/Sperrfristen.

**Parameter:**

ProcessId (STRING): hist .ID des Prozesses

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Timers (BASE64): enthält die angeforderten Mahn-/Sperrfristen im XML-Format

**Beispiel:**

Aufbau von Timers

```
<Timers>
<Timer TimerId=".." ProcessId=".." FromActivityId=".." FromActivityName=".."
ToActivityId=".." ToActivityName=".." TimerType=".." DestinationType=".."
DestinationTime=".." LoopType=".." />
</Timers>
```

**Hinweis:**

Genauere Beschreibung von Timers

§ TimerId (STRING): ID der Mahn-/Sperrfrist

§ ProcessId (STRING): ID des Prozesses

§ FromActivityId (STRING): ID der Aktivität (ab dieser ist die Frist gültig)

§ FromActivityName (STRING): Name der Aktivität (ab dieser ist die Frist gültig)

§ ToActivityId (STRING): ID der Aktivität (bis zu dieser ist die Frist gültig)

§ ToActivityName (STRING): Name der Aktivität (bis zu dieser ist die Frist gültig)

§ TimerType (LONG): Typ der Frist

§ 0 = Sperrfrist

§ 1 = Mahnfrist

§ DestinationType (LONG):

§ 0 = TIMER\_REFERENCES\_START\_OF\_ACTIVITY

§ 1 = TIMER\_REFERENCES\_END\_OF\_ACTIVITY

§ DestinationTime (LONG): gibt an wann das Ziel erreicht werden soll

§ LoopType (LONG): wird immer 0 geliefert

**Siehe auch:**

[wfm.GetHistProcessList](#) , [wfm.GetHistTimerEntries](#)

## wfm.GetHistVariablesByHistEntry

**Beschreibung:**

Dieser Job ermittelt die Workflow-Variablen zu einem History-Eintrag.

**Parameter:**

HistId (STRING): ID des History-Eintrags

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

DataFields (BASE64): enthält die angeforderten historischen Variablen im XML-Format

**Beispiel:**

Aufbau von DataFields

```
<DataFields>
<DataField Id="123.." Name="strTest">
<![CDATA[Ich bin ein String]]>
</DataField>
<DataField Id=" " Name=" ">
<![CDATA[]]>
</DataField>
</DataFields>
```

**Hinweis:**

Genauere Beschreibung von DataFields

§ Id (STRING): ID der Variablen

§ Name (STRING): Name der Variablen

§ CDATA: Wert und Aufbau der Variablen

**Siehe auch:**

[wfm.GetHistEntries](#)

## wfm.GetHistWorkflowList

**Beschreibung:**

Dieser Job ermittelt alle Workflowmodelle in der Historienverwaltung und die Anzahl, der vom Worklowmodell gestarteten Prozesse.

**Parameter:**

OrganisationId (STRING): ID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Workflows (BASE64): enthält die angeforderten Aktionen im XML-Format

**Beispiel:**

Aufbau von Workflows

```
<Workflows>
<Workflow Id="18268F5E1F532" HistWorkfowId="04DC0D15A62A4" Name="test"
ProcessCount="3" Version="70" FamilyId="0774DC0D15A62A4"
FamilyName="testfam"/>
<Workflow Id="39418268F5532" HistWorkfowId="0BFB9FF2D225E7" Name="test"
ProcessCount="1" Version="83" FamilyId="T0774DC0D15A62"
FamilyName="testfamilie"/>
</Workflows>
```

**Hinweis:**

Genauere Beschreibung von Workflows

- § Id (STRING): ID des Workflowmodells
- § HistWorkfowId (STRING): historische ID des Workflowmodells
- § Name (STRING): Name des Workflowmodells
- § ProcessCount (LONG): gibt an, wie oft das Workflowmodell gestartet wurde
- § Version (LONG): Versionsnummer des Workflowmodells
- § FamilyId (STRING): GUID der dazugehörigen Workflowfamilie
- § FamilyId Name(STRING): GUID der dazugehörigen Workflowfamilie

**Siehe auch:**

[wfm.GetOrganisations](#) , [wfm.GetHistProcessList](#)

## wfm.GetHistWorkItemRelActivitiesByProcess

**Beschreibung:**

Dieser Job ermittelt zu einem historischen Prozess alle Aktivitäten, die in einen Eingangskorb gestellt wurden, also von einem Benutzer bearbeitet wurden.

**Parameter:**

ProcessId (STRING): hist. ID des Prozesses

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Activities (BASE64): enthält die angeforderten Aktivitäten im XML-Format

**Beispiel:**

Aufbau von Activities

```
<Activities>
<Activity RActivityId="2D0131F8C0" Name="Anforderung prüfen" EntryNr="17"/>
<Activity RActivityId="32BAE" Name="Anforderung weiterleiten" EntryNr="9"/>
</Activities>
```

**Hinweis:**

Genauere Beschreibung von Activities

- § RActivityId (STRING): Instanz-ID der Aktivität
- § Name (STRING): Name der Aktivität
- § EntryNr (LONG): Eintragsnummer, spiegelt den zeitlichen Ablauf wieder

**Siehe auch:**

[wfm.GetHistProcessList](#) , [wfm.GetHistWorkItemRelEntriesByActivity](#)

## wfm.GetHistWorkItemRelEntriesByActivity

**Beschreibung:**

Dieser Job ermittelt alle historische Einträge für die angegebenen Arbeitsschritt.

**Parameter:**

RActivityId (STRING): Instanz-ID der Aktivität

**Rückgabewerte:**

HistWorkItemRelEntries (BASE64): enthält die angeforderten Einträge im XML-Format

**Beispiel:**

Aufbau von HistWorkItemRelEntries

```
<HistWorkItemRelEntries>
<HistWorkItemRelEntry RActivityId="" ActivityName="" OrganisationId=""
UserId="" UserName="" Reason="" HistType="" HistId="" EntryNr="" Time="" />
</HistWorkItemRelEntries>
```

**Hinweis:**

Genauere Beschreibung von HistWorkItemRelEntries

- § RActivityId (STRING): Instanz-ID der Aktivität
- § ActivityName (STRING): Name der Aktivität
- § OrganisationId (STRING): ID der Organisation
- § UserId (STRING): ID des Benutzers, der die Aktivität im Eingangskorb hatte
- § UserName (STRING): Name des Benutzers
- § Reason (LONG): Grund, warum hatte der Benutzer die Aktivität im Eingangskorb
  - § 0 = als Teilnehmer eingerichtet
  - § 1 = wurde dem Benutzer aufgrund einer Mahnfrist zugeordnet
  - § 2 = durch Administrator zugewiesen (HistType = 11) / entfernt (HistType = 12)
  - § 3 = durch ein Skript zugewiesen (HistType = 11) / entfernt (HistType = 12)
- § HistType (LONG): Typ des hist. Eintrags
  - § 10 = STARTWORKITEM
  - § 11 = PERSONALIZED
  - § 12 = DEPERSONALIZED
- § HistId (STRING): ID der hist. Eintrags
- § EntryNr (LONG): spiegelt den zeitlichen Ablauf wieder
- § Time (LONG): Erstellzeit des Eintrags

**Siehe auch:**

[wfm.GetHistWorkItemRelActivitiesByProcess](#)



## wfm.GetHistWorkItemRelUsersByProcess

**Beschreibung:**

Dieser Job ermittelt alle Benutzer bzw. Rollen für die Arbeitsschritte des angegebenen historischen Prozesses in den Eingangskorb gestellt wurden.

**Parameter:**

ProcessId (STRING): hist. ID des Prozesses

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Users (BASE64): enthält die angeforderten Benutzer im XML-Format

**Beispiel:**

Aufbau von Users

```
<Users>
<User UserId=".." UserName=".." />
<User UserId=".." UserName=".." />
</Users>
```

**Hinweis:**

Genauere Beschreibung von Workflows

§ UserId (STRING): ID d. Benutzers/Rolle

§ UserName (STRING): Name d. Benutzers/Rolle

**Siehe auch:**

[wfm.GetHistProcessList](#)

## wfm.GetHistWorkItemRelEntriesByUser

**Beschreibung:**

Dieser Job ermittelt alle vom angegebenen Benutzer bearbeiteten Aktivitäten und die dazugehörigen historischen Einträge des angegebenen historischen Prozesses.

**Parameter:**

ProcessId (STRING): hist. ID des Prozesses

UserId (STRING): ID des Benutzers

**Rückgabewerte:**

HistWorkItemRelEntries (BASE64): enthält die angeforderten Informationen im XML-Format

**Beispiel:**

Aufbau von HistWorkItemRelEntries

```
<HistWorkItemRelEntries>
<HistWorkItemRelEntry RActivityId="" ActivityName="" OrganisationId=""
UserId="" UserName="" Reason="" HistType="" HistId="" EntryNr="" Time="" />
</HistWorkItemRelEntries>
```

**Hinweis:**

## Genauere Beschreibung von HistWorkItemRelEntries

- § RActivityId (STRING): Instanz-ID der Aktivität
- § ActivityName (STRING): Name der Aktivität
- § OrganisationId (STRING): ID der Organisation
- § UserId (STRING): ID des Benutzers, der die Aktivität im Eingangskorb hatte
- § UserName (STRING): Name des Benutzers
- § Reason (LONG): Grund, warum hatte der Benutzer die Aktivität im Eingangskorb
  - § 0 = als Teilnehmer eingerichtet
  - § 1 = wurde dem Benutzer aufgrund einer Mahnfrist zugeordnet
  - § 2 = durch Administrator zugewiesen (HistType = 11) / entfernt (HistType = 12)
  - § 3 = durch ein Skript zugewiesen (HistType = 11) / entfernt (HistType = 12)
- § HistType (LONG): Typ des hist. Eintrags
  - § 10 = STARTWORKITEM
  - § 11 = PERSONALIZED
  - § 12 = DEPERSONALIZED
- § HistId (STRING): ID der hist. Eintrags
- § EntryNr (LONG): spiegelt den zeitlichen Ablauf wieder
- § Time (LONG): Erstellzeit des Eintrags

### Siehe auch:

[wfm.GetHistProcessList](#)

## Sonstige Jobs

- § [wfm.ConvertExportFile](#)
- § [wfm.DeleteSysClienttypes](#)
- § [wfm.Export](#)
- § [wfm.GetSysClienttypes](#)
- § [wfm.GetVersionInfo](#)
- § [wfm.GetWFMIInfo](#)
- § [wfm.Import](#)
- § [wfm.InsertSysClienttypes](#)
- § [wfm.GetProjectList](#)
- § [AdhocConfigTemplate](#)

## wfm.ConvertExportFile

### Beschreibung:

Dieser Job konvertiert eine Exportdatei in das aktuelle Format (z. B. von 4.20 nach 4.50). Die Exportdatei muss dem Job beigelegt werden. Die Ergebnisdatei wird dann mit der Antwort übertragen.

### Parameter:

Dateiliste: Name und Pfad Workflow-Export-Datei, die konvertiert werden soll

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Dateiliste: Name und Pfad der konvertierten Workflow-Export-Datei

## wfm.Export

**Beschreibung:**

Dieser Job exportiert Organisationsstruktur, Workflow-Projekte und/oder Workflows.

**Parameter:**

OrganisationId (STRING): ID der Organisation aus der exportiert werden soll

OrgDoExport (INT): Flag gibt an, ob auch Organisationsdaten exportiert werden sollen (1=Ja, 0=Nein)

WFProjectTree (BASE64): Struktur gibt, an welche Projekte und Workflowmodelle exportiert werden sollen (XML-Format)

ReportConfigDoExport (INT): Flag gibt an, ob Reportkonfigurationen der Organisation exportiert werden sollen

ReportConfigIds (String): Kommaseparierte Liste zum von zu exportierenden

Reportkonfigurationen. Ist Parameter ReportConfigDoExport = 1 und diese Liste leer, werden alle Konfigurationen der Organisation exportiert.

AdhocRoutingListTemplateDoExport (INT): Flag gibt an, ob Lauflistenvorlagen der Organisation exportiert werden sollen

AdhocRoutingListTemplateIds (String): Kommaseparierte Id-Liste von zu exportierenden Lauflistenvorlagen. Ist Parameter AdhocRoutingListTemplateDoExport = 1 und diese Liste leer, werden alle Lauflistenvorlagen der Organisation exportiert.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Dateiliste: Name und Pfad der Exportdatei

**Beispiel:****Aufbau von WFProjectTree**

```
<WFProjectTree>
<WorkflowProject Id=" " CompleteExport=0>
<WorkflowProject Id=" ">
<WorkflowProject Id=" " CompleteExport=0>
</Workflow Id=" ">
</WorkflowProject>
</WorkflowProject Id=" " CompleteExport=1>
</WorkflowProject>
</WorkflowProject Id=" " CompleteExport=1>
</WorkflowProject>
```

```
</WorkflowProject Id=" " CompleteExport=1>  
</Workflow Id=" ">  
</Workflow Id=" ">  
</WFProjectTree>
```

**Hinweis:****Genauere Beschreibung von WFProjectTree**

- § WorkflowProject: Struktur (ggf. geschachtelt) , die angibt welches WorkflowProjekt (wobei dieses auch eine WorkflowFamilie sein darf) exportiert werden soll
  - § Id (STRING): ID des WorkflowProjekts
  - § CompleteExport (LONG): Flag gibt an, ob alle Projekte oder Workflows, die unter dem Projekt hängen, exportiert werden sollen. Falls es gesetzt ist, muss die Unterstruktur nicht weiter angegeben werden.
- § Workflow: Struktur, die angibt welcher Workflow (welche Workflows) exportiert werden soll
  - § Id (STRING): ID des Workflow

**Siehe auch:**

[wfm.Import](#)

## wfm.GetVersionInfo

**Beschreibung:**

Dieser Job liefert Informationen über die Version der Workflow-Engine.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

EngineVersion (INT): Gibt die Major Version der Engine an. z. B. 450

EngineSubVersion (INT): Gibt die Sub Version der Engine an. z. B. 3

## wfm.GetWFInfo

**Beschreibung:**

Dieser Job liefert Informationen zu einem Benutzer (z. B. Abwesenheit, WF-Benutzer-ID) über die DRT-Benutzer-ID.

**Parameter:**

Requests (BASE64): enthält eine Liste von Anfragen im XML-Format

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Requests (BASE64): enthält Antworten auf die Anfragen im XML-Format

**Beispiel:**

Aufbau des Eingabeparameters Requests

```
<Requests>
<Request UserId=" " RequestType=" " />
<Request UserId=" " RequestType=" " />
</Requests>
```

**Hinweis:**

Genauere Beschreibung des Eingabeparameters Requests

§ UserId (STRING): ID des DRT-Benutzers

§ RequestType (INT): Flag für die Art der Anfrage

§ 1 = es soll die ID der aktiven Organisation ermittelt werden

§ 2 = es soll ermittelt werden, ob der DRT-Benutzer in der aktiven Organisation enthalten ist

§ 3 = es soll die ID des WF-Benutzers zur DRT-Benutzer-ID ermittelt werden

§ 4 = es soll ermittelt werden, ob der angegebene Benutzer abwesend gemeldet ist

**Beispiel:**

Aufbau des Ausgabeparameter Requests

```
<Requests>
<Request UserId=" " RequestType=" " Value=" " />
<Request UserId=" " RequestType=" " Value=" " />
</Requests>
```

**Hinweis:**

Genauere Beschreibung des Ausgabeparameters Requests

§ UserId (STRING): ID des DRT-Benutzers

§ RequestType (INT): Flag für die Art der Anfrage

§ 1 = es soll die ID der aktiven Organisation ermittelt werden

§ 2 = es soll ermittelt werden, ob der DRT-Benutzer in der aktiven Organisation enthalten ist

§ 3 = es soll die ID des WF-Benutzers zur DRT-Benutzer-ID ermittelt werden

§ 4 = es soll ermittelt werden, ob der angegebene Benutzer abwesend gemeldet ist

§ Value (INT): Antwort auf die Anfrage

§ bei RequestType = 2: 1 = DRT-Benutzer in Organisation enthalten, ansonsten 0

§ bei RequestType = 4: 1 = DRT-Benutzer abwesend gemeldet, ansonsten 0

## wfm.Import

**Beschreibung:**

Dieser Job importiert eine Organisation. Dazu wird dem Job eine Datei beigelegt.

**Parameter:**

Eingabedatei: Name und Pfad der zu importierenden Datei

DoImportOrganisation (INT): gibt an, ob auch Organisationsdaten importiert werden sollen (1=Ja, 0=Nein)

DestOrganisationId (BASE64): Falls Organisationsdaten importiert werden sollen, wird hiermit die Ziel-Organisation spezifiziert. Parameter bleibt leer, falls eine neue Organisation angelegt werden soll.

WorkflowProjects (BASE64): gibt die zu importierenden Workflowprojekte im XML-Format an

Workflows (BASE64): gibt die zu importierenden Workflowmodelle im XML-Format an

ReportConfigs (BASE64): gibt die zu importierenden Reportkonfigurationen im XML-Format an

Templates (BASE64): gibt die zu importierenden Lauflistenvorlagen im XML-Format an

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Rückgabewerte:

ChangedOrganisations (STRING): kommaseparierte ID-Liste von veränderten Organisationen

### Beispiel:

#### Aufbau von WorkflowProjects

```
<WorkflowProjects>
</WorkflowProject Id="" DestPrjId="" DestOrgId="" Overwrite=""
CompleteImport="" OldParent="">
</WorkflowProject Id="" DestPrjId="" DestOrgId="" Overwrite=""
CompleteImport="" OldParent="">
</WorkflowProject Id="" DestPrjId="" DestOrgId="" Overwrite=""
CompleteImport="" OldParent="">
</WorkflowProjects>
```

### Hinweis:

#### Genauere Beschreibung von WorkflowProjects

- § Id (STRING): ID des Projekts im Import-File
- § DestPrjId (STRING): ID des ParentProjekts
- § DestOrgId (STRING): ID der Ziel-Organisation. Dieses Attribut wird nicht angegeben, falls beim Import eine neue Organisation erzeugt wird und das Projekt diese neue Organisation als Ziel hat.
- § Overwrite (INT): Flag gibt an, ob möglicherweise bereits vorhandenes Projekt mit gleicher ID überschrieben werden soll. Anderenfalls wird ein neues Projekt mit neuer ID erzeugt.
- § CompleteImport (INT): Flag gibt an, ob die gesamte Unterstruktur (Projekte/Workflows) des Projekts importiert werden sollen. Diese braucht dann nicht aufgeführt zu werden.
- § OldParent (INT): Flag wird nur interessant, falls das übergeordnete Workflowprojekte kopiert (also nicht überschrieben) wurden. Falls das Flag gesetzt ist, wird das Projekt in das alte WF-Projekt geschrieben (wobei wieder das Overwrite Flag beachtet wird), falls nicht, wird es in das neu erzeugte (kopierte) WF-Projekt geschrieben. Sollte das übergeordnete WF-Projekt überschrieben worden sein, soll das Flag immer gesetzt sein.

### Beispiel:

#### Aufbau von Workflows

```
<Workflows>
</Workflow Id="" DestFamId="" DestOrgId="" Overwrite="" OldFamily="">
</Workflow Id="" DestFamId="" DestOrgId="" Overwrite="" OldFamily="">
</Workflow Id="" DestFamId="" DestOrgId="" Overwrite="" OldFamily="">
</Workflows>
```

### Hinweis:

#### Genauere Beschreibung von Workflows

- § Id (STRING): ID des Workflow im Import-File
- § DestFamId (STRING): ID der Ziel-WFFamily

- § DestOrgId (STRING): ID der Ziel-Organisation. Dieses Attribut wird nicht angegeben, falls beim Import eine neue Organisation erzeugt wird (siehe Knotenbeschreibung Organisation) und das Modell diese neue Organisation als Ziel hat.
- § Overwrite (INT): Flag gibt an, ob möglicherweise bereits vorhandener Workflow mit gleicher ID überschrieben werden soll. Anderenfalls wird ein neuer Workflow mit neuer ID erzeugt.
- § OldFamily (INT): Flag wird nur interessant, falls die übergeordnete Workflowfamilie kopiert (also nicht überschrieben) wurde. Falls das Flag gesetzt ist, wird das Modell in die alte WF-Familie geschrieben (wobei wieder das Overwrite Flag beachtet wird), falls nicht, wird es in die neu erzeugte (kopierte) WF-Familie geschrieben. Sollte die übergeordnete WF-Familie überschrieben worden sein, soll das Flag immer gesetzt sein.

**Beispiel:****Aufbau von ReportConfigs**

```
<ReportConfigs>
<ReportConfig ConfigId=" " Overwrite=" " DestOrgId=" " />
<ReportConfig ConfigId=" " Overwrite=" " DestOrgId=" " />
<ReportConfig ConfigId=" " Overwrite=" " DestOrgId=" " />
</ReportConfigs >
```

**Hinweis:****Genauere Beschreibung von Workflows**

- § ConfigId (STRING): ID der Konfiguration im Import-File
- § DestOrgId (STRING): ID der Ziel-Organisation.
- § Overwrite (INT): Flag gibt an, ob möglicherweise bereits vorhandene Konfiguration mit gleicher ID überschrieben werden soll. Anderenfalls wird eine neue Konfiguration mit neuer ID erzeugt.

**Siehe auch:**

[wfm.Export](#)

**wfm.GetSysClienttypes****Beschreibung:**

Dieser Job liefert alle im System definierten Clienttypen.

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

ClientTypes (Base64): Liste aller definierten Clienttypen

**Beispiel:****Aufbau von Clienttypes**

```
<ClientTypes>
<ClientType Id=" " Name=" " />
  <ClientType Id=" " Name=" " />
</Clienttypes>
```

**Hinweis:****Genauere Beschreibung von ClientTypes**

- § Id (STRING): GUID des ClientTypes

§ Name (STRING): Name des ClientTyps

**Siehe auch:**

[wfm.InsertSysClienttypes](#)

## wfm.InsertSysClienttypes

**Beschreibung:**

Job noch ist noch nicht implementiert. Dieser Job definiert neue Clienttypen.

**Parameter:**

Clienttypes (Base64): Liste aller definierten Clienttypen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Beispiel:**

Aufbau von Clienttypes

```
<Clienttypes>
<Clienttype Id=" " Name=" " />
  <Clienttype Id=" " Name=" " />
</Clienttypes>
```

**Hinweis:**

Genauere Beschreibung von Clienttypes

§ Id (STRING): GUID des Clienttyps

§ Name (STRING): Name des Clienttyps

**Siehe auch:**

[wfm.GetSysClienttypes](#)

## wfm.DeleteSysClienttypes

**Beschreibung:**

Job noch ist noch nicht implementiert. Dieser Job löscht die angegebenen Clienttypen.

**Parameter:**

ClientTypeIds (STRING): kommaseparierte GUID's der zu löschenden Clienttypen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[wfm.GetSysClienttypes](#)

## wfm.GetProjectList

**Beschreibung:**

Dieser Job liefert alle Workflowprojekte aus einer Organisation.



**Parameter:**

OrganisationId (STRING): GUID der Organisation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Rückgabewerte:**

Projects (Base64): Liste aller definierten Clienttypen

**Beispiel:**

Aufbau von Clienttypes

```
<Projects>
<Project Id="" Name="" Type="" CreatorId="" CreationTime="" Description="" />
  <Project Id="" Name="" Type="" CreatorId="" CreationTime="" Description="" />
</Projects>
```

**Hinweis:**

Genauere Beschreibung von Clienttypes

§ Id (STRING): ID des Projekts

§ Name (STRING): Name des Projekts

§ Type (INT): 1 = Wurzelprojekt, 2 = Workflowprojekt, 3 = Workflowfamilie

§ CreatorID (STRING): GUID des Erstellers

§ CreationTime (INT): Erstelltzeitpunkt

§ Description (STRING): Beschreibung

## wfm.AdhocConfigTemplate

**Beschreibung:**

Mit diesem Job lassen sich Adhoc-Vorlagen im Workflow konfigurieren.

**Parameter:**

UserId (STRING): ID des Benutzers

OrgId (STRING): Instanz der Aktivität

Action (Int): 1: Speichern einer Vorlage, 2: Löschen einer Vorlage, 3: Veröffentlichen einer Vorlage, 4: Privatisieren einer Vorlage

In Abhängigkeit von der speziellen Aktion können weitere Parameter erforderlich sein:

1: Speichern einer Vorlage

TemplateId (String): ID der Vorlage. Ist leer, wenn die Vorlage erstmalig gespeichert wird.

TemplateName(String): Name der Vorlage

Public (Int): 0: es handelt sich um eine private Vorlage, 1: die Vorlage ist öffentlich

Template(BASE64):

2: Löschen einer Vorlage

TemplateId (String): ID der Vorlage.

3: Veröffentlichen einer Vorlage

TemplateId (String): ID der Vorlage.

#### 4: Privatisieren einer Vorlage:

TemplateId (String): ID der Vorlage.

#### Rückgabe:

Die Rückgabe ist abhängig von der gewählten Aktion (Action (Int)):

##### 1: Speichern einer Vorlage

TemplateId (String): ID der Vorlage.

##### 2: Löschen einer Vorlage

kein Rückgabeparameter

##### 3: Veröffentlichen einer Vorlage

kein Rückgabeparameter

##### 4: Privatisieren einer Vorlage:

kein Rückgabeparameter

#### Beispiel:

##### Aufbau von Template

```
<RoutingList Id="3294B433BFF6454D9C861B86B5A8AD5D"
ActivityId="3294B433BFF6454D9C861B86B5A8AD5D" Expandable="1">
<Entries>
<Entry Nr="203" Expandable="1">
<Item Id="99825B18A8334987935684FDA3D6A40D"
ActivityId="6EE4490A48164A0FA6DC34A80099AF66" ActivityName="Rechnung erstellen"
ModelActivityName="Rechnung erstellen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
</Entry>
<Entry Nr="253" Expandable="1">
<Item Id="E15594D692C14FDA9AFDE8FA0A43F6E4"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen BL"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
<Item Id="C6DA9503CD874D69A9B703D0E06A52E8"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen GF"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
</Entry>
</Entries>
</RoutingList>
```

#### Hinweis:

Genauere Beschreibung von RoutingList

§ RoutingList: Laufliste mit folgender Struktur (oder Untermengen davon)

§ Id (String): ID der Laufliste. Dieser Wert wird vom Server gesetzt und darf nicht verändert werden.

§ ActivityId (String): Aktivitätsid

- § Expandable (Int): 0: Laufliste kann nicht erweitert werden, 1: Laufliste kann erweitert werden.
- § Entries: Die Struktur fasst Einträge der Laufliste zusammen. Ein Eintrag besteht aus mehreren Elementen, die parallel ausgeführt werden können.
- § Entry: Beschreibt einen Eintrag in der Laufliste.
- § Nr (Int): Dient der relativen Sortierung der Einträge innerhalb der Laufliste. Die absoluten Werte haben für den Client keine Bedeutung.
- § Expandable (Int): 0: Eintrag kann nicht erweitert werden, 1: Eintrag kann erweitert werden.
- § Item: Beschreibt ein Element der Laufliste. Hierbei handelt es sich um eine Aktivität, eine ausführende Person und ggfs. einen Termin.
- § Id (STRING): Dient der Identifizierung. Diese ID darf nicht verändert werden und muss bei allen Jobs identisch mitgeschickt werden. Wurde ein Item durch den Client erstellt, muss dieser hier eine ID angeben.
- § ActivityId (String): ID der Aktivität im Workflowmodell
- § ActivityName (String): Name der Aktivität (muss nicht unbedingt mit dem Namen im Workflowmodell übereinstimmen).
- § ActivityModelName (String): Name der Aktivität im Workflowmodell
- § TimerId(String): ID einer Mahnfrist
- § TimerName(String): Name der Mahnfrist
- § TimerDuration(Int): Dauer der Frist
- § TimerDurationType(Int): 0: keine Frist, 1: relativ, 2: absolut
- § Changeable(Int): 0: keine Änderung möglich, 1: Das Element darf vom Client verändert werden.
- § Deleteable(Int): 0: Löschen nicht erlaubt, 1: Element darf gelöscht werden
- § Remark (String): Hinweis zur Bearbeitung (Text)
- § ObjectsIds (String): Liste von GUIDS der Bearbeiter (Rollen oder Personen), durch Komma getrennt

## wfm.AdhocGetTemplateList

### Beschreibung:

Liefert eine mehrere Adhoc-Vorlagen für den angegebenen Benutzer.

### Parameter:

UserId (STRING): ID des Benutzers

OrgId (STRING): Instanz der Aktivität

TemplateId (String): ID der Vorlage. Ist dieser Parameter leer, so werden alle Adhoc-Vorlagen ermittelt, die für den Benutzer sichtbar sind (= alle öffentlichen Vorlagen und die vom Benutzer privatisierten Vorlagen)

### Rückgabe:

Templates (BASE64): Liste der ermittelten Adhoc-Vorlagen

Template: Beschreibt eine Adhoc-Vorlage

TemplateId(String): ID der Adhoc-Vorlage

TemplateName(String): Name der Adhoc-Vorlage

Public(int): 0: die Vorlage ist nicht öffentlich, 1: die Vorlage ist öffentlich

§ RoutingList: Laufliste mit folgender Struktur (oder Untermengen davon)

§ Id (String): ID der Laufliste. Dieser Wert wird vom Server gesetzt und darf nicht verändert werden.

§ ActivityId (String): Aktivitätsid

§ Expandable (Int): 0: Laufliste kann nicht erweitert werden, 1: Laufliste kann erweitert werden.

§ Entries: Die Struktur fasst Einträge der Laufliste zusammen. Ein Eintrag besteht aus mehreren Elementen, die parallel ausgeführt werden können.

§ Entry: Beschreibt einen Eintrag in der Laufliste.

§ Nr (Int): Dient der relativen Sortierung der Einträge innerhalb der Laufliste. Die absoluten Werte haben für den Client keine Bedeutung.

§ Expandable (Int): 0: Eintrag kann nicht erweitert werden, 1: Eintrag kann erweitert werden.

§ Item: Beschreibt ein Element der Laufliste. Hierbei handelt es sich um eine Aktivität, eine ausführende Person und ggfs. einen Termin.

§ Id (STRING): Dient der Identifizierung. Diese ID darf nicht verändert werden und muss bei allen Jobs identisch mitgeschickt werden. Wurde ein Item durch den Client erstellt, muss dieser hier eine ID angeben.

§ ActivityId (String): ID der Aktivität im Workflowmodell

§ ActivityName (String): Name der Aktivität (muss nicht unbedingt mit dem Namen im Workflowmodell übereinstimmen).

§ ActivityModelName (String): Name der Aktivität im Workflowmodell

§ TimerId(String): ID einer Mahnfrist

§ TimerName(String): Name der Mahnfrist

§ TimerDuration(Int): Dauer der Frist

§ TimerDurationType(Int): 0: keine Frist, 1: relativ, 2: absolut

§ Changeable(Int): 0: keine Änderung möglich, 1: Das Element darf vom Client verändert werden.

§ Deleteable(Int): 0: Löschen nicht erlaubt, 1: Element darf gelöscht werden

§ Remark (String): Hinweis zur Bearbeitung (Text)

§ ObjectsIds (String): Liste von GUIDS der Bearbeiter (Rollen oder Personen), durch Komma getrennt

### Beispiel:

#### Aufbau von Templates

```
<Templates>
<Template TemplateId="" TemplateName="" Public="">
<RoutingList Id="3294B433BFF6454D9C861B86B5A8AD5D"
ActivityId="3294B433BFF6454D9C861B86B5A8AD5D" Expandable="1">
<Entries>
<Entry Nr="203" Expandable="1">
<Item Id="99825B18A8334987935684FDA3D6A40D"
ActivityId="6EE4490A48164A0FA6DC34A80099AF66" ActivityName="Rechnung erstellen"
ModelActivityName="Rechnung erstellen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
```

```

<ObjectIds></ObjectIds>
</Item>
</Entry>
<Entry Nr="253" Expandable="1">
<Item Id="E15594D692C14FDA9AFDE8FA0A43F6E4"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen BL"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
<Item Id="C6DA9503CD874D69A9B703D0E06A52E8"
ActivityId="6EE4490A48164A0FA6DC34A80099AF67" ActivityName="Rechnung genehmigen GF"
ModelActivityName="Rechnung genehmigen" Remark="" TimerId="" TimerDuration=""
TimerDurationType="" Changeable="1" Deleteable="0">
<ObjectIds></ObjectIds>
</Item>
</Entry>
</Entries>
</RoutingList>
</Template>
</Templates>

```

## Serverinterne Jobs

Die hier aufgeführten Jobs sind Server-intern und werden somit im Allgemeinen nicht von aussen aufgerufen.

## wfm.DBCommands

### Beschreibung:

Dieser Job führt gecachte Datenbankkommandos aus. Der Job wird im Allgemeinen nicht von aussen aufgerufen.

### Parameter:

DBCommandsId (STRING): GUID des auszuführenden Datenbankkommandos im DB-Kommandos-Cache

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## BatchJobs

Die hier aufgeführten Jobs werden regelmäßig von der Workflow-Engine selbst ausgeführt.

§ [wfm.CheckJob](#)

§ [wfm.WorkerJob](#)

§ [wfm.WorkItemNoti](#)

## wfm.CheckJob

### Beschreibung:

Dieser Job stellt fest, ob Mahn- und Sperrfristen abgelaufen sind und reagiert entsprechend darauf. Durch Sperrfristen betroffene Aktivitäten werden aktiviert und die Sperrfristen werden aus der Datenbank entfernt. Durch Mahnfristen betroffene Aktivitäten werden als verspätet gekennzeichnet

und die entsprechend definierte Aktion (E-Mailnachricht, Weiterleiten an Stellvertreter ...) wird durchgeführt.

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.WorkerJob

### Beschreibung:

Dieser Job arbeitet die Aktivitäten-Queue ab. Es werden Aktivitäten gestartet bzw. beendet.

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.WorkItemNoti

### Beschreibung:

Dieser Job verschickt den ServerJob 'ServerNotifyClients'. Der Job ServerNotifyClients benachrichtigt z. B. alle angeschlossenen Clients über Veränderungen im Eingangskorb.

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## ServerCommunicationJobs

Die hier aufgeführten Jobs dienen der Kommunikation zwischen verschiedenen Servern. Sie werden also im Gegensatz zu anderen Jobs grundsätzlich nicht von Clients, sondern von einem (anderen) Server aufgerufen.

§ [wfm.ServerNotifyClients](#)

§ [wfm.ServerUpdateWorkflowModels](#)

§ [wfm.ServerUserAbsent](#)

## wfm.ServerNotifyClients

### Beschreibung:

Dieser Job schickt eine Nachricht zur Aktualisierung des Eingangskorbes an die Clients, deren Benutzer in der Liste von Benutzer-IDs aufgeführt ist. Ist die Liste leer, wird eine Nachricht an alle angeschlossenen Clients versandt.

### Parameter:

UserGUIDs (STRING): kommaseparierte Liste von Benutzer-GUID's

Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## wfm.ServerUpdateWorkflowModels

### Beschreibung:

Dieser Job löscht alle in der Liste aufgeführten Workflow-Modelle aus dem WorkflowEngine-Cache und schickt eine Benachrichtigung an alle angeschlossenen OS:4.DRT-Workflow\_Editoren.

**Parameter:**

OrganisationId (STRING): GUID der Organisation, aus der die Modelle kommen

WorkflowIds (STRING): kommasseparierte Liste von WorklowModell-GUID's

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**wfm.ServerUserAbsent****Beschreibung:**

Dieser Job benachrichtigt alle angeschlossenen OS:4.DRT-Workflow\_Editoren darüber, für welche Benutzer sich der Anwesenheits-Status geändert hat.

**Parameter:**

OrganisationId (STRING): GUID der Organisation, aus der die Modelle kommen

AbsentIds (STRING): kommasseparierte Liste von Benutzer-GUID's, die anwesend sind

PresentIds (STRING): kommasseparierte Liste von Benutzer-GUID's, die abwesend sind

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Core Services

Core Services beinhalten Funktionen in den Bereichen Administration, Lizenzierung, Sessionmanagement, Engine-Verwaltung und interne Steuerungen. Diese werden durch den Applikationsserver-Kern (axsvckrn.exe) gekapselt. Damit ist es möglich, auch ohne das Hinzuladen von diversen Engines bereits Grundfunktionen zur Administration aufzurufen.

### Namespaces

- § [Administrations-Core-Services](#) (Namespace adm)
- § [Kernel-Core-Services](#) (Namespace krn)
- § [Lizenz-Core-Services](#) (Namespace lic)

### Administrations-Core-Services (Namespace adm)

Im Namespace 'adm' sind Funktionen zur Verwaltung von Systemdateien und einige Konfigurationsaufgaben am Server zu erledigen. Dieser Namespace wird direkt vom Kernel implementiert.

- § [adm.CleanUpConfig](#)
- § [adm.CleanUpLog](#)
- § [adm.EnumServerGroups](#)
- § [adm.EnumServers](#)
- § [adm.GetServerFamilyInfo](#)
- § [adm.GetServersActivity](#)
- § [adm.GetSystemFile](#)
- § [adm.LogdirDeleteFiles](#)
- § [adm.LogdirDownloadFiles](#)
- § [adm.LogdirGetInfo](#)
- § [adm.StoreSystemFile](#)

### adm.CleanUpConfig

#### **Beschreibung:**

Der Job sucht nach der spezifizierten Konfigurationsdatei und leert sie.

#### **Parameter:**

Flags (INT): z. Z. nicht unterstützt

Types (INT): Typ der Konfigurationsdatei

- § 1 = Objektdefinitionsdatei
- § 2 = AS.cfg
- § 4 = AsListen.dat
- § 8 = AsImpExp.Cfg
- § 16 = AsForm.Cfg
- § 32 = AsCold.Cfg



§ 4294967295 = alle, die sich finden

Versions (INT): Version der Datei

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## adm.CleanUpLog

**Beschreibung:**

Der Job sucht nach den spezifizierten Log-Dateien und entfernt sie.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Type (INT): Typ der Log-Dateien, die gelöscht werden sollen

§ 1 = FLW-Dateien

§ 2 = ERR-Dateien

§ 4 = LOG-Dateien

§ 8 = REP-Dateien

§ 4294967295 = alle, die sich finden

Days (INT): Tage

Path (STRING): Pfad des Logverzeichnis

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## adm.EnumServerGroups

**Beschreibung:**

Dieser Job liefert eine Liste aller Servergruppen.

**Parameter:**

Flags (INT): 0 = Rückgabewert ist Group[00..nn]; 1 = Rückgabewerte sind Info und InfoType

**Rückgabewerte:**

[Info] (STRING): MIME-codierter Puffer mit Informationen zur Servergruppe

[InfoType] (STRING): Semikolon-separierte Namen der Werte, die in Info zurückgeliefert werden

[Group[00..nn]] (STRING): Semikolon-separierte Informationen zur Servergruppe

§ ID der Servergruppe

§ Name der Servergruppe

§ Beschreibung zur Servergruppe

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## adm.EnumServers

### Beschreibung:

Dieser Job liefert alle Server einer Servergruppe.

### Parameter:

Flags (INT): 0 = Rückgabewert ist Server[00..nn]; 1 = Rückgabewerte sind Info und InfoType

[GroupID] (INT): ID der Servergruppe

### Rückgabewerte:

[Info] (STRING): MIME-codierter Puffer mit Informationen zur Servergruppe

[InfoType] (STRING): Semikolon-separierte Namen der Werte, die in Info zurückgeliefert werden

[Server [00..nn]]: Semikolon-separierte Informationen zum Server

§ ID der Servers

§ Name der Servers

§ IP-Adresse

§ Port

§ Dienstname

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## adm.GetServerFamilyInfo

### Beschreibung:

Dieser Job liefert die ID und den Namen der Serverfamilie.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

### Rückgabewerte:

GUID (STRING): ID der Serverfamilie

Name (STRING): Name der Serverfamilie

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## adm.GetServersActivity

### Beschreibung:

Dieser Job liefert die Ids alle Server und den dazugehörigen Status.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

### Rückgabewerte:

Server[00..nn] (STRING): Semikolon-separierte Informationen zum Server

§ ID des Servers

§ Status des Servers

§ 1 = Server läuft

§ 2 = Server abgestürzt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## adm.GetSystemFile

**Beschreibung:**

Der Job übergibt dem Client die Systemdatei (z. B. as.cfg) und sperrt sie. Systemdateien stehen in der DB-Tabelle 'osresources' und haben den resourcetype 1.

**Parameter:**

Flags (INT):

§ LOWORD(Flags) = 0: die Systemdatei wird zum Lesen geöffnet

§ LOWORD(Flags) = 2: die Systemdatei wird zum Schreiben geöffnet

§ Version != 0 oder HIWORD(Flags) != 0 oder LOWORD(Flags) != 2

§ HIWORD(Flags) = 1: LowDateTime und HighDateTime werden zurückgegeben

§ HIWORD(Flags) = 2: LowDateTime wird zurückgegeben

FileName (STRING): Name der Systemdatei

Version (INT): Version der Systemdatei die angefordert werden soll

**Rückgabewerte:**

[FileCount] (INT): FileCount gleich 1

[LowDateTime] (INT): LowDateTime

[HighDateTime] (INT): HighDateTime

[Dateiliste]: Name und Pfad der Systemdatei

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[adm.StoreSystemFile](#)

## adm.LogdirDeleteFiles

**Beschreibung:**

Dieser Job löscht aus dem Log-Verzeichnis des Servers die angegebenen Dateien.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Files (STRING): durch '?' getrennte Dateinamen

**Rückgabewerte:**

Deleted (INT): Anzahl der gelöschten Datei

Failed (INT): Anzahl der Dateien, die nicht gelöscht werden konnten

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[adm.LogdirGetInfo](#)

## adm.LogdirDownloadFiles

**Beschreibung:**

Dieser Job liefert die angegebenen Dateien aus dem Log-Verzeichnis des Servers. Die Dateien werden gepackt zurückgegeben.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Files (STRING): durch '?' getrennte Dateinamen

**Rückgabewerte:**

Files (STRING): Namen der zurückgelieferten Dateien

Dateiliste: Name und Pfad der gepackten Datei (enthält alle angeforderten Dateien)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[adm.LogdirGetInfo](#)

## adm.LogdirGetInfo

**Beschreibung:**

Dieser Job liefert eine Liste aller Dateien aus dem Log-Verzeichnis des Servers.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

FileInformation[00000000..nnnnnnnnnn] (STRING): durch '?' getrennte Datei-Informationen

§ Dateiname

§ Dateigröße in Byte

§ Zeitstempel der letzten Änderung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[adm.LogdirDownloadFiles](#)[adm.StoreSystemFile](#)**Beschreibung:**

Der Job speichert die vom Client erhaltene Systemdatei.

**Parameter:**

Flags (INT):

- § LOWORD(Flags) = 0: Systemdatei speichern und Historiendatei schreiben
- § LOWORD(Flags) = 1: Locken der Datei rückgängig machen
- § HIWORD(Flags) = 2: Datumsstempel im LowDateTime-Format speichern; ansonsten LowDateTime-Format und HighDateTime-Format verwenden

FileName (STRING): Name der Systemdatei

LowDateTime (INT): Datumsstempel im LowDateTime-Format

HighDateTime (INT): Datumsstempel im HighDateTime -Format

Dateiliste: Name und Pfad der Systemdatei

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**[adm.GetSystemFile](#)[Kernel-Core-Services \(Namespace krn\)](#)

Die Jobs des Kernel-Executors 'krn' dienen der internen Verwaltung der Applikationsserverprozesse. Dies sind insbesondere Funktionen zur Batch-Verwaltung, Serverüberwachung, Registry-Verwaltung und zum Administrieren geladener Engines zur Laufzeit.

- § [Registry-Verwaltung](#)
- § [Batch-Verwaltung](#)
- § [Server-Verwaltung](#)
- § [Session-Verwaltung](#)
- § [Engine-Verwaltung](#)
- § [Sonstige Jobs](#)

[Registry-Verwaltung](#)

Diese Jobs dienen der Registry-Verwaltung. Es können Registry-Einträge abgefragt oder geändert werden.

- § [krn.REBackup](#)
- § [krn.REGetCurrentSchema](#)
- § [krn.REGetRegValue](#)
- § [krn.RELoad](#)
- § [krn.RESave](#)

## § [krn.RESetRegValue](#)

### krn.REBackup

**Beschreibung:**

Der Job erzeugt ein Backup im xml-Format vom aktuellen Registry-Schema.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

FileName (STRING): Name der Zielfile inklusive Pfadangabe, die Dateiendung sollte .xml sein

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### krn.REGetCurrentSchema

**Beschreibung:**

Dieser Job liefert das aktuelle Registry-Schema in xml-Notation.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Schema (STRING): enthält das aktuelle Schema in xml-Notation

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### krn.REGetRegValue

**Beschreibung:**

Dieser Job liefert den Wert eines angegebenen Registry-Eintrags.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Name (STRING): Name des Registry-Eintrags

**Rückgabewerte:**

Value (STRING): Wert des Registry-Eintrags

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.RESetRegValue](#)

### krn.RELoad

**Beschreibung:**

Dieser Job lädt das aktuelle Registry-Schema in den Speicher.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.RESave

**Beschreibung:**

Der Job speichert das aktuelle Registry-Schema. Der Schlüssel ist HKLM\SOFTWARE\OPTIMAL SYSTEMS\[Dienstname des Applikationsservers]\Schemata \[Versionsnummer(z. B. 4.0)].

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.RESetRegValue

**Beschreibung:**

Dieser Job ändert den Wert des angegebenen Registry-Eintrags.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Name (STRING): Name des Registry-Eintrags

Value (STRING): Wert des Registry-Eintrags

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.REGetRegValue](#) , [krn.RESave](#)

## Batch-Verwaltung

§ [krn.BatchAdd](#)

§ [krn.BatchChange](#)

§ [krn.BatchEnum](#)

§ [krn.BatchGetStatistic](#)

§ [krn.BatchRemove](#)

## krn.BatchAdd

**Beschreibung:**

Der Job fügt einen Batch der Registry hinzu.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Registry (STRING): Name in der Registry

Name (STRING): Name des hinzuzufügenden Batches

Namespace (STRING): Namespaces des auszuführenden Jobs

JobName (STRING): Name des auszuführenden Job (ohne Namespace)

Scheduling (STRING): gibt den Zeitpunkt der Ausführung an

Period (INT): Dauer in Millisekunden, zwischen den Aufrufen des Jobs

Enabled (BOOL): 1 = Batch ist eingeschaltet, ansonsten 0

DoLog (BOOL): 1 = Job wird protokolliert, ansonsten 0

Parameters: Name, Datentyp, Wert des Jobs (MIME-codierter Puffer)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.BatchChange](#) , [krn.BatchRemove](#)

## krn.BatchChange

**Beschreibung:**

Dieser Job ändert einen bestehenden Batch.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Registry (STRING): Name in der Registry

Name (STRING): Name des hinzuzufügenden Batches

Namespace (STRING): Namespaces des auszuführenden Jobs

JobName (STRING): Name des auszuführenden Job (ohne Namespace)

Scheduling (STRING): gibt den Zeitpunkt der Ausführung an

Period (INT): Dauer in Millisekunden, zwischen den Aufrufen des Jobs

Enabled (BOOL): 1 = Batch ist eingeschaltet, ansonsten 0

DoLog (BOOL): 1 = Job wird protokolliert, ansonsten 0

Parameters: Name, Datentyp, Wert des Jobs (MIME-codierter Puffer)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.BatchEnum](#)

## krn.BatchEnum

**Beschreibung:**

Der Job liefert eine Liste der vorhandenen Batches.



**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Batch[1..n] (STRING): Informationen zum Batch

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Hinweis:**

zurückgelieferte Batchinformationen

- § Registry: Name des Registry-Eintrags
- § Name: Name des Batch
- § Period: Dauer in Millisekunden, zwischen den Aufrufen des Jobs
- § Scheduling:
- § JobName: Name des auszuführenden Job (ohne Namespace)
- § Namespace: Namespaces des auszuführenden Jobs
- § Enabled: 1 = Batch ist eingeschaltet, ansonsten 0
- § DoNotLog: 1 = Job wird protokolliert, ansonsten 0
- § TimeCreated: Erstellzeitpunkt des Batch
- § TimeEnabled: Zeitpunkt der Aktivierung
- § TimeFired: Zeitpunkt der letzten Ausführung
- § Parameters: MIME-codierte Parameter des Jobs(Name, Datentyp, Wert)

**Siehe auch:**

[krn.BatchAdd](#) , [krn.BatchChange](#) , [krn.BatchRemove](#) , [krn.BatchGetStatistic](#)

## krn.BatchGetStatistic

**Beschreibung:**

Der Job gibt statistische Parameter zum ausgewählten Batch aus.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Name (STRING): Name des Batches, dessen Job-Parameter ausgegeben werden sollen

**Rückgabewerte:**

Ticks (STRING): Anzahl der Jobausführungen

PeriodAve (STRING): Durchschnittlicher Zeitraum zwischen den Ticks (in Sekunden)

PeriodLast (STRING): Zeitraum zwischen dem letzten und vorletzten Tick (in Sekunden)

DeviationAveP (STRING): Durchschnittliche Abweichung der Intervalle zwischen den Ticks und der Vorgabe (in Prozent)

DeviationAveS (STRING): Durchschnittliche Abweichung der Intervalle zwischen den Ticks und der Vorgabe (in Sekunden)

DeviationLastP (STRING): Abweichung des Intervalls zwischen den letzten beiden Ticks und der Vorgabe (in Prozent)

DeviationLastS (STRING): Abweichung des Intervalls zwischen den letzten beiden Ticks und der Vorgabe (in Sekunden)

DeviationMaxP (STRING): Maximale Abweichung der Intervalle zwischen den Ticks und der Vorgabe (in Prozent)

DeviationMaxS (STRING): Maximale Abweichung der Intervalle zwischen den Ticks und der Vorgabe (in Sekunden)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.BatchEnum](#)

## krn.BatchRemove

**Beschreibung:**

Der Job entfernt einen existierenden Batch.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Registry (STRING): Name des Registry-Eintrags

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.BatchEnum](#)

## Server-Verwaltung

Diese Jobs dienen der Server-Verwaltung.

§ [krn.AppsEventsEnum](#)

§ [krn.AppsEventsSubscribe](#)

§ [krn.CheckCrashedServers](#)

§ [krn.CheckServerConnection](#)

§ [krn.GetServerInfo](#)

§ [krn.GetServerInfoEx](#)

§ [krn.MakeBeatPing](#)

§ [krn.RefillServerList](#)

§ [krn.ShutDown](#)

## krn.AppsEventsEnum

**Beschreibung:**

Dieser Job liefert alle definierten Eventtypen, die im Server implementiert sind.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Info (STRING): Semikolon-separierter MIME-codierter Puffer, der Eventtypen erhält

Infotyp (STRING): enthält Beschreibung der Daten, die im Parameter Info zurückgeliefert werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## [krn.AppsEventsSubscribe](#)

**Beschreibung:**

Dieser Job veranlasst den Server Benachrichtigungen (Notifikationen) für die angegebenen Events an den Jobaufrufer zu senden.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Events (STRING): Semikolon-separierte Events

§ krn.SessionLogin

§ krn.SessionLogout

§ krn.Connected

§ krn.Disconnected

§ krn.JobCall

§ krn.Executor

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## [krn.CheckCrashedServers](#)

**Beschreibung:**

Der Job sucht nach Servern, die abgestürzt sind und gibt deren Ressourcen frei. In der DB-Tabelle 'ospingtable' wird für diesen Server der Serverstatus auf 2 = 'hung server' gesetzt.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## [krn.MakeBeatPing](#)

## [krn.CheckServerConnection](#)

**Beschreibung:**

Dieser Job überprüft die Serververbindung, indem eine globale Zählervariable bei jedem Aufruf dieses Jobs um 1 inkrementiert wird.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Callnumber (LONG): Anzahl der Jobaufrufe

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.GetServerInfo

**Beschreibung:**

Dieser Job liefert eine bestimmte Server-Information.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Info (INT): Informationstyp (siehe [Tab](#)), der angefordert werden soll

**Rückgabewerte:**

Info (INT): ist gleich dem Eingabeparameter

Name (STRING): der zur Info gehörende String

Value (STRING): gibt die abgefragte Server-Information aus

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.GetServerInfoEx](#)

Infotyp	Infostring	Beschreibung
1	ServerID	ID des Servers
2	ComputerName	Name des Servers
3	InstanceName	Name der Programminstanz
4	ComString	Server-Comstring (z. B. 127.0.0.1)
5	DataBaseSourceC	Datenbankname (z. B. os400)
6	DataBaseParser	Datenbankparser (z. B. oxorantl.dll)
7	ClientETC	Pfad zu den Client-Konfigurationsdateien
8	StatusLine	String des Statusbalken
9	ServerType	0 = Hauptserver; 1 = Nebenserver
10	ValidDomains	

11	DataBaseModule	
12	DataBaseParsType	
13	DataBaseSchema	
14	DataBaseProvider	
15	DataBaseConString	
16	ETC	vollständiger Pfad des Serververzeichnisses 'etc'
17	DataBaseNewMethod	
18	DataBaseModuleS	
19	DataBaseModuleClient	
1000	FileVersionListUpdate	liefert alle Dateinamen+Erstellzeit der Serververzeichnisses server\etc\update
1001	FileVersionListClient	liefert alle Dateinamen+Erstellzeit der Serververzeichnisses server\etc\update\client
1002	FileVersionListAdmin	liefert alle Dateinamen+Erstellzeit der Serververzeichnisses server\etc\update\admin
1003	FileVersionListIndex	liefert alle Dateinamen+Erstellzeit der Serververzeichnisses server\etc\update\index
1004	FileVersionListTemplate	liefert alle Dateinamen+Erstellzeit der Serververzeichnisses server\etc\templates

## krn.GetServerInfoEx

### Beschreibung:

Dieser Job liefert die angegebenen Server-Informationen.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Info (STRING): Semikolon-separierte Parameter, die ausgegeben werden sollen  
(;Parameter;Parameter;...)

§ ? -> die Namen aller Parameter werden als Liste zurückgegeben

§ leerer String -> alle Parameter und ihre Werte werden als Liste zurückgegeben

### Rückgabewerte:

[Param[000...NNN]] (STRING): Ausgabe nur bei Info = ?; Namen aller Parameter, welche über Info übergeben werden können

[Parametername] (STRING): Serverinformation

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Siehe auch:

[krn.GetServerInfo](#)

## krn.MakeBeatPing

**Beschreibung:**

Der Job führt einen Ping zum Applikationsserver aus, die Tabelle 'ospingtable' wird mit Serverstatus 1 = 'Server aktiv' aktualisiert.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**[krn.CheckCrashedServers](#)

## krn.RefillServerList

**Beschreibung:**

Dieser Job lädt die Serverinformation aus den DB-Tabellen 'ospingtable' und 'sever' in den Speicher.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.ShutDown

**Beschreibung:**

Dieser Job fährt den Server runter.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Session-Verwaltung

Diese Jobs dienen der Session-Verwaltung.

§ [krn.SessionAttach](#)

§ [krn.SessionDeleteLost](#)

§ [krn.SessionDrop](#)

§ [krn.SessionDropDB](#)

§ [krn.SessionEnum](#)

§ [krn.SessionEnumDB](#)

§ [krn.SessionEnumResourcesDB](#)

- § [krn.SessionGetInfo](#)
- § [krn.SessionLogin](#)
- § [krn.SessionLogout](#)
- § [krn.SessionPropertiesEnum](#)
- § [krn.SessionPropertiesGet](#)
- § [krn.SessionPropertiesSet](#)
- § [krn.UserSessionCreate](#)
- § [krn.UserSessionDelete](#)

## krn.SessionAttach

### **Beschreibung:**

Dieser Job erzeugt eine neue Arbeitssitzung mit dem Applikationsserver oder nimmt eine noch bestehende Arbeitssitzung wieder auf.

### **Parameter:**

Flags (INT): z. Z. nicht unterstützt

SessionGUID (STRING): GUID der Session, die fortgesetzt werden soll; leer = neue Session wird erzeugt

### **Rückgabewerte:**

SessionGUID (STRING): GUID der neuen Session oder der bestehenden Session

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.SessionDeleteLost

### **Beschreibung:**

Dieser Job löscht alle Session aus der Datenbank, die nicht mehr online sind.

### **Parameter:**

Flags (INT): 1 = Parameter AgeHours wird beachtet

AgeHours (INT): Anzahl der Stunden

### **Rückgabewerte:**

DeletedSessions (STRING): ID der gelöschten Sessions

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.SessionDrop

### **Beschreibung:**

Dieser Job löscht die angegebene Arbeitssitzung.

### **Parameter:**

Flags (INT): z. Z. nicht unterstützt

SessionGUID (STRING): ID der Arbeitssitzung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krm.SessionDropDB

**Beschreibung:**

Dieser Job löscht alle Einträge zur Arbeitssitzung in der Datenbank (oslockedres, ossession).

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

SessionGUID (STRING): ID der Arbeitssitzung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krm.SessionEnum

**Beschreibung:**

Dieser Job liefert eine Liste aller bestehenden Arbeitssitzungen zum Server.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Sessions (STRING): Semikolon-separierte GUID-Liste aller bestehenden Arbeitssitzungen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krm.SessionEnumDB

**Beschreibung:**

Dieser Job liefert eine Liste aller Datenbankeinträge zu Arbeitssitzungen.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

SessionInfoType (STRING): enthält Beschreibung der Daten, die im Parameter Sessions zurückgeliefert werden

Sessions (STRING): MIME-codierter Puffer mit Informationen zu Arbeitssitzungen

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krm.SessionEnumResourcesDB

**Beschreibung:**



Dieser Job liefert zur angegebenen Arbeitssitzung alle verwendeten Ressourcen, die in der Datenbank gespeichert sind.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

SessionGUID (STRING): ID der Arbeitssitzung

**Rückgabewerte:**

Resources (STRING): Semikolon-separierte Ressourcen (Format: GUID1=Locktime1,Name1;GUID2=Locktime2,Name2;...)

§ ID der Resource

§ Zeitpunkt, zu dem die Resource gelockt wurde

§ Kurzbezeichnung der Resource

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.SessionEnum](#)

## krn.SessionGetInfo

**Beschreibung:**

Dieser Job liefert Informationen über die in der GUID übergebenen Arbeitssitzungen, die an dem Server existieren.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Sessions (STRING): Semikolon-separierte ID's der Arbeitssitzungen

**Rückgabewerte:**

SessionInfoType (STRING): Bezeichnungen der Informationen, die im Parameter SessionInfo zurückgeliefert werden

ConnectionInfoType (STRING): Bezeichnungen der Informationen, die im Parameter SessionInfo zurückgeliefert werden

SessionInfo (STRING): MIME-codierter Puffer mit Informationen zur Session

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.SessionLogin

**Beschreibung:**

Dieser Job meldet einen User für die aktive Session an. Der User wird über seinen Namen und das verschlüsselte Passwort identifiziert.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

UserName (STRING): Name des Benutzers

UserPwd (STRING): verschlüsseltes Passwort, für die Verschlüsselung des Passworts kontaktieren Sie bitte OPTIMAL SYSTEMS GmbH

[EntMgr] (LONG): Parameter ist nur bei Enterprise Manager Start zu setzen

**Rückgabewerte:**

Description (STRING): Beschreibung, falls ein Fehler auftritt

Action (STRING): Aktion, die ausgeführt wurde

[UserGUID] (STRING): GUID des Benutzers, der eingeloggt wurde (nur bei Enterprise Manager)

[UserID] (INT): ID des Benutzers, der eingeloggt wurde (nur bei Enterprise Manager)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.SessionAttach](#) , [krn.SessionLogout](#) , [krn.SessionPropertiesSet](#)

## krn.SessionLogout

**Beschreibung:**

Dieser Job beendet die Benutzung der aktiven Session.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.SessionPropertiesEnum

**Beschreibung:**

Dieser Job liefert die Namen aller Eigenschaften einer Arbeitssitzung mit dem Server.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Names (STRING): Semikolon-separierte Eigenschaften für eine Arbeitssitzung

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.SessionPropertiesGet

**Beschreibung:**

Dieser Job liefert die angegebenen Eigenschaften einer angegebenen Arbeitssitzung.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Properties (STRING): Semikolon-separierte Eigenschaften, die angezeigt werden sollen (für jede Eigenschaft, die hier angegeben ist, wird ein Ausgabeparameter erzeugt)

SessionGUID (STRING): GUID der Arbeitssitzung

**Rückgabewerte:**

Die Ausgabeparameter werden durch den Eingabeparameter Properties bestimmt.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Hinweis:**

Eigenschaften einer Session

- § address: Adresse des Applikationsservers
- § sessguid: GUID der Session
- § statname: Computername der Arbeitsstation
- § instname: Name der Instanz
- § statguid: GUID der Arbeitsstation
- § userguid: GUID des Benutzers
- § hasserveraccount: hat die Session einen ServerAccount
- § loggedin: ist ein User eingeloggt
- § haschannel: hat die Session einen Kanal?
- § autologin: Login über Dialog oder Autologin
- § supervisor: Name des Supervisors
- § langid: ID der Sprache

**Siehe auch:**

[krn.SessionPropertiesSet](#) , [krn.SessionPropertiesEnum](#)

## krn.SessionPropertiesSet

**Beschreibung:**

Dieser Job setzt die angegebenen Eigenschaften einer Session.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Properties (STRING): Semikolon-separierte Eigenschaften, die gesetzt werden sollen

[Eigenschaftsname] (STRING): Wert der Eigenschaft (für jede Eigenschaft, die im Parameter Properties angegeben ist, wird ein Eingabeparameter erzeugt)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Hinweis:**

Eigenschaften einer Session (Auszug)

- § address: Adresse des Clients

- § sessguid: GUID der Session
- § statname: Computername der Arbeitsstation
- § instname: Name der Instanz
- § statguid: GUID der Arbeitsstation
- § userguid: GUID des Benutzers
- § hasserveraccount: hat die Session einen ServerAccount
- § loggedin: ist ein User eingeloggt
- § haschannel: hat die Session einen Kanal?
- § autologin: Login über Dialog oder Autologin
- § supervisor: Name des Supervisors
- § langid: ID der Sprache

**Siehe auch:**

[krm.SessionPropertiesGet](#) , [krm.SessionPropertiesEnum](#)

## krm.UserSessionCreate

### **Beschreibung:**

Dieser Job erstellt eine Benutzer-Arbeitssitzung für eine B2B-Verbindung. Eine B2B-Verbindung besteht zwischen zwei Servern (z. B. DRT-Server und Java-Server).

### **Parameter:**

Flags (INT): z. Z. nicht unterstützt

### **Rückgabewerte:**

SessionGUID (STRING): ID der Arbeitssitzung

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krm.UserSessionDelete](#)

## krm.UserSessionDelete

### **Beschreibung:**

Dieser Job löscht eine Benutzer-Arbeitssitzung für eine B2B-Verbindung. Eine B2B-Verbindung besteht zwischen zwei Servern (z. B. DRT-Server und Java-Server).

### **Parameter:**

Flags (INT): z. Z. nicht unterstützt

SessionGUID (STRING): ID der Arbeitssitzung

### **Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.UserSessionCreate](#)

## Engine-Verwaltung

Diese Jobs dienen der Engine-Verwaltung.

§ [krn.EnumJobs](#)

§ [krn.EnumNameSpaces](#)

§ [krn.GetNameSpaceParams](#)

§ [krn.LoadExecutor](#)

§ [krn.NameSpaceEnum](#)

§ [krn.NameSpaceGetInfo](#)

§ [krn.NameSpaceGetJobsInfo](#)

§ [krn.ReloadExecutor](#)

§ [krn.UnloadExecutor](#)

### krn.EnumJobs

**Beschreibung:**

Der Job liefert eine Liste der implementierten Jobs für einen angegebenen Namespace.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

NameSpace (STRING): Kurzbezeichnung des Namespace, für den die Liste der Jobs erzeugt werden soll

**Rückgabewerte:**

[Jobname] (STRING): Name der implementierten Jobs

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.EnumNameSpaces](#)

### krn.EnumNameSpaces

**Beschreibung:**

Dieser Job liefert eine Liste der implementierten Namespaces.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

NameSpace[1..n] (STRING): Name des Namespaces, eine Sortierung erfolgt nach dem Alphabet

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.EnumJobs](#)

## krn.GetNamespaceParams

**Beschreibung:**

Der Job gibt die Namespace-Parameter zu einem angegebenen Namespace an.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Namespace (STRING): Kurzbezeichnung des Namespace

**Rückgabewerte:**

Child (INT): 1 = Executor wurde in einem eigenen Prozess gestartet, ansonsten 0

ExecutorPresent (BOOL): 1 = Executor ist vorhanden, ansonsten 0

Internal (BOOL): Interner Namespace

§ 1 = interner Namespace (im Kernel implementiert)

§ 0 = im Executor implementiert

Queue (STRING): Name der Queue für den Namespace

State (INT): Status des Namespaces als Zahl

§ 0 = CREATED

§ 1 = LOADING

§ 2 = LOADED

§ 3 = UNLOADING

§ 4 = UNLOADED

StateText (STRING): Status als Text

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**[krn.EnumNameSpaces](#)

## krn.LoadExecutor

**Beschreibung:**

Der Job lädt einen Executor.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Name (STRING): Kurzbezeichnung des Namespaces, der geladen werden soll

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**[krn.ReloadExecutor](#)

## karn.NamespaceEnum

**Beschreibung:**

Der Job liefert eine Liste der implementierten Namespaces.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Namespaces (STRING): Semikolon-separierte Kurzbezeichnungen aller Namespaces

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## karn.NamespaceGetInfo

**Beschreibung:**

Dieser Job liefert alle Informationen zum angegebenen Namespace.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

NameSpaces (STRING): Kurzbezeichnung des Namespace

**Rückgabewerte:**

NamespaceInfo (STRING): MIME-codierter Puffer, der Informationen enthält

NamespaceInfoType (STRING): enthält Beschreibung der Daten, die im Parameter NamespaceInfo zurückgeliefert werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## karn.NamespaceGetJobsInfo

**Beschreibung:**

Dieser Job liefert Informationen (Name, Anzahl der Jobaufrufe..) zu den Jobs des angegebenen Namespace.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

NameSpaces (STRING): Kurzbezeichnung des Namespace

**Rückgabewerte:**

NamespaceInfo (STRING): MIME-codierter Puffer, der Informationen enthält

NamespaceInfoType (STRING): enthält Beschreibung der Daten, die im Parameter NamespaceInfo zurückgeliefert werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.ReloadExecutor

**Beschreibung:**

Der Job lädt einen Executor erneut.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Name (STRING): Namespaces, der erneut geladen werden soll

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.UnloadExecutor

**Beschreibung:**

Der Job löscht einen angegebenen Namespace (Executor) aus dem Server.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Name (STRING): Namespaces, der gelöscht werden soll

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Allgemeine Verwaltung

§ [krn.EnumModules](#)

§ [krn.JobThreadBreak](#)

§ [krn.JobThreadGetInfo](#)

§ [krn.QueueEnum](#)

§ [krn.QueueGetParams](#)

§ [krn.QueueGetStatistic](#)

## krn.EnumModules

**Beschreibung:**

Der Job liefert eine Liste der geladenen Module (Bibliotheken). System32-Dlls können bei diesem Vorgang ausgeblendet werden.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

NoSystem32 (BOOL): System32-Dlls ausblenden (0 = nein, 1 = ja)

**Rückgabewerte:**

Module[1..n] (STRING): durch Semikolon getrennte Informationen zu den jeweiligen Modulen

§ Basename: Name des Moduls ohne Pfadangabe

§ FileName: Name des Moduls mit Pfadangabe



- § Version: Version des Moduls
- § Create: Zeitpunkt der Erstellung der Datei
- § Write: Zeitpunkt der letzten Änderung der Datei
- § Access: Zeitpunkt des letzten Zugriffs auf die Datei

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.JobThreadBreak

**Beschreibung:**

Dieser Job unterbricht die Ausführung des laufenden angegebenen Jobs. Der Job kann nur für korrekt laufende und diesen Job unterstützende Jobs benutzt werden.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

ThreadID (INT): ID des Thread

JobNumber (INT): Nummer des Jobs

Queue (STRING): Name der Queue

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.JobThreadGetInfo

**Beschreibung:**

Dieser Job liefert Informationen zum Thread.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Info (STRING): Semikolon-separierte MIME-codierte Informationen zu Threads

Infotyp (STRING): enthält Beschreibung der Daten, die im Parameter Info zurückgeliefert werden

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.QueueEnum

**Beschreibung:**

Der Job erzeugt eine Liste mit den Namen der vorhandenen Queues. Typische Queues sind: common, dbpipe, ocr, workflow und redir.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Queue[1..n] (STRING): Name der jeweiligen Queue

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.QueueGetParams](#) , [krn.QueueGetStatistic](#)

## [krn.QueueGetParams](#)

**Beschreibung:**

Der Job gibt die Parameter einer angegebenen Queue zurück.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Queue (STRING): Name der Queue

**Rückgabewerte:**

NumThreads (INT): Anzahl der Threads

Priority (INT): Priorität

§ 0 = gering

§ 1 = normal

§ 2 = hoch

MaxQueueSize (INT): Maximale Anzahl der Jobs, aufgenommen werden können

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.QueueEnum](#) , [krn.QueueGetStatistic](#)

## [krn.QueueGetStatistic](#)

**Beschreibung:**

Dieser Job liefert statistische Informationen zu einer angegebenen Queue.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Queue (STRING): Name der Queue

**Rückgabewerte:**

JobsPosted (INT): Anzahl der gesandten Jobs

JobsWaiting (INT): Anzahl der wartenden Jobs

LastPop (STRING): Zeitpunkt des letzten Pop

LastPush (STRING): Zeitpunkt des letzten Push

NameSpaces (STRING): Semikolon-separierte Namespaces, die diese Queue nutzen

Threads (STRING): Semikolon-separierte Threads dieser Queue

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[krn.QueueEnum](#)

## Sonstige Jobs

§ [krn.CheckDiskSpace](#)

§ [krn.GetFileVersionList](#)

§ [krn.GetNextIndex](#)

§ [krn.RunScript](#)

§ [krn.SendAdminMail](#)

§ [krn.SendMail](#)

§ [krn.SendMessageToClients](#)

§ [krn.ProcessGetInformation](#)

§ [krn.GetCounter](#)

## krn.CheckDiskSpace

**Beschreibung:**

Dieser Job liefert Informationen zur Kapazität der angegebenen Festplatte.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Disk (STRING): Name der Festplatte

§ ROOT = ermittelt Kapazität vom Laufwerk, auf dem sich das Serververzeichnis befindet

§ DATA = ermittelt Kapazität vom Laufwerk, auf dem sich das Work-Serververzeichnis befindet

§ LOG = ermittelt Kapazität vom Laufwerk auf sich das Log-Serververzeichnis befindet

MinSpace (INT): Minimum Freier Speicher in MB (wird dieser Wert unterschritten, wird der Administrator per E-Mail benachrichtigt; leer = MinSpace wird aus Registry gelesen)

InformAdmin (BOOL): 1 = Administrator wird per E-Mail informiert, ansonsten 0

**Rückgabewerte:**

Total (INT): Größe der Festplatte in MB

Free (INT): Freier Speicher in MB

Min (INT): Minimum Freier Speicher in MB

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.GetFileVersionList

**Beschreibung:**

Dieser Job gibt eine Liste der Dateien des angeforderten Verzeichnisses mit ihren Erstellungszeit-Informationen als Zeichenkette aus. Bei Dlls, Ocx- und Exe-Dateien wird statt der Erstellungszeit der Datei die Versionsnummer ausgegeben.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Directory (STRING): Verzeichnis, die Eingabe '.' entspricht server\etc, '..' entspricht Server-Verzeichnis

**Rückgabewerte:**

FileVersionList (STRING): Zeichenkette, mit den Dateinamen und Erstellungszeit,

Formatierung: [Dateiname]+[Datetime/Versionsnummer]#

[Dateiname2]+[Datetime2Versionsnummer2]#..

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.GetNextIndex

**Beschreibung:**

Dieser Job liefert den nächsten Index aus der Datenbanktabelle 'osnextindex' für alle DB-Einträge, die eine eindeutige ID benötigen.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabewerte:**

Index (INT): der angefragte Index

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.RunScript

**Beschreibung:**

Dieser Job führt ein angegebenes VB-Skript aus. Skripttext kann man als Parameter 'Script' übergeben, als Parameter 'ScriptFile', oder als Dateiliste. In dieser Reihenfolge wird der Skripttext ermittelt.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Script (STRING): auszuführendes VB-Skript

CtxName (STRING): Name des Kontextes; er kann leer sein. In diesem Fall wird Defaultname verwendet.

GUI (BOOL): legt fest, ob MsgBox aus dem Skript aufgerufen werden darf. Ob MsgBox wirklich aufrufbar ist, hängt von der Server-Umgebung ab.

Eval (BOOL): legt fest, ob Eval (falls true) oder Exec (falls false) aufgerufen werden soll.

Main (STRING, optional): Name der Main-Funktion; Defaultwert „Main“

ScriptFile (STRING, optional): Name der Datei mit dem Skripttext

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Krn.EmptyJob

### Beschreibung:

Dieser Job macht nichts. Er kann zur Ausführung von Before- und After-Event-Skripten verwendet werden. Der Job hat keine spezifischen Parameter.

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.SendAdminMail

### Beschreibung:

Der Job sendet eine E-Mail an den voreingestellten Administrator. In der Registry muss dazu unter dem Schlüssel 'HKLM\\Software\\Optimal Systems\\[Applikationsservername] \\Schemata' mindestens die Registry-Einträge Mailserver (SMTP IP-Adresse angeben) und AdminMail (E-Mail-Adresse) gesetzt sein. Die übergebene Dateiliste wird dann zu den Attachments der Mail.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Sender (STRING): Name des Absenders

Subject (STRING): Betreffzeile der Email

Text (STRING): Textinhalt der Email

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Siehe auch:

[Registry-Verwaltung](#)

## krn.SendMail

### Beschreibung:

Der Job sendet eine E-Mail an einen via E-Mail-Adresse identifizierten Empfänger. In der Registry muss dazu unter dem Schlüssel 'HKLM\\Software\\Optimal Systems\\[Applikationsservername] \\Schemata' mindestens der Registry-Eintrag 'Mailserver' (SMTP IP-Adresse angeben) gesetzt sein.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Receiver: (STRING): E-Mail-Adresse des Empfängers

Sender (STRING): Name des Absenders

Subject (STRING): Betreffzeile der E-Mail

Text (STRING): Textinhalt der E-Mail

FileNamePrefix (STRING): (Optional) Präfix für Namen der Dateien (Attachments), die mit einer E-Mail gesendet werden. Wenn ein Präfix angegeben wurde, wird eine Datei, die `xyz.abc` heißt, umbenannt in `<Prefix>.xyz.abc`.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[Registry-Verwaltung](#)

## krn.SendMessageToClients

**Beschreibung:**

Diese Job sendet eine Nachricht an einen oder alle angeschlossenen Clients.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Computer (STRING): leer = alle Computer benachrichtigen, ansonsten ein Computername

Instance (STRING): Name des Programms

User (STRING): Name des Benutzers der benachrichtigt werden soll

Message (STRING): Typ der Nachricht

Info (STRING): Nachrichtentext (Programmtechnische Nachricht)

Text (STRING): Nachrichtentext für Benutzer (z. B. MessageBox)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.ProcessGetInformation

**Beschreibung:**

Diese Job ermittelt Windows Performance-Counter mit Information über Kernel-Process. Es werden Infos nur von dem Server ermittelt, an dem die eigene Session angemeldet ist.

**Parameter:**

Flags (INT): Wenn 0 oder 1, wird die Information in Binärform zurückgeliefert (vorgesehen für Monitor und enaio® enterprise-manager). Wenn 2, wird die Information in Textform zurückgeliefert und dabei im Channel 11 protokolliert.

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## krn.GetCounter

**Beschreibung:**

Diese Job verwaltet Counter in der Tabelle `oscounters`. Es ist möglich, den Wert eines Counters abzufragen, den Counter zurückzusetzen oder zu erstellen. Beim Aufruf des Jobs wird der Wert des Counters hochgesetzt, es sei denn, der Counter soll zurückgesetzt werden. Das Zurücksetzen erfolgt je nach Countertyp. Der Server merkt sich die Zeit des Job-Aufrufs um festzustellen, ob beim nächsten Aufruf der Counter zurückzusetzen ist.

Counter hat einen Typ:

0 = Counter wird manuell per Job-Parameter zurückgesetzt

1 = Counter wird täglich zurückgesetzt

2 = Counter wird monatlich zurückgesetzt

3 = Counter wird jährlich zurückgesetzt

Counter werden mit GUID und Typ identifiziert.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

CounterGUID (STRING): GUID des Counters

CounterType (INT): Typ des Counters

Reset (BOOLEAN): Legt fest, ob der Counter zurückgesetzt werden soll, falls der CounterType 0 ist.

Initial (INT): Wert, mit dem der Counter initialisiert wird, wenn er noch nicht existiert oder zurückgesetzt wird

**Rückgabewerte:**

Counter (INT): Wert des Counters

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Lizenz-Core-Services (Namespace lic)

Dieser Namespace, der neben dem ADM-Executor und KRN-Executor auch im Server-Kernel angesiedelt ist, fasst die Jobs zusammen, welche für die Lizenzverwaltung von enaio® zuständig sind. Für die Benutzung von verschiedenen Funktionen, muss vor der Ausführung ein Login auf die verschiedenen Lizenzen durchgeführt werden.

Insbesondere zur Nutzung der Standard-, DMS- und Workflowengine wird durch den Kernel geprüft, ob sich die aufrufende Instanz authentifiziert und eine Lizenz belegt hat.

Das Login für Lizenzen erfolgt über den Job LicLogin oder LicLoginEx. Hier müssen die benötigten Lizenzstrings (insbesondere ASC, MWC) mit übergeben werden, bevor eine Nutzung der DMS- und Workflowjobs möglich ist.

§ [lic.CheckLicense](#)

§ [lic.LicCopyDefault](#)

§ [lic.LicFreeResource](#)

§ [lic.LicGetGlobalInfo](#)

§ [lic.LicGetGlobalInfoEx](#)

§ [lic.LicGetModuleInfo](#)

§ [lic.LicGetQueueStatus](#)

§ [lic.LicLogin](#)

§ [lic.LicLoginEx](#)

§ [lic.LicLogout](#)

§ [lic.LicLogoutEx](#)

§ [lic.LicResetData](#)

## lic.CheckLicense

### Beschreibung:

Dieser Job prüft, ob die angefragten Module für die Station des enaio® clients lizenziert sind. Die Lizenz wird in der Datenbank nicht gesperrt.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Modules (STRING): verkürzte, durch Leerzeichen getrennte Namen der anzufragenden Module

### Rückgabewerte:

Result (STRING): durch Leerzeichen getrennte Rückgabecodes für die jeweils angefragten Module

§ 0 = für das Modul existiert eine Lizenz

§ 602 = Fehler

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicCopyDefault

### Beschreibung:

Dieser Job verteilt alle Named-Lizenzen, die in der DB-Tabelle 'oslicresources' für die Standard-Station definiert sind, auf alle anderen Stations. Dies geschieht z. B. beim Netzwerksetup.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicFreeResource

### Beschreibung:

Dieser Job gibt eine Resource frei, die in der DB-Tabelle 'osresources' zu finden ist. Zu den Ressourcen gehören: die Module (ADM, M\_X, usw.) und wichtige Systemdateien (aslisten.dat, cfg's, Hintergrundbitmap usw.).

### Parameter:

Flags (INT): z. Z. nicht unterstützt

SessionGUID (STRING): die aktuelle SessionGUID

ResourceID (STRING): die ResourceID der freizumachenden Resource

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode



## lic.LicGetGlobalInfo

### Beschreibung:

Dieser Job liefert den Wert des angegebenen Parameters, welche in den Lizenzdaten (aslic.dat) enthalten sind.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Info (STRING): Name des angefragten Parameters

§ IDENT = Identifikationsmethode

§ ADDRESS = GUID oder IP-Adresse

§ CREATED = Erstelldatum der Lizenz

§ CREATEDFROM = erstellt von

§ CUSTOMER00 = Lizenznehmer

§ LASTMODIFIED = zuletzt geändert am

§ MODIFIEDFROM = geändert von

§ SERVICENAME = Name des Dienstes

§ EXPIRES = gültig bis

### Rückgabewerte:

Result (STRING): Wert des angefragten Parameters

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Siehe auch:

[lic.LicGetGlobalInfoEx](#)

## lic.LicGetGlobalInfoEx

### Beschreibung:

Dieser Job liefert die Werte der angegebenen Parameter, welche in den Lizenzdaten (aslic.dat) enthalten sind.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Info (STRING): Namen der angefragten Parameter mit Format ;Parameter;Parameter;...

§ leer = alle Parameter werden zurückgegeben

§ ? = alle Parameternamen werden zurückgegeben

### Rückgabewerte:

[Parametername] (STRING): Wert des angefragten Parameters

[PARAM[000...nnn]]: Name des Parameters (nur bei Info = ?)

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicGetModuleInfo

### Beschreibung:

Dieser Job ermittelt Informationen (Typ, max. Anzahl der Benutzer) zur Lizenz des angegebenen Moduls.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Module (STRING): verkürzter Name des anzufragenden Moduls

### Rückgabewerte:

Result (STRING): Zeichenkette, die die Lizenzcharakteristiken des angegebenen Moduls beschreibt

§ MaxUseCount: max. Anzahl der Clients, die das Modul nutzen können

§ Typ der Lizenz

§ N = das Modul kann nur von den Clients genutzt werden, die an den bestimmten Arbeitsplätzen arbeiten

§ C = das Modul kann von den Clients der jeweiligen Arbeitsplätze genutzt werden, aber nur von einer bestimmten Anzahl (MaxUseCount) von Clients genutzt werden

§ Anzahl der Konfigurationen, die erstellt werden dürfen

### Rückgabe:

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicGetQueueStatus

### Beschreibung:

Dieser Job liefert Informationen zu Lizenzen für angegebene Module und System-Dateien, welche aktuell vom Server für die angegebenen Stations ausgestellt sind.

### Parameter:

Flags (INT): z. Z. nicht unterstützt

Modules (STRING): verkürzte, durch Leerzeichen getrennte Modulnamen, die angefragt werden sollen (leer = alle Module und System-Dateien)

Stations (STRING): durch Leerzeichen getrennte Stationsnamen, die angefragt werden sollen (leer = alle Stations)

### Rückgabewerte:

Dateiliste: Name und Pfad der Datei; enthält angeforderte Informationen zu den Lizenzen (Dateiformat .rpt)

§ Zeitpunkt der Lizenzvergabe (Zeitstempel)

§ Kurzbezeichnung für Modul/System-Datei

§ Typ der Lizenz (0 = Modul, 1 = System-Datei)

§ Name der Arbeitsstation

§ Name des Benutzers

- § Flags -> z. Z. nicht genutzt
- § Parameter -> z. Z. nicht genutzt
- § ID der Servergruppe
- § GUID der Session
- § ID des Modul/System-Datei

FileCount (INT): immer 1

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicLogin

**Beschreibung:**

Dieser Job erteilt eine Lizenz für das angegebene Modul. Bevor die Clientapplikation einige enaio®-Module nutzen kann, muss sie vom Server die Erlaubnis für ihre Nutzung erhalten. Der Server prüft (z. B. max. Anzahl der Lizenzen, Zugriff auf das Modul nur von bestimmten Arbeitsplätzen), ob der Client berechtigt ist, dieses Modul zu nutzen. Wenn die Prüfung erfolgreich ist, sperrt der Server die Lizenz in der Datenbank.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Module (STRING): Kurzbezeichnung des Moduls

**Rückgabewerte:**

Result (STRING): 0 = Lizenz erteilt, >0 = Fehler

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[lic.LicLoginEx](#) , [lic.LicLogout](#)

## lic.LicLoginEx

**Beschreibung:**

Dieser Job erteilt Lizenzen für die angegebenen Module. Bevor die Clientapplikation einige enaio®-Module nutzen kann, muss sie vom Server die Erlaubnis für ihre Nutzung erhalten. Der Server prüft (z. B. max. Anzahl der Lizenzen, Zugriff auf das Modul nur von bestimmten Arbeitsplätzen), ob der Client berechtigt ist, dieses Modul zu nutzen. Wenn die Prüfung erfolgreich ist, sperrt der Server die Lizenz in der Datenbank.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Module (STRING): durch Leerzeichen getrennte Kurzbezeichnungen der Module

**Rückgabewerte:**

Result (STRING): durch Leerzeichen getrennte Ergebnismeldungen (0 = Lizenz erteilt, >0 = Fehler)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

**Siehe auch:**

[lic.LicLogoutEx](#)

## lic.LicLogout

**Beschreibung:**

Dieser Job gibt eine Lizenz auf ein angegebenes Modul frei, welches vorher mittels LicLogin oder LicLoginEx gesperrt wurde.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Module (STRING): Kurzbezeichnung des Moduls

**Rückgabewerte:**

Result (STRING): 0 = Lizenz freigegeben, >0 = Fehler

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicLogoutEx

**Beschreibung:**

Dieser Job gibt Lizenzen auf mehrere Module frei, welche vorher mittels LicLogin oder LicLoginEx gesperrt wurden.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

Modules (STRING): durch Leerzeichen getrennte Kurzbezeichnungen der Module

**Rückgabewerte:**

Result (STRING): durch Leerzeichen getrennte Ergebnismeldungen (0 = Lizenz erteilt, >0 = Fehler)

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## lic.LicResetData

**Beschreibung:**

Dieser Job veranlasst den Server erneut Lizenzinformation (aslic.dat) aus der Datenbank (oslicense) zu lesen. Der Job wird den aktiven Servern der Servergruppe von dem Server gesendet, der die Lizenzinformation geändert hat. Die Server, die diese Information erhalten, sollen ihre inneren Datenstrukturen modifizieren, die zum Lizenzsystem gehören.

**Parameter:**

Flags (INT): z. Z. nicht unterstützt

**Rückgabe:**

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

## Datenübernahme-Services (Namespace dtr)

Dieser Namespace beinhaltet Jobs für den Aufruf des Datenübernahmeservers.

### Wichtig:

Da der Datenübernahmeserver MS Office verwendet, darf der Applikationsserver nicht unter dem lokalen Systemkonto laufen, sondern muss zur Anmeldung ein Benutzerkonto verwenden.

Für den serverseitigen Aufruf des Datenübernahmeservers wird als Default-Sprache 'Deutsch' verwendet. Über einen Registrierungs-Eintrag für den enaio® Server kann die Sprache angegeben werden:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\OPTIMAL SYSTEMS\MS-Office-Anbindung\OS:4.x-Office-Utilities\
```

Legen Sie dort den DWORD-Wert 'Language' mit dem sprachspezifischen Wert an (vgl. 'Sprachcodes').

### dtr.SynchronizeData

#### Beschreibung

#### Parameter:

Flags (INT): z. Z. nicht unterstützt

ObjectType (INT): Typ des Objekts

ObjectID (INT): Objekt ID

[TemplateAlias] (String): Name des Templates, welches mit den bestehenden Indexdaten gefüllt werden soll. Statt eines Templatenamens kann auch die Templatedatei über die Dateiliste übergeben werden.

[Dateiliste]: Name und Pfad der Vorlagendatei . Alternativ zum Parameter 'Template'

#### Rückgabe:

[Dateiliste]: Name und Pfad des gefüllten Dokuments

(INT): 0 = Job erfolgreich, ansonsten der Fehlercode

### Überwachen der serverseitigen Datenübernahme

Die serverseitige Datenübernahme über 'dtr.SynchronizeData' kann zur Überwachung die Anwendung `oxvbww2servermonitor.exe` aus dem Verzeichnis `\server\` starten, um im Fehlerfall die Datenübernahme zu beenden und ebenfalls die Office-Anwendung zu beenden.

Anwendung und Konfigurationsdatei werden im Verzeichnis `\server\` erst nach dem ersten Aufruf von 'dtr.SynchronizeData' nachinstalliert.

Die Anwendung muss über die Datei `oxvbww2servermonitor.exe.config` konfiguriert werden.

#### Parameter:

```
<add key="prozesskill" value="" />
```

Zeitraum in Sekunden, nach dem die Datenübernahme beendet wird.

0: Funktion wird deaktiviert.

<code>add key="emailto" value="" /&gt;</code>	E-Mail-Adresse, an den im Fehlerfall eine E-Mail gesendet wird. Falls leer, wird keine E-Mail gesendet.
<code>&lt;add key="emailfrom" value="" /&gt;</code>	E-Mail-Absender.
<code>&lt;add key="body" value="body" /&gt;</code>	Text der E-Mail.
<code>&lt;add key="subject" value="" /&gt;</code>	Betreff der E-Mail.
<code>&lt;add key="log" value="" /&gt;</code>	1: Fehlerprotokoll wird geschrieben. 0: Fehlerprotokoll wird nicht geschrieben.
<code>&lt;add key="logfile" value="" /&gt;</code>	Bezeichnung und Pfad für das Protokoll.
<code>&lt;add key="waitprocess" value="" /&gt;</code>	Zeitraum in Sekunden, nach dem die Office-Anwendung beendet wird.
<code>&lt;add key="prozessname" value="" /&gt;</code>	Bezeichnung der Office-Anwendung, die beendet wird. Beispiel: WINWORD

# OxSvrSpt

## Allgemeine Beschreibung

Die Bibliothek `OxSvrSpt.dll` stellt den Zugriff auf die Serverfunktionalitäten über eine Microsoft-COM-Schnittstelle zur Verfügung. Der Schwerpunkt liegt auf der einfachen Handhabbarkeit durch den Benutzer und den VB- und Scripting-konformen COM-Schnittstellen. Dabei wird der easy-to-use-Ansatz verwendet, der es ermöglicht, einfache Aufrufe mittels weniger Codezeilen durchzuführen.

Folgende Eigenschaften werden dabei unterstützt:

- § Fehlerbehandlung über COM-Error (IErrorInfo)
- § Alle aufgetretenen Fehler werden als COM-Fehler geworfen. Liefert ein Aufruf einer Methode mehrere Fehler zurück (z. B. Aufruf eines Server-Jobs), wird der erste Fehler als COM-Fehler geworfen und die übrigen Fehler stehen in einer Sammlung (Collection) zur Verfügung.
- § Transparente Verwendung des Base64-Parameters
- § Die Verarbeitung des Base64-Parameters des Servers wird komplett durch die OxSvrSpt übernommen. Dem Benutzer werden dafür Stream- und XML-Zugriffsmethoden zur Verfügung gestellt.
- § Automatische Kodierung und Dekodierung des Binärparameters (BASE64)
- § Die vom Benutzer an die OxSvrSpt-Bibliothek übergebenen Daten werden automatisch für die Übertragung zum Server MIME-BASE-64-kodiert und beim Abholen der Daten vom Server dekodiert. Der Benutzer kommt mit der Kodierung nicht mehr in Berührung.
- § Kodierung und Dekodierung von XML-Daten durch Bibliothek
- § Über die XML-Eigenschaften der Parameter- und FileParameter-Schnittstellen besteht die Möglichkeit, die Basic-String-Repräsentation (BSTR) der XML-Daten zu lesen und zu setzen. Die OxSvrSpt-Bibliothek übernimmt dabei die richtige Kodierung und Dekodierung in die entsprechende Binärrepräsentation (UTF-8, UTF-16, ...). Die Stream-Methoden dieser Schnittstellen bietet darüber hinaus die Möglichkeit, die Daten direkt über eine IStream-Schnittstelle weiterzuverarbeiten. Es ist damit zum Beispiel möglich die Parameter direkt in ein MSXML-DOM-Dokument zu laden bzw. aus einem solchen zu übernehmen.
- § Automatische Kodierung der Passworte über das Modul
- § Für die Anmeldung besteht die Möglichkeit, das Passwort kodiert, bzw. unkodiert zu übergeben. Die notwendige Umwandlung für die Übertragung zum Server wird durch die OxSvrSpt-Bibliothek übernommen.
- § Collections, die mittels ForEach verwendet werden können
- § Alle COM-Collections sind ForEach-fähig. Dadurch besteht die Möglichkeit aus Visual Basic, Scripting-Umgebungen und .net über diese zu iterieren.
- § Korrekte Verarbeitung der Collections und Parameter im Debugger

- § In Debug-Umgebungen, die direkt auf die Eigenschaften von COM-Objekten zugreifen und diese darstellen können (z. B. Visual Basic) wird sichergestellt, dass die verwendeten Streamzugriffe nicht gestört werden. Die Collections sind VB-tauglich, woraus sich die Möglichkeit ergibt die Elemente im VB-Debugger einzusehen.
- § Rückgabeparameter werden von der aufgerufenen Eigenschaft, bzw. Methode erzeugt
- § Alle Rückgabeparameter werden so zur Verfügung gestellt, dass die Erstellung über eine Methode der `OxSvrSpt`-Bibliothek erfolgt. Es ist nicht notwendig aus einer COM-Scriptumgebung heraus eine Variable zum Befüllen an die Bibliothek zu übergeben. Das Server-Objekt ist mit Ausnahme des Hilfsobjekts das einzige erzeugbare Objekt. Alle anderen Schnittstellen werden von diesem, bzw. untergeordneten Schnittstellen zur Verfügung gestellt.
- § Einfache Verarbeitung von Benachrichtigungen
- § Für die Verarbeitung von Benachrichtigungen steht eine Event-Schnittstelle zur Verfügung, die äquivalent zur Job-Schnittstelle die Verarbeitung der Ein- und Ausgabeparameter übernimmt.
- § Methoden zum Lesen und Schreiben von ASCII-Daten aus Streams
- § Wenn aus Visual Basic bzw. COM-basierten Scriptumgebungen heraus versucht wird Basic-Zeichenketten zu serialisieren, werden die Daten als Widechars verarbeitet. Über die zur Verfügung gestellten Methoden besteht die Möglichkeit Daten, bei denen es sich nicht um XML handelt aus Base64-Parametern als ASCII zu lesen und zu schreiben.

## Module

### Einbinden der Bibliothek

Dieser Abschnitt zeigt die Verwendung der `OxSvrSpt`-Bibliothek aus unterschiedlichen Programmiersprachen heraus. Abhängig von der verwendeten Programmiersprache und Entwicklungsumgebung unterscheiden sich die Möglichkeiten auf die Bibliothek zuzugreifen.

#### Visual Basic

Für die Verwendung der `OxSvrSpt` über die Typbibliothek ist die enaio® Serverzugriffsbibliothek über den "Verweise"-Menüpunkt der Entwicklungsumgebung einzubinden.

Die Erstellung erfolgt über folgenden Code:

```
Private m_oServer As new OxSvrSpt.Server
```

Die Komponente kann auch mittels Late-Binding eingebunden werden. Der Quellcode dafür entspricht dem VB-Script-Code.

#### Visual Basic Script

In VB-Script kann das Server-Objekt folgendermaßen erzeugt werden:

```
Dim oServer  
Set oServer = CreateObject( "OxSvrSpt.Server" )
```

#### Visual C++

Für den Einsatz der `OxSvrSpt`-Bibliothek in Visual C++ wird die Verwendung der `import`-Direktive empfohlen. Der folgende Quellcodeabschnitt zeigt diesen Import. Dies sollte an einer zentralen Stelle, z. B. der `StdAfx.h` erfolgen.



```
// warning C4192: automatischer Ausschluss von 'IStream'
// waehrend des Importierens der Typbibliothek 'OxSvrSpt.tlb'
#pragma warning ( disable: 4192 )
#import "OxSvrSpt.dll" raw_method_prefix("raw_") rename_namespace( "OxSvrSpt" )
#pragma warning ( default: 4192 )
```

Für die Verwendung sollten nicht die `raw`-Methoden, sondern die `high`-Methoden verwendet werden. Um Verwechslungen zu vermeiden, werden deshalb in der obigen `import`-Direktive die `raw`-Funktionen mit dem Präfix `raw_` versehen. Die `high`-Methoden mappen die COM-Fehler automatisch auf `_com_error`-Ausnahmebehandlungen. Zusätzlich stellen diese sicher, dass übergebene Basic-Zeichenketten vom richtigen Typ sind. Der folgende Quellcodeausschnitt zeigt die direkte Verwendung einer `Widechar`-Zeichenketten als Übergabeparameter, anstelle einer Basic-Zeichenkette. Da der Zugriff auf eine Basic-Zeichenkette in C++ dem einer `WIDECHAR`-Zeichenkette entspricht ist dies kompilierbar und läuft auch so lange, wie sich der COM-Client und der COM-Server im selben Apartment befinden.

### Davon darf jedoch nicht ausgegangen werden!

Ein derartiger Aufruf kann zu schwer lokalisierbaren Fehlern führen.

```
HRESULT Test( BSTR Message );
...
HRESULT hr = Test( L"irgendwas" );
```

Wird anstelle der `raw`-Methode jedoch die `high`-Methode verwendet, sorgt die `_bstr_t`-Klasse für die korrekte Verarbeitung.

```
HRESULT Test(_bstr_t Message );
...
HRESULT hr = Test( L"irgendwas" );
```

### Hinweis:

Bei der Umsetzung von `high`-Methoden ohne `retval`-Rückgabewert deklariert die `import`-Direktive diese Methoden als `HRESULT` anstelle von `void`. Dieses `HRESULT` liefert jedoch nie einen Fehlerwert, da im Fehlerfall eine `_com_error`-Ausnahmebedingung geworfen wird. Bei diesen Methoden ähnelt der Aufbau stark dem der `raw`-Funktionen. Auch aus diesem Grund sollte, um Verwechslungen zu vermeiden, der `raw_method_prefix` verwendet werden.

Die Erstellung des Objekts sollte über den Konstruktor des `IServerPtr` und nicht über die Methode `CreateInstance`, da letztere beim Fehlschlagen keine `_com_error`-Ausnahmebedingung wirft. Es ist in diesem Fall ansonsten notwendig, den `HRESULT`-Wert selber auszuwerten, um eine sprechende Fehlermeldung zu erhalten.

```
try
{
OxSvrSpt::IServerPtr spServer( __uuidof( OxSvrSpt::Server ) );
}
catch( _com_error& ex )
{
// hier erfolgt die Fehlerbehandlung
}
```

Alternativ kann die Erstellung des `IServer`-Objekts auch über den Namen der Co-Klasse erfolgen. Die entspricht dann in etwa dem Late-Binding bei der Objekterstellung in Visual Basic.

```

try
{
OxSvrSpt::IServerPtr spServer( "OxSvrSpt::Server" );
}
catch( _com_error& ex )
{
// hier erfolgt die Fehlerbehandlung
}

```

### Microsoft C#

Für die Verwendung der Komponente muss der Verweis enaio® Serverzugriffsbibliothek im Microsoft Visual Studio hinzugefügt werden. Dafür klicken Sie auf **Projekt>Verweis hinzufügen>COM**. Dann kann der Namensraum OxSvrSpt mit der using-Direktive eingebunden werden.

```
Server server = new Server();
```

## Anmeldung

Dieser Abschnitt zeigt die Verwendung des ISession-Objekts in verschiedenen Programmiersprachen. Es werden Anmeldeverfahren vorgestellt, mit denen Sie sich mittels eines unverschlüsselten und verschlüsselten Passworts, einer Sitzungs-GUID und dem NT-Benutzernamen am enaio®-Server anmelden können.

### Möglichkeiten zur Anmeldung

Um sich am System anzumelden, muss ein Objekt der Klasse IServer erstellt worden sein. Dieses stellt Methoden, wie zum Beispiel Login() zur Verfügung, über die Sie sich anmelden können. Bei einer erfolgreichen Authentifizierung liefern die Methoden ein ISession-Objekt zurück. Schlägt die Authentifizierung fehl, wird eine COM-Ausnahmebedingung geworfen. Übergibt man für die Parameter "Benutzername" und "Passwort" eine leere Zeichenkette, dann versucht die Bibliothek eine Anmeldung mit dem NT-Benutzer. Die automatische Anmeldung ist möglich, wenn diese im enaio® administrator aktiviert wurde. NTLM-Authentifikation wird derzeit noch nicht implementiert.

- § Die Login()-Methode dient zur Anmeldung des angegebenen Benutzers am angegebenen Server.
- § Die LoginGUID()-Methode dient zur Anmeldung mit einer bestehenden SessionGUID.
- § Die LoginBalanced()-Methode dient zur Anmeldung an eine Gruppe von Servern. Jeder mögliche Server der Liste besteht aus dem Servernamen, dem Port und einer Wichtung. Die Wichtung gibt an, mit welcher Wahrscheinlichkeit die Verbindung zum zugehörigen Server aufgebaut wird. Die Wichtungen sollten in ihrer Summe nicht größer als 100 sein.
- § Die OpenSession()-Methode dient zur Anmeldung an eine bestehende DefaultSession. DefaultSessions können mit den zuvor beschriebenen Methoden erstellt werden, indem Sie den Parameter DefaultSession auf True setzt.

### Visual Basic und VB-Script

Die Login()-Methode bekommt die Parameter "Benutzername", "Passwort", "Server" und "Port" übergeben. In diesem Fall werden "Passworttyp" und DefaultSession automatisch auf false gesetzt.

```

Dim session As session
' Anmeldung eines Benutzers am angegebenen Server
Set session = server.Login("root", "optimal", "localhost", "4000")
' Anmeldung über eine bestehende SessionGUID
Set session = server.LoginGUID("D57D21256EFB4C91B79EDD5A4928400B", "localhost",
"4000")
' Anmeldung eines Benutzers an einer Gruppe von Servern
Set session = server.LoginBalanced("root", "optimal",
"localhost#4000#90;10.1.3.100#4600#10")

```

## Visual C++

In C++ können die Parameter "Passworttyp" und "DefaultSession" nicht weggelassen werden, da C++ keine Standardwerte unterstützt. Die folgenden Beispiele zeigen die Anmeldung mit einem verschlüsselten Passwort.

```

// Anmeldung eines Benutzers am angegebenen Server
OxSvrSpt::ISessionPtr spSession = spServer->Login("root",
"HB01601611651521561421550000000000000000000000000000", "localhost",
"4000", OxSvrSpt::PasswortTypeEnum::pwEncrypted, false );
// Anmeldung über eine bestehende SessionGUID
OxSvrSpt::ISessionPtr spSession = spServer->
LoginGUID("D57D21256EFB4C91B79EDD5A4928400B",
"localhost", "4000", false );
// Anmeldung eines Benutzers an einer Gruppe von Servern
OxSvrSpt::ISessionPtr spSession = spServer->LoginBalanced("root",
"HB01601611651521561421550000000000000000000000000000",
"localhost#4000#90;10.1.3.100#4600#10",
OxSvrSpt::PasswortTypeEnum::pwEncrypted, false );
// Eine DefaultSession erstellen (Parameter DefaultSession = true)
OxSvrSpt::ISessionPtr spDefaultSession = spServer->Login("root",
"HB01601611651521561421550000000000000000000000000000", "localhost",
"4000", OxSvrSpt::PasswortTypeEnum::pwEncrypted, true );
// An eine DefaultSession anmelden
OxSvrSpt::ISessionPtr spSession = spServer->OpenSession(
(bstr_t)spDefaultSession->Properties->Item["SessionGUID"]->Value,
OxSvrSpt.OpenSession");

```

## Visual C#

In C# können die Parameter "Passworttyp" und "DefaultSession" nicht weggelassen werden. In diesem Beispiel soll die Anmeldung mit dem NT-Benutzernamen gemacht werden, weswegen leere Zeichenketten für die Parameter "Benutzername" und "Passwort" übergeben werden.

```
// Anmeldung eines Benutzers am angegebenen Server
Session session = server.Login("", "", "localhost", "4000",
PasswortTypeEnum.pwNotEncrypted, false);
// Anmeldung über eine bestehende SessionGUID
Session session = server.LoginGUID("D57D21256EFB4C91B79EDD5A4928400B",
"localhost", "4000", false);
// Anmeldung eines Benutzers an einer Gruppe von Servern
Session session = server.LoginBalanced("", "",
"localhost#4000#90;10.1.3.100#4600#10",
PasswortTypeEnum.pwNotEncrypted, false);
// Eine DefaultSession erstellen (Parameter DefaultSession = true)
Session defaultSession = server.Login("root", "optimal", "localhost", "4000",
PasswortTypeEnum.pwNotEncrypted, true);
// An eine DefaultSession anmelden
Session session =
server.OpenSession(defaultSession.Properties["SessionGUID"].Value.ToString(),
"OxSvrSpt.OpenSession");
```

## Lizenzverwaltung

Für das Prüfen und die Verwendung von Lizenzen steht im `ISession`-Objekt die `ILicenses`-Collection zur Verfügung. Über diese können die zu verwendenden Lizenzen beim Server angemeldet, abgemeldet und überprüft werden. Alle angemeldeten Lizenzen werden in der Collection behalten.

```
/*
JavaScript-Version
Diese ermöglicht im Gegensatz zur VBS-Version die Verwendung der
COM-Exceptions. Beim Auftreten eines Fehlers wird der Programmfluss
abgebrochen und zur Fehlerbehandlung gesprungen.
*/
try
{
var oServer, oSession, oJob;
// Zugriffsobjekt erstellen und einloggen
oServer = new ActiveXObject( "OxSvrSpt.Server" );
oSession = oServer.Login( "root", "optimal", "localhost", "4000" );
// Lizenzen hinzufügen und löschen
oSession.Licenses.Add( "ASC" );
oSession.Licenses.Delete( "ASC" );
oSession.Licenses.Add( "ASC" );
// hat geklappt:o
WScript.Echo( "ok" );
}
catch( ex )
{
// aufgetretene Fehler ausgeben
WScript.Echo( ex.description );
}
```

## Serverereignisse

### Beschreibung

Sie haben die Möglichkeit über `Notifications` vom Server über bestimmte Ereignisse informiert zu werden.

### VB

In Visual Basic stehen die `Notifications` über den Ereignismechanismus zur Verfügung, wie im folgenden Beispiel dargestellt.

```

Private WithEvents m_oSession As OxSvrSpt.Session
Private m_oServer As OxSvrSpt.Server
public sub Start()
m_oServer.Properties("NotifyNeeded") = True
Set m_oSession = m_oServer.Login( , , "localhost", "4000")
end sub
Private Sub m_oSession_Notify(Job As OxSvrSpt.INotifyJob)
On Error GoTo ErrTrap
Dim strFileXML As String
strFileXML = Job.InputFileParameters(1).XML
' Rückgabeparameter
Job.OutputParameters.AddNewIntegerParameter "test", 100
Job.OutputParameters.AddNewStringParameter "meier", "huhu"
Exit Sub
ErrTrap:
MsgBox Err.Description
End Sub

```

## VBScript

Wenn Sie in einer Scripting-Host-Umgebung (VB-Script als vbs) Notifications verwendet möchten, müssen Sie die Instanz über die `CreateObject()`-Methode der WScript-Umgebung erzeugen.

```

Dim oServer
Set oServer = WScript.CreateObject( "OxSvrSpt.Server" , "oServer_" )

```

Wird das `IServer`-Objekt über den `WScript`-Host erstellt, besteht die Möglichkeit ein Präfix für Ereignis-Funktionen mit anzugeben. Über diesen Mechanismus können die Notifications abgefangen werden.

Weiterhin stehen die Callback-Funktionalitäten der OxSvrCom-Bibliothek über die Methoden `CreateJobSink()` und `CreateJobSink()` der `ISession`-Schnittstelle zur Verfügung.

```
Dim oServer, oSession
Set oServer = WScript.CreateObject( "OxSvrSpt.Server" , "oServer_" )
' Notifications sollen verwendet werden
oServer.Properties("NotifyNeeded") = True
' Auto-Login verwenden
set oSession = oServer.Login( , , "localhost", "4000" )
...
'
' diese Funktion wird vom Server-Objekt aufgerufen, wenn innerhalb der
' angegebenen Ruhezeit eine Notification eintrifft
'
' @param oJob
'         beinhaltet die Daten für den Aufruf der Notification.
'         Diese entsprechen dem Job-Objekt der Session. Der einzige
'         Unterschied besteht darin, dass nicht die Eingangs-,
'         sondern die Ausgangsparameter die Methoden zum Erstellen
'         von Parametern aufweisen.
'
Sub oServer_Notify( oJob )
Dim strText
' Name der Notification ausgeben
strText = "Name: " + oJob.Name + vbCrLf
' alle Eingangsparameter ausgeben
strText = strText + "Parameter:" + vbCrLf
Dim oParameter
For Each oParameter In oJob.InputParameters
strText = strText + "    " + oParameter.Name + " - " + CStr( oParameter.Value ) +
vbCrLf
next
' Ergebnis in eine Textbox schreiben
MsgBox strText
End sub
```

## XML-Verarbeitung

Abhängig von der verwendeten XML-Kodierung unterscheiden sich die XML-Daten in ihrer binären Darstellung.

Beispiel zur Ermittlung der Objektdefinition

JavaScript-Version

```

/*
JavaScript-Version
Diese ermöglicht im Gegensatz zur VBS-Version die Verwendung der
COM-Exceptions. Beim Auftreten eines Fehlers wird der Programmfluss
abgebrochen und zur Fehlerbehandlung gesprungen.
*/
try
{
var oServer, oSession, oJob;
// Zugriffsobjekt erstellen und einloggen
oServer = new ActiveXObject( "OxSvrSpt.Server" );
oSession = oServer.Login( "root", "optimal", "localhost", "4000" );
// Job für die Ermittlung der Objektdefinition erstellen
oJob = oSession.NewJob( "dms.GetObjDef" );
// Anfrageparameter setzen
oJob.InputParameters.AddNewIntegerParameter( "Flags", 0 );
// Job ausführen
oJob.Execute();
// XML aus der Ausgabedatei ermitteln
// Beim Auslesen wird das XML-Zeichenkodierung
// berücksichtigt.
// Nach der Beendigung dieses Aufrufs wird die übergebene
// Datei automatisch gelöscht.
WScript.Echo( oJob.OutputFileParameters(1).XML );
}
catch( ex )
{
// aufgetretene Fehler ausgeben
WScript.Echo( ex.description );
}

```

### VB-Scriptversion

```

Option Explicit
Dim oServer, oSession, oJob, o
Set oServer = CreateObject( "OxSvrSpt.Server" )
set oSession = oServer.Login( "root", "optimal", "localhost", "4000" )
set oJob = oSession.NewJob( "dms.GetObjDef" )
oJob.InputParameters.AddNewIntegerParameter "Flags", 0
oJob.Execute
msgbox oJob.OutputFileParameters(1).XML, "Objektdefinition"

```

## Binärdatenverarbeitung

Für die Verarbeitung von Binärdaten in den Ein- und Ausgabeparametern stehen mehrere Möglichkeiten zur Verfügung. Sowohl die `IParameter`-, als auch die `IFileParameter`-Objekte stellen hierfür zwei unterschiedliche Verfahren zur Verfügung.

- § 1. Zugriff über Chunks (Byte-Arrays)
- § 2. Zugriff über einen Stream (IStream)

```

'
' Beispiel-Script zum byteweisen Auslesen von Binärdaten über die
' GetChunk-Methode eines Parameters.
'
Option Explicit
Dim oServer, oSession, oJob, oFileParameter
Set oServer = CreateObject("OxSvrSpt.Server")
set oSession = oServer.Login("root", "optimal", "localhost", 4000)
Set oJob = oSession.NewJob("dummy")
Set oFileParameter = oJob.InputFileParameters.AddTempFile()
oFileParameter.xml = "<?xml version='1.0' encoding='utf-16' ?><abc>äöü</abc>"
' Stream zum Lesen auf die Anfangsposition setzen
oFileParameter.ResetStream
Dim abReadData
abReadData = oFileParameter.GetChunk(oFileParameter.ActualSize)
Dim cPos
For cPos = LBound(abReadData) + 1 To UBound(abReadData)
Dim bData
bData = AscB(MidB(abReadData, cPos, 1))
WScript.Echo(cPos & ":" & bData)
next
WScript.Echo("fertig")

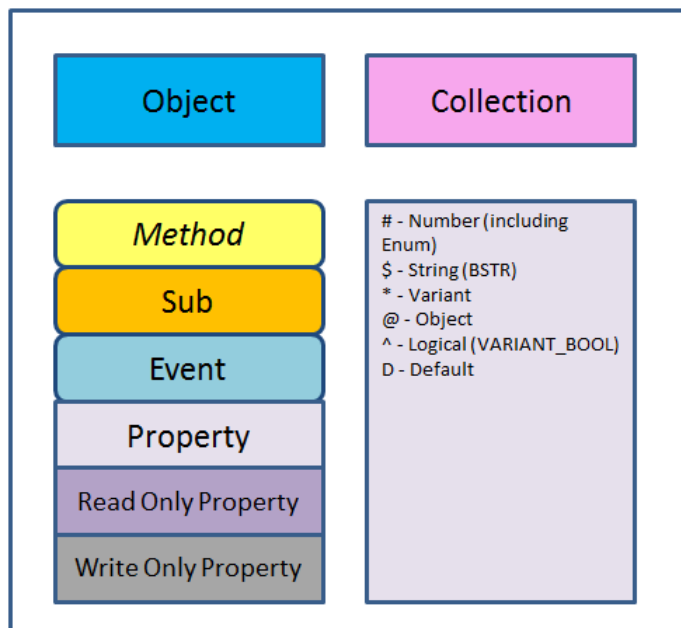
```

## Fehlerbehandlung

In allen Fehlerfällen wirft die OxSvrSpt-Bibliothek einen COM-Fehler. Darüber hinaus werden die Fehler sowohl beim `IServer`-, als auch beim `IJob`-Objekt zusätzliche in die `ICollection` eingetragen. In beiden Fällen besteht die Möglichkeit, dass mehr als ein Fehler zurückgeliefert wird. Der COM-Fehler entspricht dabei immer dem ersten Fehler in der `ICollection`.

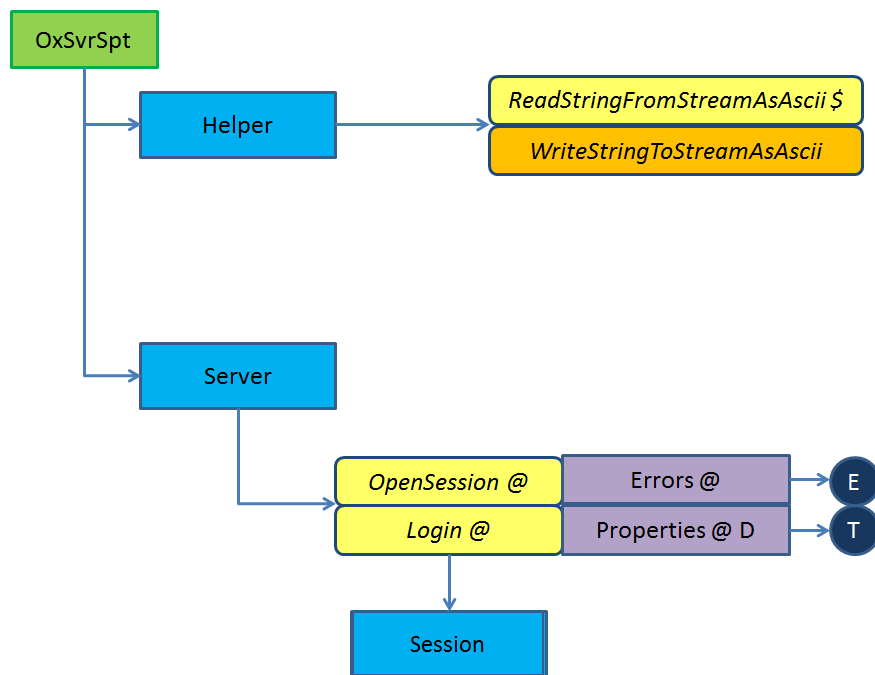
## Aufbauplan

Die folgenden Abbildungen zeigen die Struktur der OxSvrSpt:

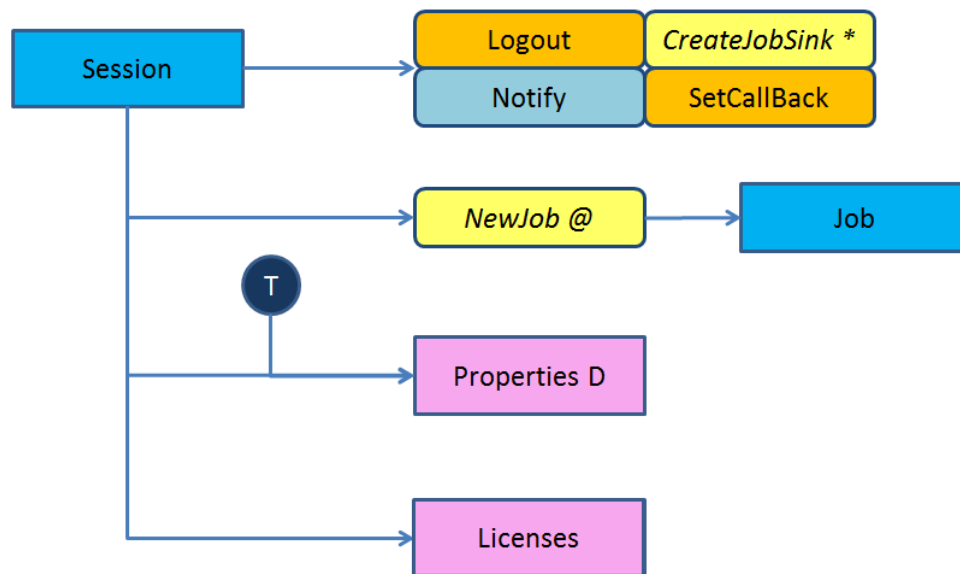


Legende für den Ablaufplan

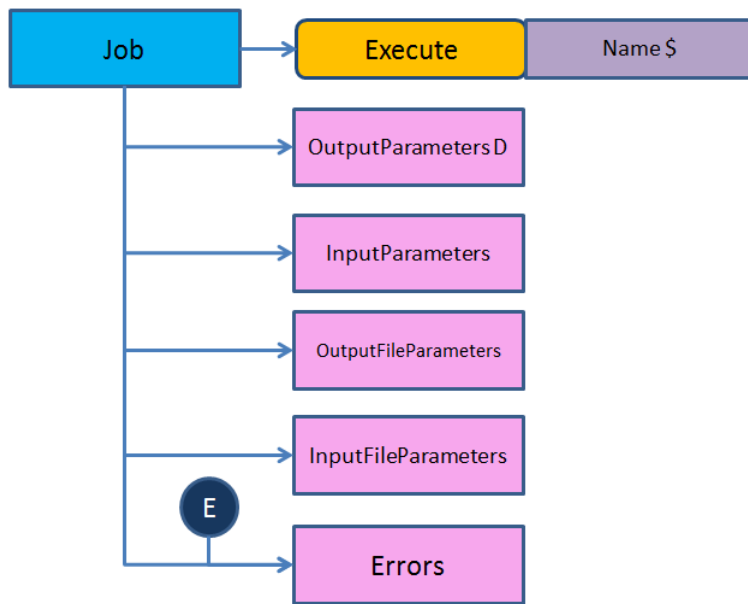




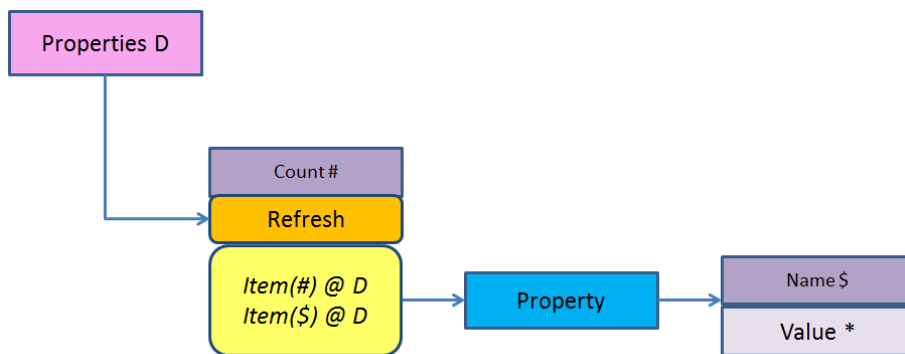
#### OxSvrSpt-Bibliothek



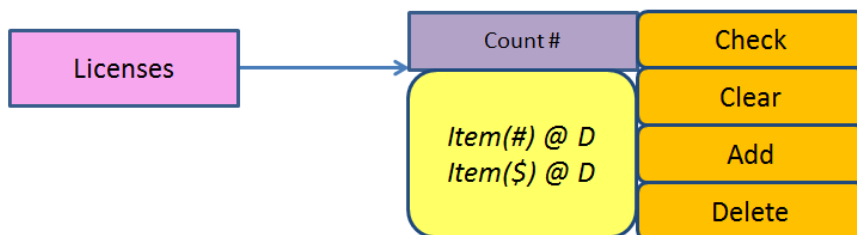
#### Sessoin-Objekt



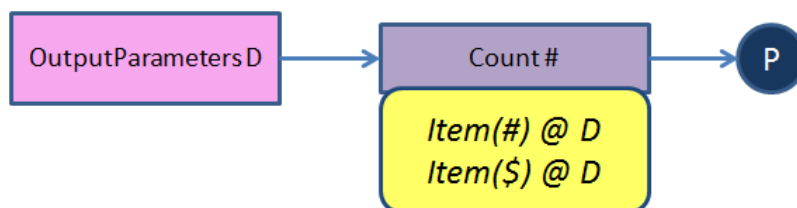
#### Job-Objekt



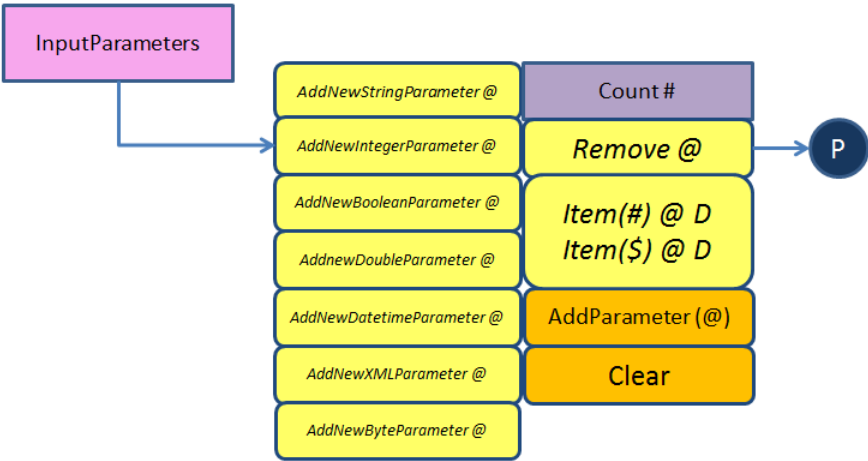
#### Properties D-Collection



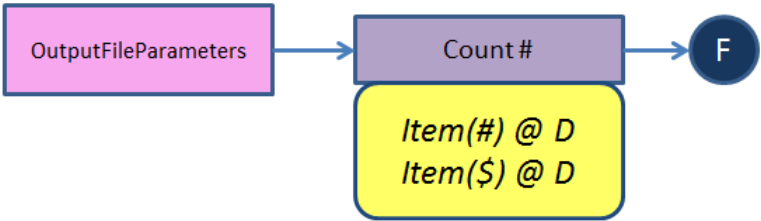
#### Licenses-Collection



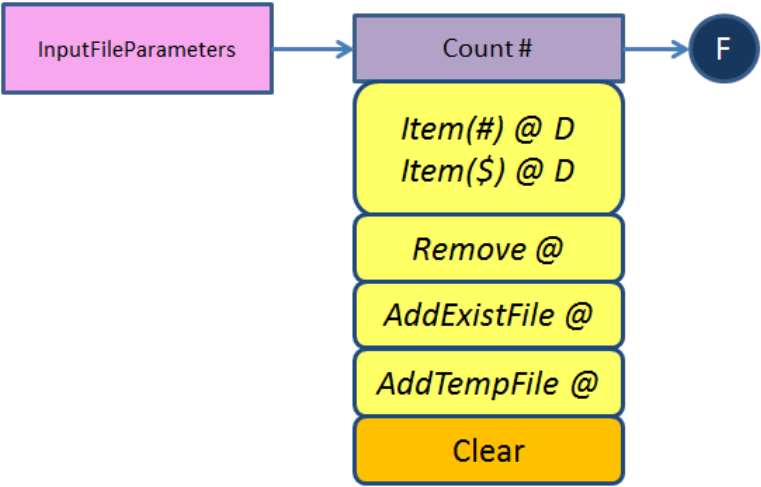
#### OutputParameters D-Collection



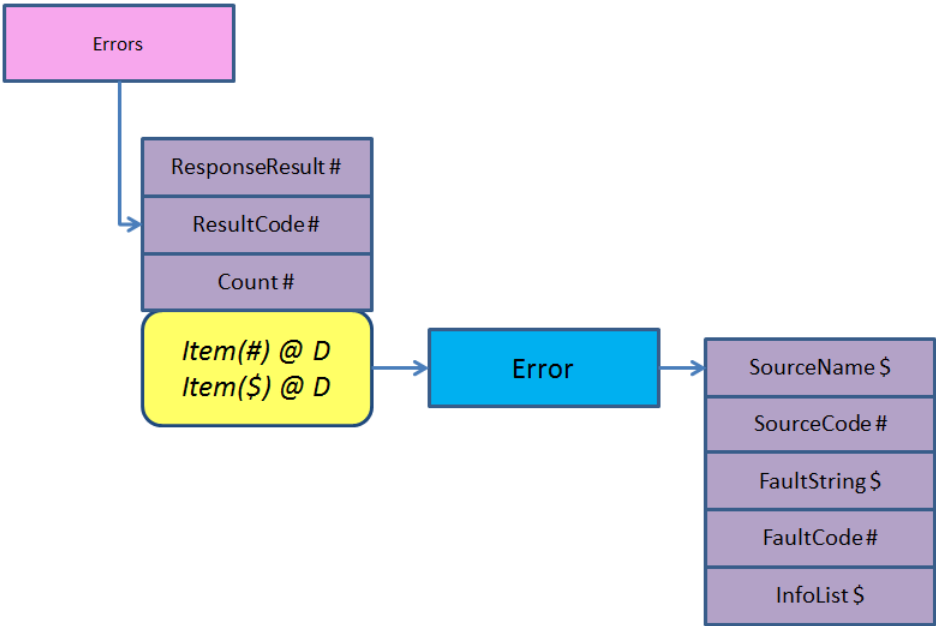
InputParameters-Collection



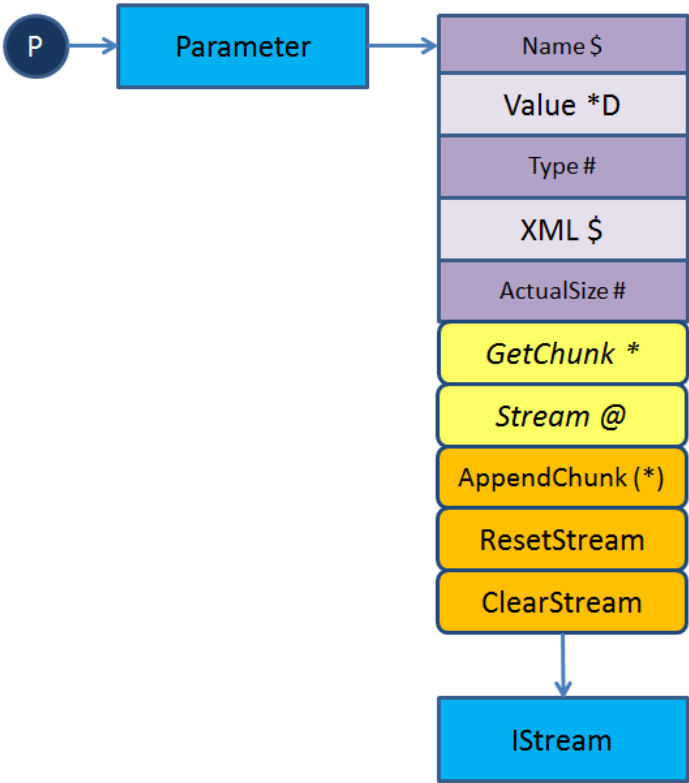
OutputFileParameters-Collection



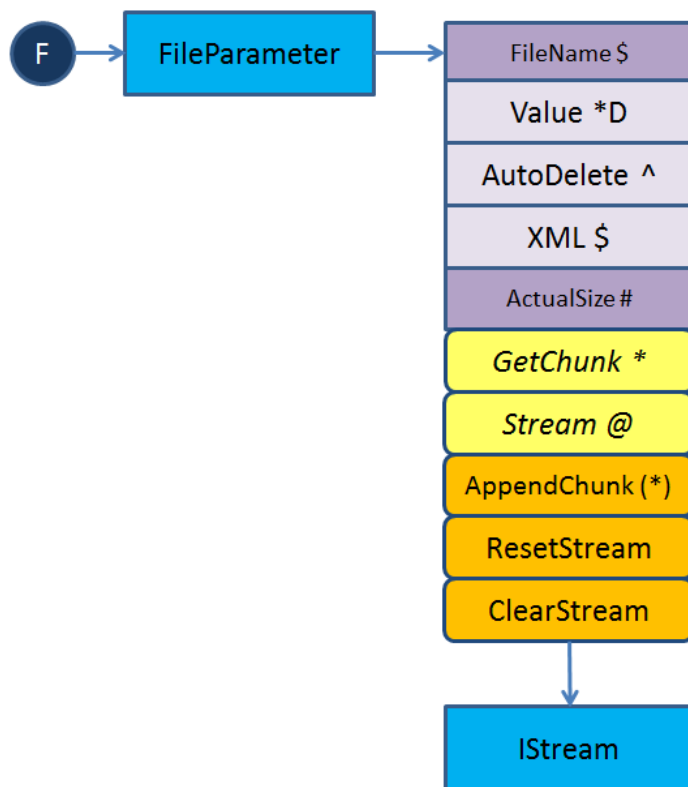
InputFileParameters-Collection



Errors-Collection



Parameter-Objekt



FileParameter-Objekt

## Anbringen von Wasserzeichen an PDF-Dokumente

Alternativ zu der Wasserzeichendruckkennzeichnung von PDF-Dokumenten über die Einstellungen in enaio® administrator, gibt es die Möglichkeit Wasserzeichen mit Nutzung von erweiterten Möglichkeiten anzubringen. Dazu muss als Zielformat 'pdf' angegeben werden und der Job-Parameter 'Watermark' mit 1 gesetzt sein. Alle folgenden Parameter bestehen aus einem Präfix und einem Postfix, der Parameter 'HeaderText' hat also den Präfix 'Header' und den Postfix 'Text'. Die Parameter sind im Folgenden detailliert beschrieben:

Es können vier Bereiche für Textangaben definiert werden mit folgenden Parameter-Präfixen: 'Header', 'Footer', 'Side' und 'Center'.

Die folgenden Parameter-Postfixe sind möglich bei denen jeweils einer der oben genannten Präfixen verwendet werden muss. Alle Parameter-Postfixe sind optional, aber mindestens ein '\*Text' Parameter muss angegeben werden, damit das erweiterte Wasserzeichen angebracht wird, ansonsten werden die enaio® administrator Einstellungen verwendet. Im Folgenden sind die Postfixe aufgeführt:

Text (STRING):

Text der aufgebracht werden soll. Wird kein Text angegeben werden alle anderen Angaben zu dem entsprechenden Wasserzeichentyp ignoriert und kein Text aufgebracht.

Folgende Ersetzungsvariablen sind im angegebenen Text möglich:

Variable	Beschreibung
----------	--------------

#ZEIT#	aktuelle Uhrzeit
#DATUM#	aktuelles Datum
#BENUTZER#	Benutzername
#BENUTZER-VOLLSTAENDIG#	vollständiger Benutzername
#COMPUTER-IP#	IP-Adresse
#COMPUTER-GUID#	GUID-des Rechners
#COMPUTER-NAME#	Name des Rechners

TextColor (INT):

0 – 7 (Standard ist 0) dabei gilt:

Wert	Beschreibung
------	--------------

0	Schwarz
1	Weiß
2	Gelb
3	Rot
4	Grün
5	Magenta
6	Cyan
7	Blau

Alternativ dazu:

TextColorRGB (STRING): 0-255,0-255,0-255 (R,G,B-Werte für eine Farbe durch Komma getrennt)

Font (STRING): Helvetica..., Times..., oder Courier... (Standard ist Helvetica)

FontBold (INT): 0 oder 1 ( 1-> Bold, 0 -> Standard)

FontItalic (INT): 0 oder 1 ( 1-> Italic, 0 -> Standard)

FontSize (INT): Fontgröße in Punkten (Standard ist 10)

Über folgende Parameter kann die Position des Texts festgelegt werden:

Position (INT): 0-8 (Standard ist 0) für 'Header' 0, für 'Footer' 1, für 'Side' 4 und für 'Center' 8

Wert	Beschreibung
0	Links-Oben
1	Links-Unten
2	Rechts-Oben
3	Rechts-Unten
4	Links-Zentriert
5	Rechts-Zentriert
6	Zentriert-Oben
7	Zentriert-Unten
8	Zentriert-Zentriert

OffsetX (INT): (Standard ist 0) Offset in mm bezüglich linke Blattkante.

OffsetY (INT): (Standard ist 0) Offset in mm bezüglich obere Blattkante.

PlaceType (INT): 0-2 (Standard ist 0) 0 -> Alle Seiten, 1 -> ungerade Seiten, 2 -> gerade Seiten

FillStyle (INT): 0-2 (Standard ist 0) 0 -> Gefüllt, 1 -> nur Schattenriss, 2 -> gefüllter Schattenriss

Angle (INT): 0-360 (Standard ist 0) Drehung in Grad

Beispiel für einen Aufruf:

```
Set oServerJob = o.CreateServerJob("cnv.ConvertDocument")
oServerJob.AddFile "myfile" (Datei die konvertiert werden soll)
oServerJob.AddInputParameter "DestinationFormat", "pdf", 1
oServerJob.AddInputParameter "HeaderText", "Kopfzeile", 1
oServerJob.AddInputParameter "HeaderFont", "Courier", 1
oServerJob.AddInputParameter "HeaderFontSize", "10", 2
oServerJob.AddInputParameter "HeaderTextColor", "3", 2
oServerJob.AddInputParameter "FooterText", " Erstellt durch #BENUTZER#, #DATUM#,
#ZEIT#", 1
oServerJob.AddInputParameter "FooterFontSize", "14", 2
oServerJob.AddInputParameter "FooterFontBold", "1", 2
oServerJob.AddInputParameter "FooterFontItalic", "1", 2
oServerJob.AddInputParameter "FooterTextColorRGB", "255,0,0", 1
```

Anm.: Statt Präfix 'Header' und 'Footer' wie in diesem Beispiel sind auch 'Side' oder 'Center' möglich.

Beispiel für die Positionierung mit Typ 'Center':

```
oServerJob.AddInputParameter "CenterAngel", "45", 2
oServerJob.AddInputParameter "CenterOffsetX", "-10", 2
oServerJob.AddInputParameter "CenterOffsetY", "10", 2
oServerJob.AddInputParameter "CenterPlaceType", "1", 1
oServerJob.AddInputParameter "CenterFillStyle", "2", 1
```

Anm.: Auf Einhaltung von Gross- und Kleinschreibung ist zu achten. (Job-Parameter sind generell Case sensitive).

## Datenstrukturen

### \_INotificationEvents

#### Beschreibung:

\_INotifiactionEvent ist eine Event-Schnittstelle für die Verarbeitung von Notifications.

```
import "OxSvrSpt.idl"
```

#### Öffentliche Methoden:

```
void Notify([in, out] INotifyJob **Job)
```

#### Parameter:

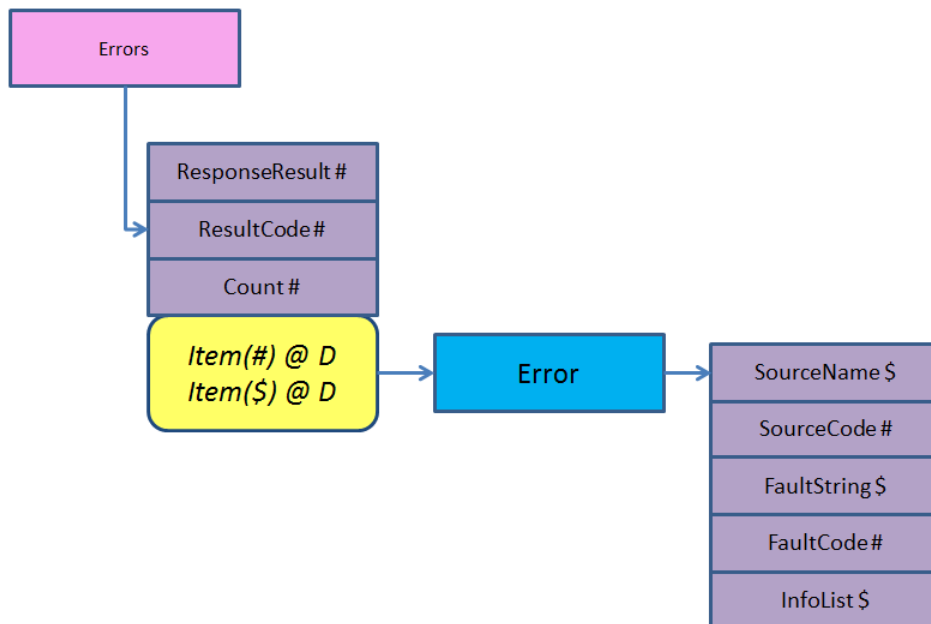
[in, out]: Job Schnittstelle für den Zugriff auf die Ein- und Ausgabeparameter der Notification. Diese verhält sich äquivalent zur IJob-Schnittstelle.

### IError

#### Beschreibung:

IError repräsentiert eine Fehlermeldung des Servers.

```
import "OxSvrSpt.idl"
```



#### Properties:

```
long FaultCode [get]
```

```
BSTR SourceName [get]
```

```
long SourceCode [get]
```

```
BSTR FaultString [get]
```

```
BSTR InfoList [get]
```



**Dokumentation der Properties:**

\$ long FaultCode [get]

FaultCode liefert den Fehlercode zurück.

**Parameter:**

[out]: pVal (VB-Rückgabewert) Fehlerbeschreibung

\$ BSTR SourceName [get]

SourceName liefert den Namen der Quelle zurück, in der der Fehler aufgetreten ist.

**Parameter:**

[out]: pVal (VB-Rückgabewert) Bezeichner der Quelle

\$ long SourceCode [get]

SourceCode liefert die Quellcodezeile, in der der Fehler aufgetreten ist.

**Parameter:**

[out]: pVal (VB-Rückgabewert) Bezeichner der Quelle

\$ BSTR FaultString [get]

FaultString liefert die Fehlerbeschreibung zurück.

**Parameter:**

[out]: pVal (VB-Rückgabewert) Fehlerbeschreibung

\$ BSTR InfoList [get]

InfoList siehe Dokumentation der OxSvrCom.

**Parameter:**

[out]: pVal (VB-Rückgabewert) InfoList

## IErrors

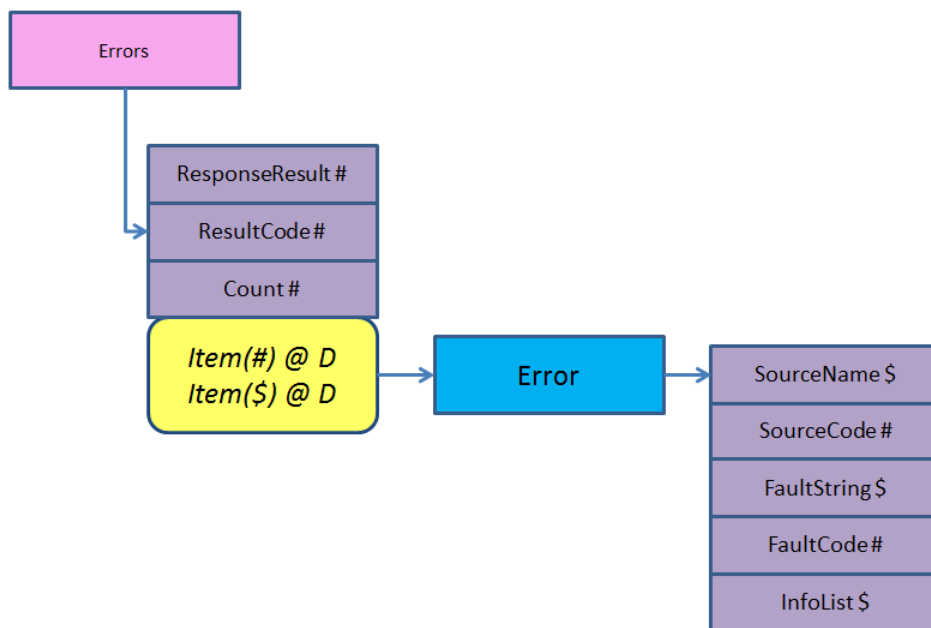
**Beschreibung:**

IErrors ist eine Collection, in der Fehler gesammelt werden, die vom Server bei einer Operation geliefert werden.

```
import "OxSvrSpt.idl"
```

**Hinweis:**

Als Schlüsselwort wird für die einzelnen Einträge der Name "errorX" verwendet, wobei X die Position des Fehlers in der Collection ist. Dieses Schlüsselwort kommt jedoch in den seltensten Fällen zum Einsatz. Anstelle dessen, sollten Sie die Collection entweder mittels `foreach` oder über die Position abgefragt werden.

**Properties:**

```

long Count [get]
long ResponseResult [get]
IError Item([in] VARIANT Index) [get]

```

**Dokumentation der Properties:**

`$ long Count [get]`  
 Count liefert die Anzahl der Elemente in der Collection zurück.

**Parameter:**

[out]: `pINumber` (VB-Rückgabewert) Anzahl der Elemente der Collection.

`$ IError Item([in] VARIANT Index) [get]`  
 Item liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert `errCollectionIndexOutOfRange` zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert `errCollectionItemNotFound` zurückgeliefert.

**Parameter:**

[in]: `Index` Position bzw. Name des angeforderten Elements

[out]: `pptItemzugehöriges JobError-Objekt`

`$ long ResponseResult [get]`  
 ResponseResult liefert den Rückgabewert des Serveraufrufs, der den Fehler ausgelöst hat.

**Parameter:**

[out]: `pVal` (VB-Rückgabewert) Rückgabewert des Serveraufrufs.

`$ long ResultCode [get]`

ResultCode liefert den letzten Fehlerwert des Serveraufrufs.

**Parameter:**

[out]: pVal (VB-Rückgabewert) Fehlerwert des Servers.

## IFileParameter

**Beschreibung:**

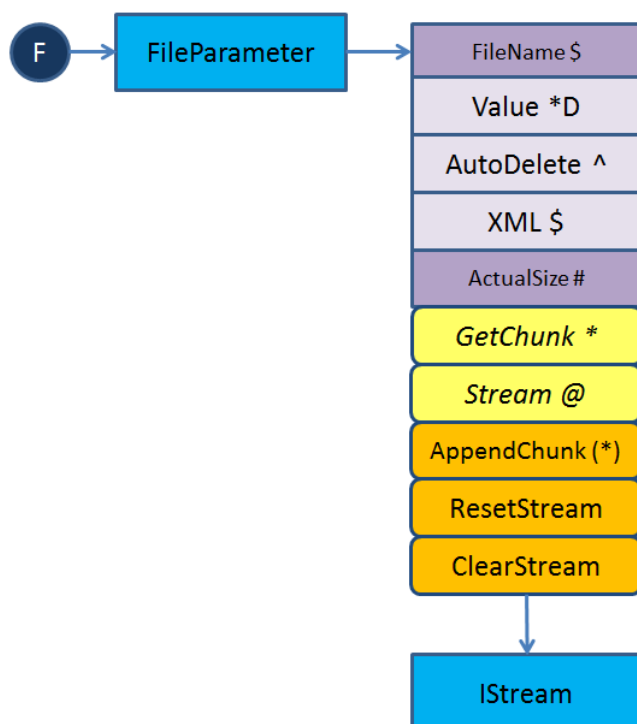
IFileParameter repräsentiert sowohl einen Eingangs-, als auch einen Ausgangs-Parameter eines Jobs, der eine Datei enthält.

```
import "OxSvrSpt.idl"
```

Die IFileParameter-Schnittstelle stellt Methoden für den Zugriff auf den FileParameter und die zugrunde liegende Datei zur Verfügung.

Für den Zugriff auf die Datei stehen ähnlich der IParameter-Schnittstelle sowohl Stream-, Chunk-, sowie XML-Funktionalitäten zur Verfügung.

Solange keine der Datei-Zugriffsfunktionalitäten aufgerufen werden, wird die Datei nicht geöffnet. Beim ersten Zugriff auf eine Methode, die mit der Datei arbeitet, wird diese im modeReadWrite|shareDenyNone-Modus geöffnet. Über UnloadStream ist es möglich, diese ausdrücklich wieder zu schließen, um für den Zugriff durch andere Prozesse einen exklusiven Zugriff zu ermöglichen.



**Öffentliche Methoden:**

```
HRESULT Stream ([out, retval] IStream ** ppStream)
```

```
HRESULT AppendChunk ([in]VARIANT Data)
```

```
HRESULT GetChunk ([in] long Length, [out, retval] VARIANT * pResult)
```

```

HRESULT ResetStream ()
HRESULT ClearStream ()
HRESULT UnloadStream ([in, defaultvalue(-1)] VAIRANT_BOOL AutoReload)

```

### Properties:

```

BSTR FileName [get, set]
long ActualSize [get]
BSTR XML [get, set]
VARIANT_BOOL AutoDelete [get, set]

```

### Dokumentation der Elementfunktionen:

§ HRESULT AppendChunk ([in]VARIANT Data)

AppendChunk hängt zusätzliche Bytes an den Stream an.

Beim ersten Zugriff auf eine der Streamfunktionalitäten der Schnittstelle wird die zugrundeliegende Datei geöffnet und offen gehalten. Wird von einem anderen Aufrufer anschließend ein exklusiver Zugriff auf diese Datei benötigt, kann sie über den Aufruf der Methode `UnloadStream` wieder freigegeben werden.

Nach dem Aufruf dieser Methode steht der Positionszeiger des internen Streams auf dessen Ende.

#### Parameter:

[in]: Data enthält die Daten, die an den Stream angehängt werden sollen.

Die Daten werden über die `OLE32 ChangeType`-Methode nach `VT_ARRAY|VT_UI1` konvertiert und weiterverarbeitet. Wird z. B. ein `BSTR` übergeben, werden die Daten als `WIDECHAR` weiterverarbeitet. Das Schreiben von 8bit-Zeichen in den Stream kann über die Methoden des `Helper-COM`-Objektes erfolgen.

#### Beispiel:

```

Dim abWriteData() As Byte
ReDim abWriteData(0 To 5)
abWriteData(0) = 0
abWriteData(1) = 1
abWriteData(2) = 2
abWriteData(3) = 3
abWriteData(4) = 4
abWriteData(5) = 5
oFileParameter.AppendChunk abWriteData

```

§ HRESULT ClearStream ()

ClearStream löscht die Daten des Streams und der zugrunde liegenden Datei.

§ HRESULT GetChunk ([in] long Length, [out, retval] VARIANT \* pResult)

GetChunk liefert die angegebene Anzahl von Bytes aus dem Stream zurück.

Beim ersten Zugriff auf eine der Streamfunktionalitäten der Schnittstelle wird die zugrundeliegende Datei geöffnet und offen gehalten. Wird von einem anderen Aufrufer anschließend ein exklusiver Zugriff auf diese Datei benötigt, kann sie über den Aufruf der Methode `UnloadStream` wieder freigegeben werden.

Diese Methode beginnt ab der aktuellen Position im Stream die Daten zu lesen. Wird das Ende des Streams erreicht, bevor die gewünschte Anzahl an Zeichen gelesen wurde, werden nur die bis dahin gelesenen Zeichen zurückgeliefert. Die Anzahl der gelesenen Zeichen kann anhand der Größe des zurückgelieferten Puffers bestimmt werden (siehe Beispiel).

**Parameter:**

[in]: Length      Anzahl der maximal zurückgelieferten Bytes.

[out]: pResult      (VB-Rückgabeparameter) enthält die gelesenen Bytes. Diese werden in einem Variant vom Typ VT\_ARRAY|VT\_UI1 geliefert.

**Beispiel:**

```
' der folgende Programm-Ausschnitt entspricht:
' Dim var
' var = oFileParameter.Value
' dort bleibt jedoch zusätzlich die aktuelle Position des Streams erhalten
' Stream zum Lesen auf die Anfangsposition setzen
oFileParameter.ResetStream
Dim abReadData() As Byte
...
' Array auf die benötigte Größe anpassen
ReDim abReadData(0 To oParameter.ActualSize - 1)
' Daten lesen
abReadData = oFileParameter.GetChunk(oParameter.ActualSize)
' Größe der gelesenen Daten anhand der zurückgegebenen Daten ermitteln
Dim nSize As Long
nSize = UBound(abReadData) - LBound(abReadData)
```

§ HRESULT ResetStream ()

ResetStream setzt den Datenstream auf den Anfang zurück.

Beim ersten Zugriff auf eine der Streamfunktionalitäten der Schnittstelle wird die zugrundeliegende Datei geöffnet und offen gehalten. Wird von einem anderen Aufrufer anschließend ein exklusiver Zugriff auf diese Datei benötigt, kann sie über den Aufruf der Methode UnloadStream wieder freigegeben werden.

§ HRESULT Stream ([out, retval] IStream \*\* ppStream)

Stream liefert den Stream der Binärdaten

Der Stream enthält die Binärdaten der Datei.

Beim ersten Zugriff auf eine der Streamfunktionalitäten der Schnittstelle wird die zugrundeliegende Datei geöffnet und offen gehalten. Wird von einem anderen Aufrufer anschließend ein exklusiver Zugriff auf diese Datei benötigt, kann sie über den Aufruf der Methode UnloadStream wieder freigegeben werden.

**Parameter:**

[out]: ppStream      (VB-Rückgabeparameter) enthält den Stream als IStream-Schnittstelle.

§ HRESULT UnloadStream ([in, defaultvalue(-1)] VAIRANT\_BOOL AutoReload)

UnloadStream entlädt das interne Stream-Objekt

Wenn keine externen Referenzen mehr auf das Stream-Objekt verweisen, wird dieses freigegeben und die zugrundeliegende Datei geschlossen. Bei einem erneuten Zugriff über eine Methode, die das Streamobjekt verwendet wird dieser erneut aufgebaut.

Hinweis:

Ist `AutoDelete` auf `true` gestellt und wird der `Stream` entladen, wird wenn keine andere Referenz auf den `Stream` mehr besteht, die zugrunde liegende Datei gelöscht.

**Parameter:**

[in]: `AutoReload` gibt an, ob beim nächsten Zugriff auf eine Methode von `IFileParameter`, die den `Stream` benötigt, dieser wieder automatisch neu initialisiert wird. Vor allem bei der Verwendung von Debug-Umgebungen, die automatisch die betreffenden Eigenschaften neu laden, kann es zu Problemen führen, wenn die Datei exklusiv durch andere Programme weiterverarbeitet werden soll. In diesem Fall sollte der Wert `VARIANT_FALSE` für diesen Parameter verwendet werden. In Sprachen, die Standardparameter unterstützen wird dieser Wert beim Nichtangeben auf `VARIANT_TRUE` gesetzt.

**Dokumentation der Properties:**

\$ `long ActualSize [get]`

`ActualSize` liefert die Größe des Datenstreams in Byte zurück.

Beim ersten Zugriff auf eine der Streamfunktionalitäten der Schnittstelle wird die zugrundeliegende Datei geöffnet und offen gehalten. Wird von einem anderen Aufrufer anschließend ein exklusiver Zugriff auf diese Datei benötigt, kann sie über den Aufruf der Methode `UnloadStream` wieder freigegeben werden.

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) enthält die Größe des Datenstreams.

\$ `VARIANT_BOOL AutoDelete [get, set]`

`AutoDelete` liefert, ob die Datei von diesem Objekt nach dem Gebrauch automatisch gelöscht werden soll und setzt, ob die Datei von diesem Objekt nach dem Gebrauch automatisch gelöscht werden soll.

Die Kontrolle über das automatische Löschen der Datei erfolgt im zugrundeliegenden File-Stream. Es besteht daher die Möglichkeit, mit dem `Stream` auch nach der Zerstörung des `FileParameter`-Objektes weiterzuarbeiten. Wenn keine der Streamfunktionalitäten verwendet wurde, überprüft der `FileParameter` im Release, ob die Datei zu löschen ist.

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) soll die Datei automatisch gelöscht werden?

Die Kontrolle über das automatische Löschen der Datei erfolgt im zugrunde liegenden File-Stream. Es besteht daher die Möglichkeit, mit dem `Stream` auch nach der Zerstörung des `FileParameter`-Objektes weiterzuarbeiten. Wenn keine der Streamfunktionalitäten verwendet wurde, überprüft der `FileParameter` im Release, ob die Datei zu löschen ist.

**Parameter:**

[in]: `newVal` soll die Datei automatisch gelöscht werden?

\$ `BSTR FileName [get, set]`

`FileName` liefert den vollständigen Namen der Datei zurück und setzt einen neuen Dateinamen.

Diese Eigenschaft ist die Standardeingenschaft der Schnittstelle

**Parameter:**

[out]: pVal (VB-Rückgabeparameter) Name der Datei

Wird bereits mit einer Datei gearbeitet, werden deren Ressourcen freigegeben. Schlägt die Zuweisung eines neuen Dateinamens fehl (die Datei wird nicht gefunden) wird ein Fehler gemeldet und das Objekt behält die alten Einstellungen.

**Parameter:**

[in]: newVal neuer Dateiname

§ BSTR XML [get, set]

XML liefert den Inhalt des File-Parameters als XML-String und schreibt die übergebene XML-Zeichenkette, in die zugrunde liegende Datei.

Beim ersten Zugriff auf eine der Streamfunktionalitäten der Schnittstelle wird die zugrunde liegende Datei geöffnet und offen gehalten. Wird von einem anderen Aufrufer anschließend ein exklusiver Zugriff auf diese Datei benötigt, kann sie über den Aufruf der Methode `UnloadStream` wieder freigegeben werden.

Die ausgegebene XML-Zeichenkette wird über den XML-Parsers ausgelesen.

Lässt sich aus den Daten keine valide XML-Zeichenkette generieren, wird ein Fehler geworfen. Die Fehlermeldung richtet sich nach dem zur Validierung verwendeten XML-Parsers (MS-XML4).

Der Positionszeiger des `Streams` wird durch den Aufruf dieser Eigenschaft nicht verändert.

**Parameter:**

[out]: pVal (VB-Rückgabeparameter) enthält die als XML-Zeichenkette dekodierten Daten.

Die alten Daten der Datei werden durch den Aufruf dieser Eigenschaft überschrieben. Nach dem Aufruf dieser Eigenschaft weist der Positionszeiger des `Streams` auf den Anfang desselben.

Die Daten werden entsprechend der in der XML-Zeichenkette angegebenen Kodierung in die Datei geschrieben.

Die übergebenen Daten werden bei der Konvertierung validiert. Werden nicht-XML-konforme Daten übergeben, wird ein Fehler ausgelöst. Der `Stream` hat in diesem Fall die Länge 0. Die Fehlermeldung richtet sich nach dem zur Validierung verwendeten XML-Parser (MS-XML4).

**Parameter:**

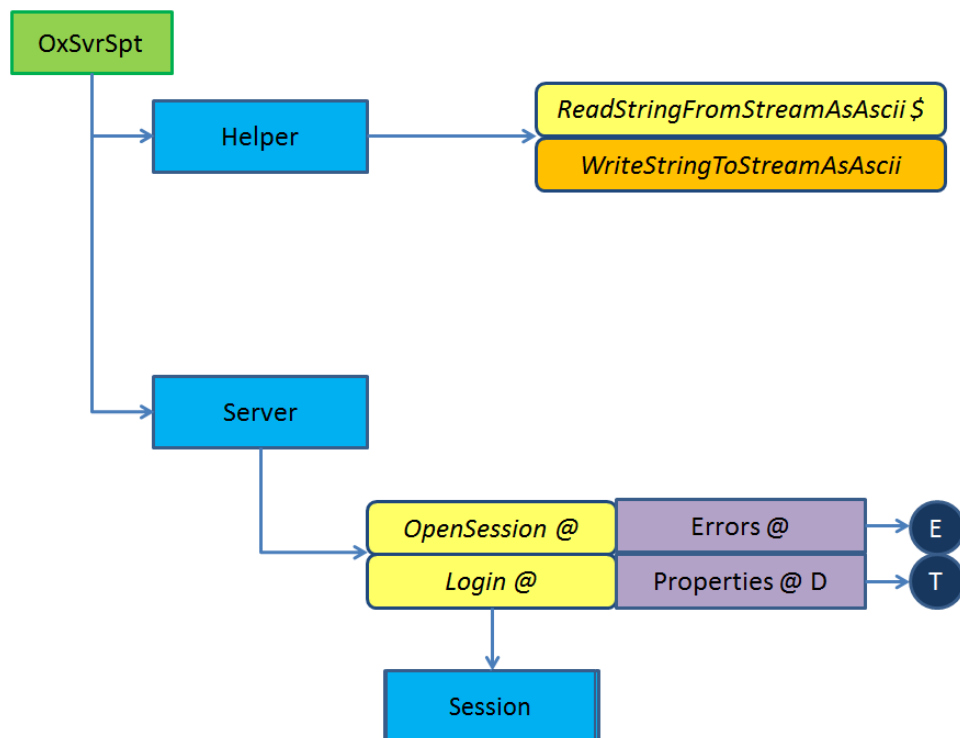
[in]: newVal Basicstring mit den XML-Daten.

## IHelper

**Beschreibung:**

`IHelper` stellt allgemeine Hilfsfunktionen für den Zugriff aus Scriptumgebungen zur Verfügung.

```
import "OxSvrSpt.idl"
```



### Öffentliche Methoden:

`HRESULT WriteStringToStreamAsAscii ([in] IStream * Stream, [in] BSTR Text)`

`HRESULT ReadStringFromStreamAsAscii ([in] IStream * Stream, [in] long Length, [out, retval] BSTR * pOut)`

`HRESULT CreateStream ([out, retval] IStream ** ppVal)`

### Dokumentation der Elementfunktionen:

§ `HRESULT CreateStream ([out, retval] IStream ** ppVal)`

`CreateStream` erzeugt ein neues `IStream`-Objekt und liefert dieses zurück.

§ `HRESULT ReadStringFromStreamAsAscii ([in] IStream * Stream, [in] long Length, [out, retval] BSTR * pOut)`

`ReadStringFromStreamAsAscii` liest Text als ASCII-Zeichen aus dem Stream.

Es wird die angegebene Menge an ASCII-Zeichen aus dem `Stream` gelesen und als Basic-Zeichenkette zurückgeliefert. Beinhaltet der `Stream` weniger als die angeforderte Menge an Zeichen, werden alle noch bis zum Streamende vorhandenen Zeichen ausgelesen und zurückgeliefert. Die Methode beginnt mit dem Lesen an der Stelle, an der sich der aktuelle Positionszeiger des Streams befindet. Nach dem Lesen verbleibt der Positionszeiger auf der Stelle, auf der er nach der Leseoperation stand.

### Parameter:

[in]: `Stream` `IStream`-Instanz, aus der gelesen werden soll.

[in]: `Length` Anzahl der Zeichen, die maximal gelesen werden sollen.



[out]: pOut (VB-Rückgabeparameter) gelesene und in einen BSTR konvertierte Zeichen aus dem Stream.

```
§ HRESULT WriteStringToStreamAsAscii ([in] IStream * Stream, [in] BSTR Text)
```

WriteStringToStreamAsAscii schreibt den übergebenen Text in den angegebenen Stream.

Der Text wird als BSTR erwartet und in ASCII-konvertierter Form in den Stream geschrieben. Vor dem Schreiben wird der Positionszeiger des Streams auf das Ende gesetzt.

#### Parameter:

[in]: Stream Stream, in den die Daten geschrieben werden sollen.

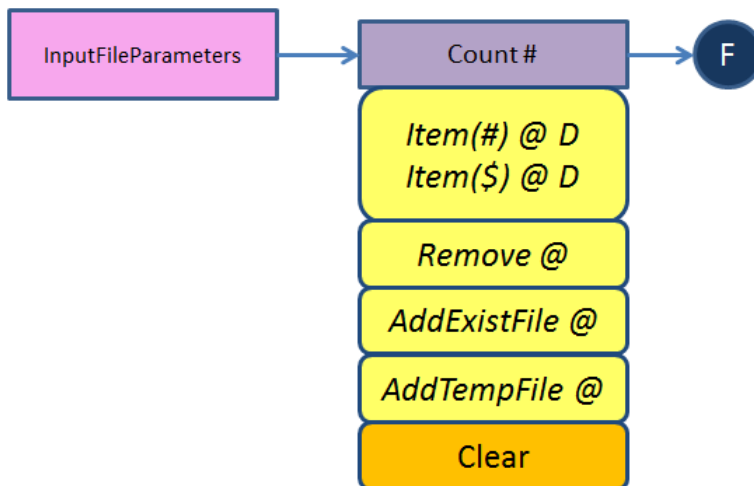
[in]: Text der Text als Basic-Zeichenkette, der in den Stream übernommen werden soll.

## InputFileParameters

### Beschreibung:

IInputFileParameters ist eine Collection für die Verwaltung der Input-Dateien eines Jobs.

```
import "OxSvrSpt.idl"
```



### Öffentliche Methoden:

```
HRESULT Delete ([in] VARIANT Index)
```

```
HRESULT Clear ()
```

```
HRESULT AddExistFile ([in] BSTR FileName, [in, defaultValue(0)]  
VARIANT_BOOL AutoDelete, [out, retval] IFileParameter ** ppNewParameter)
```

```
HRESULT AddTempFile ([in, defaultValue(0xffff)] VARIANT_BOOL AutoDelete,  
[in, defaultValue("tmp")] BSTR FileExtension, [out, retval]  
IFileParameter ** ppNewParameter)
```

### Properties:

```
long Count [get]
```

```
IFileParameter Item([in] VARIANT Index) [get]
```

### Dokumentation der Elementfunktionen:

```
$ HRESULT AddExistFile ([in] BSTR FileName, [in, defaultvalue(0)]  
    VARIANT_BOOL AutoDelete, [out, retval] IFileParameter **  
    ppNewParameter)
```

AddExistFile fügt eine existierende Datei als neuen Fileparameter hinzu und liefert dieses zurück

Die übergebene Datei wird standarmäßig nicht gelöscht.

Es wird geprüft, ob die Datei existiert. Sie wird jedoch erst geöffnet, wenn über die betreffenden Methoden der IFileParameter-Schnittstelle auf deren Daten zugegriffen wird.

#### Parameter:

[in]: FileName Name der Datei.

[in]: AutoDelete (Standardwert: VARIANT\_FALSE) gibt an, ob die Datei nach der Verwendung automatisch durch diese Klasse gelöscht werden soll.

[out]: ppNewParameter (VB-Rückgabewert) zugehöriges FileParameter-Objekt.

```
$ HRESULT AddTempFile ([in, defaultvalue(0xffff)] VARIANT_BOOL  
    AutoDelete, [in, defaultvalue("tmp")] BSTR FileExtension, [out,  
    retval] IFileParameter ** ppNewParameter)
```

AddTempFile erzeugt einen File-Parameter und erstellt automatisch eine temporäre Datei für die Datenübergabe zum Server

Diese Datei wird standarmäßig nach der Verwendung gelöscht.

Im Bedarfsfall kann der Name der Datei kann über die Eigenschaft FileName des zurückgelieferten IFileParameter-Objektes ermittelt werden.

#### Parameter:

[in]: AutoDelete (Standardwert: VARIANT\_FALSE) gibt an, ob die Datei nach der Verwendung automatisch durch diese Klasse gelöscht werden soll.

[in]: FileExtension (Standardwert: tmp) gibt für die temporäre Datei eine Dateiendung vor.

[out]: ppNewParameter (VB-Rückgabewert) zugehöriges FileParameter-Objekt.

```
$ HRESULT Clear ()
```

Clear entfernt alle Elemente der Collection.

```
$ HRESULT Delete ([in] VARIANT Index)
```

Delete löscht den Parameter anhand der Position in der Lister oder dem Namen.

#### Parameter:

[in]: Index Name oder Position des zu löschenden Lizenzeintrags.

### Dokumentation der Properties:

```
$ long Count [get]
```

Count liefert die Anzahl der Elemente in der Collection zurück.

#### Parameter:

[out]: plNumber (VB-Rückgabewert) Anzahl der Elemente der Collection.

\$ IFileParameter Item([in] VARIANT Index) [get]

Item liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert errCollectionIndexOutOfRange zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert errCollectionItemNotFound zurückgeliefert.

#### Parameter:

[in]: Index Position bzw. Name des angeforderten Elements.

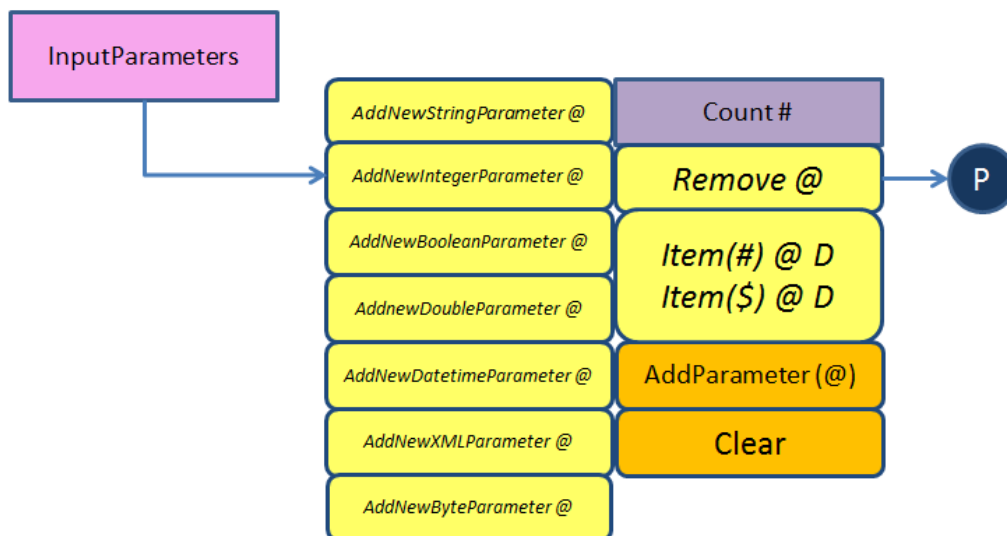
[out]: ppItem (VB-Rückgabewert) zugehöriges FileParameter-Objekt.

## InputParameters

### Beschreibung:

IInputParameters ist eine Collection für die Erstellung und Verwaltung der InputParameter eines Jobs.

```
import "OxSvrSpt.idl"
```



### Öffentliche Methoden:

```
HRESULT AddNewStringParameter ([in] BSTR Name, [in] BSTR Value, [out,
retval] IParameter ** ppVal)
```

```
HRESULT AddNewIntegerParameter ([in] BSTR Name, [in] long Value, [out,
retval] IParameter ** ppVal)
```

```
HRESULT AddNewBooleanParameter ([in] BSTR Name, [in] VARIANT_BOOL Value,
[out, retval] IParameter ** ppVal)
```

```
HRESULT AddNewDoubleParameter ([in] BSTR Name, [in] double Value, [out,
retval] IParameter ** ppVal)
```

```

HRESULT AddNewDatetimeParameter ([in] BSTR Name, [in] DATE Value, [out,
retval] IParameter ** ppVal)

HRESULT AddNewXMLParameter ([in] BSTR Name, [in, defaultvalue("")] BSTR
XML, [out, retval] IParameter ** ppVal)

HRESULT AddNewByteParameter ([in] BSTR Name, [in, optional] VARIANT
Value, [out, retval] IParameter ** ppVal)

HRESULT AddParameter ([in] IParameter * Parameter)

HRESULT Remove ([in] VARIANT Index, [out, retval] IParameter ** ppVal)

HRESULT Clear ()

```

### Properties:

```

long Count [get]

IParameter Item([in] VARIANT Index) [get]

```

### Dokumentation der Elementfunktionen:

```

§ HRESULT AddNewBooleanParameter ([in] BSTR Name, [in] VARIANT_BOOL
Value, [out, retval] IParameter ** ppVal)

```

AddNewBooleanParameter erzeugt einen neuen Boolean-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück. Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst.

#### Parameter:

[in]: Name      Name des Parameters.

[in]: Value    Initialisierungswert des Parameters.

[out]: ppVal      (VB-Rückgabeparameter) erzeugtes und initialisiertes Parameterobjekt.

#### Ausnahmebehandlung:

errInputParametersCantCreate (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

errParameterNameEmpty (1201) Es wurde kein Name für den Parameter angegeben.

errParameterDoubleName (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

```

§ HRESULT AddNewByteParameter ([in] BSTR Name, [in, optional] VARIANT
Value, [out, retval] IParameter ** ppVal)

```

AddNewByteParameter erzeugt einen neuen Byte-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück. Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst.

Der Wert kann bei der Initialisierung als Zeichenkette übergeben werden bzw. nachträglich über die AppendChunk bzw. die Stream-Funktionen hinzugefügt werden. Wenn der Wert für Value später initialisiert werden soll, muss ein Variant vom Typ VT\_NULL oder VT\_ERROR übergeben werden.

Unter VB und VB-Script braucht der `Value`-Parameter der Methode nicht angegeben zu werden, wenn die Binär-Daten zu einem späteren Zeitpunkt dem Parameterobjekt hinzugefügt werden sollen.

**Parameter:**

[in]: `Name` Name des Parameters .

[in]: `Value` Initialisierungswert des Parameters.

[out]: `ppVal` (VB-Rückgabeparameter) erzeugt und initialisiertes Parameterobjekt.

**Ausnahmebehandlung:**

`errInputParametersCantCreate` (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

`errParameterNameEmpty` (1201) Es wurde kein Name für den Parameter angegeben.

`errParameterDoubleName` (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

**Beispiel:**

```
Dim oParameter As OXSvrSpt.Parameter
Set oParameter = m_oInputParameters.AddNewByteParameter(strName)
```

```
§ HRESULT AddNewDatetimeParameter ([in] BSTR Name, [in] DATE Value,
[out, retval] IParameter ** ppVal)
```

`AddNewDatetimeParameter` erzeugt einen neuen Double-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück. Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst. VT\_8

**Parameter:**

[in]: `Name` Name des Parameters .

[in]: `Value` Initialisierungswert des Parameters.

[out]: `ppVal` (VB-Rückgabeparameter) erzeugtes und initialisiertes Parameterobjekt.

**Ausnahmebehandlung:**

`errInputParametersCantCreate` (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

`errParameterNameEmpty` (1201) Es wurde kein Name für den Parameter angegeben.

`errParameterDoubleName` (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

```
§ HRESULT AddNewDoubleParameter ([in] BSTR Name, [in] double Value,
[out, retval] IParameter ** ppVal)
```

`AddNewDoubleParameter` erzeugt einen neuen Double-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück. Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst. VT\_8

**Parameter:**

[in]: `Name` Name des Parameters .

[in]: Value Initialisierungswert des Parameters.

[out]: ppVal (VB-Rückgabeparameter) erzeugt und initialisiertes Parameterobjekt.

#### **Ausnahmebehandlung:**

errInputParametersCantCreate (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

errParameterNameEmpty (1201) Es wurde kein Name für den Parameter angegeben.

errParameterDoubleName (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

```
§ HRESULT AddNewIntegerParameter ([in] BSTR Name, [in] long Value, [out, retval] IParameter ** ppVal)
```

AddNewIntegerParameter erzeugt einen neuen Integer-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück.

Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst. VT\_I4

#### **Parameter:**

[in]: Name Name des Parameters.

[in]: Value Initialisierungswert des Parameters.

[out]: ppVal (VB-Rückgabeparameter) erzeugt und initialisiertes Parameterobjekt.

#### **Ausnahmebehandlung:**

errInputParametersCantCreate (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

errParameterNameEmpty (1201) Es wurde kein Name für den Parameter angegeben.

errParameterDoubleName (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

```
§ HRESULT AddNewStringParameter ([in] BSTR Name, [in] BSTR Value, [out, retval] IParameter ** ppVal)
```

AddNewStringParameter erzeugt einen neuen String-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück.

Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst.

#### **Parameter:**

[in]: Name Name des Parameters.

[in]: Value Initialisierungswert des Parameters.

[out]: ppVal (VB-Rückgabeparameter) erzeugt und initialisiertes Parameterobjekt.

#### **Ausnahmebehandlung:**

errInputParametersCantCreate (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

`errParameterNameEmpty` (1201) Es wurde kein Name für den Parameter angegeben.

`errParameterDoubleName` (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

### Beispiel:

C++

```
try
{
    IInputParametersPtr spParameters( spJob->InputParameters );
    _bstr_t bstrParameterName( L"Name" );
    _bstr_t bstrParameterValue( L"Wert" );
    spParameters->AddNewStringParameter( bstrParameterName, bstrParameterValue );
}
catch( _com_error& e )
{
    // Fehlerbeschreibung ermitteln
    _bstr_t bstrError = e.Description( );
    // wenn keine Fehlerbeschreibung vom COM-Fehlerobjekt mitgeliefert wurde ...
    if( bstrError.length() == 0 )
    {
        // ... die Fehlermeldung des Systems ermitteln
        bstrError = e.ErrorMessage();
    }
    // TODO bstrError enthält die Fehlerbeschreibung zur
    // weiteren Verarbeitung.
}
```

```
§ HRESULT AddNewXMLParameter ([in] BSTR Name, [in, defaultvalue("")]
    BSTR XML, [out, retval] IParameter ** ppVal)
```

`AddNewXMLParameter` erzeugt einen neuen XML-Parameter, fügt diesen zur Collection hinzu und liefert ihn zurück. Der Name des Parameters darf keine leere Zeichenkette sein, ansonsten wird ein Fehler zurückgeliefert. Wird versucht, einen Parameter mit einem Namen hinzuzufügen, der bereits existiert, wird ein Fehler ausgelöst.

Der Wert kann bei der Initialisierung als Zeichenkette übergeben werden bzw. nachträglich über die `AppendChunk` bzw. die `Stream`-Funktionen hinzugefügt werden.

Unter VB und VB-Script braucht der XML-Parameter der Methode nicht angegeben zu werden, wenn die XML-Daten zu einem späteren Zeitpunkt dem Parameterobjekt hinzugefügt werden sollen.

### Parameter:

[in]: Name      Name des Parameters.

[in]: XML      Initialisierungswert des Parameters.

[out]: ppVal      (VB-Rückgabeparameter) erzeugt und initialisiertes Parameterobjekt.

### Ausnahmebehandlung:

`errInputParametersCantCreate` (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

`errParameterNameEmpty` (1201) Es wurde kein Name für den Parameter angegeben.

`errParameterDoubleName` (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

### Beispiel:

```
Dim oParameter As OxSvrSpt.Parameter
Set oParameter = m_oInputParameters.AddNewXMLParameter(strName)
```

```
$ HRESULT AddParameter ([in] IParameter * Parameter)
```

AddParameter fügt den übergebenen Parameter in die Collection ein.

Es ist durch diese Methode möglich Parameter von anderen Aufrufen ohne Kopie weiterzureichen.

**Parameter:**

[in]: Parameter      hinzuzufügender Parameter.

**Ausnahmebehandlung:**

errInputParametersCantCreate (1101) Der Parameter konnte nicht erzeugt werden. Es stehen in diesem Fall keine weiteren Informationen zur Verfügung.

errParameterNameEmpty (1201) Es wurde kein Name für den Parameter angegeben.

errParameterDoubleName (1202) Es existiert bereits ein Parameter mit dem angegebenen Namen.

**Beispiel:**

```
Dim oParameter As OxSvrSpt.Parameter
...
' Parameter in anderem Job (-aufruf) füllen
...
m_oInputParameters.AddParameter(oParameter)
```

```
$ HRESULT Clear ()
```

Clear entfernt alle Elemente der Collection.

```
$ HRESULT Remove ([in] VARIANT Index, [out, retval] IParameter ** ppVal)
```

Remove entfernt den Eintrag mit dem übergebenen Bezeichner aus der Parameterliste und liefert diesen zurück.

Wird der angegebene Eintrag nicht gefunden, wird ein Fehler zurückgeliefert.

**Parameter:**

[in]: Index      Name oder Position des Eintrags, der aus der Parametercollection entfernt werden soll. Wird der Index-Parameter als Integer oder Long übergeben, wird nach dem Item an der entsprechenden Position gesucht. In C++ ist in diesem Fall ein Variant vom Typ VT\_12 oder VT\_14 zu übergeben. Wird der Index-Parameter als Zeichenkette (VT\_BSTR) übergeben, wird nach dem Namen des Items gesucht.

[out]: ppVal      (VB-Rückgabeparameter) liefert das entfernte Parameterobjekt zurück.

**Ausnahmebehandlung:**

errCollectionItemNotFound (1304) Der angeforderte Eintrag wurde nicht gefunden.

errCollectionIndexOutOfRange (1303) Der angeforderte Index befindet sich außerhalb des Bereichs.

**Dokumentation der Properties:**

```
$ long Count [get]
```

Count liefert die Anzahl der Elemente in der Collection zurück.

**Parameter:**



[out]: plNumber (VB-Rückgabewert) Anzahl der Elemente der Collection

\$ IParameter Item([in] VARIANT Index) [get]

Item liefert liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert `errCollectionIndexOutOfRange` zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert `errCollectionItemNotFound` zurückgeliefert.

#### Parameter:

[in]: Index Position bzw. Name des angeforderten Elements

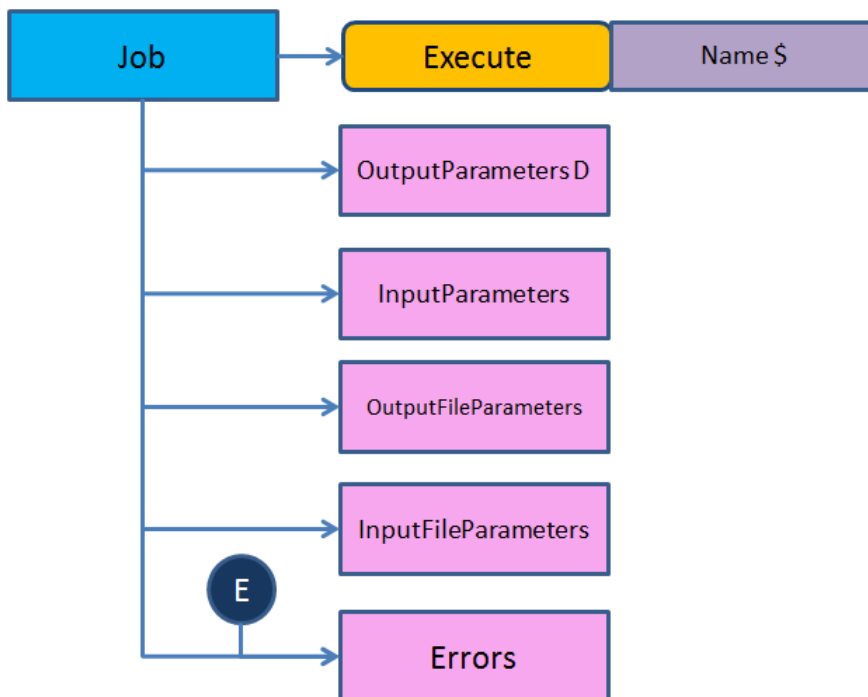
[out]: ppItem zugehöriges Parameter-Objekt

## IJob

### Beschreibung:

IJob kapselt den Zugriff auf die Parameter und den Aufruf eines Jobs.

import "OxSvrSpt.idl"



### Öffentliche Methoden:

HRESULT Execute ()

### Properties:

IOutputParameters OutputParameters [get]

BSTR Name [get]

IInputParameters InputParameters [get]

IErrors Erros [get]

```
IInputFileParameters InputFileParameters [get]
IOutputFileParameters OutputFileParameters [get]
```

### Dokumentation der Elementfunktionen:

§ HRESULT Execute ()

Nach dem Aufruf dieser Methode stehen die Ausgabeparameter in den Collections

OutputParameters und OutputFileParameters bereit. Tritt bei der Ausführung des Jobs ein Fehler auf, wird ein COM-Fehler geworfen. Dies geschieht sowohl bei den logischen, als auch bei den Systemfehlern. Der COM-Fehler enthält die oberste Meldung der Error-Collection. Die weiteren Meldungen können der Error-Collection über die Eigenschaft Errors dieses Objekts entnommen werden.

### Beispiele:

VB

```
' einloggen
Dim oServer As New OxSvrSpt.Server
Dim oSession As OxSvrSpt.Session
Set oSession = oServer.Login("root", "optimal", "localhost", "4000",
pwNotEncrypted)

' Job erstellen
Dim oJob As OxSvrSpt.Job
Set oJob = oSession.NewJob("dms.GetResultList")

' Parameter hinzufügen
Dim oParameter As OxSvrSpt.Parameter
oJob.InputParameters.AddNewIntegerParameter "Flags", 16
Set oParameter = oJob.InputParameters.AddNewXMLParameter("XML")
Dim oDomDocument As New MSXML2.DOMDocument40
Dim bSuccess As Boolean
bSuccess = oDomDocument.Load("c:\dmstest.xml")
oDomDocument.save oParameter.Stream

' Job ausführen
oJob.Execute

' XML aus dem XML-Fileparameter auslesen
Dim strXML As String
strXML = oJob.OutputFileParameters(1).XML
```

C++ über den Import-Mechanismus

```
#import "../Debug/OxSvrSpt.dll" raw_method_prefix("raw_")
using namespace OxSvrSpt;
.
.
.
try
{
    _bstr_t bstrUser          = L"root";
    _bstr_t bstrPassword      = L"optimal";
    _bstr_t bstrServer        = L"adunkel";
    _bstr_t bstrPort          = L"4000";
    _bstr_t bstrJobName       = L"krn.GetServerInfo";
    OxSvrSpt::IServerPtr spServer( __uuidof( OxSvrSpt::Server ));
    OxSvrSpt::ISessionPtr spSession = spServer->Login( bstrUser, bstrPassword,
    bstrServer, bstrPort, pwNotEncrypted );
    OxSvrSpt::IJobPtr spJob = spSession->NewJob( bstrJobName );
    spJob->InputParameters->AddNewIntegerParameter( L"Flags", 0 );
    spJob->InputParameters->AddNewIntegerParameter( L"Info", 1 );
    spJob->Execute();
    _variant_t varName = spJob->OutputParameters->GetItem( L"Name" )->Value;
}
catch( _com_error& e )
{
    _bstr_t bstrError = e.Description( );
    if( bstrError.length() == 0 )
    {
        bstrError = e.ErrorMessage();
    }
    AfxMessageBox( bstrError );
}
```

### Dokumentation der Properties:

§ `Errors` `Errors` [get]

`Errors` liefert die Collection mit den beim Server aufgetretenen Fehler zurück

Diese enthält Objekte vom Typ `IError`.

#### Parameter:

[out]: `pVal` (VB-Rückgabewert) Collection mit den aufgetretenen Fehlern

§ `InputFileParameters` `InputFileParameters` [get]

`InputFileParameters` liefert die Collection mit den `FileParameters` für die Übergabe an den Server

Diese enthält Objekte vom Typ `IFileParameters`

#### Parameter:

[out]: `pVal` (VB-Rückgabewert) Collection mit den Dateien für den Serveraufruf

§ `InputParameters` `InputParameters` [get]

`InputFileParameters` liefert die Collection der `InputParameters` zurück.

#### Parameter:

[out]: `pVal` (VB-Rückgabewert) Collection der Input-Parameter

§ `Name` `Name` [get]

`Name` liefert den Namen des `Jobs`.

Der Name des `Jobs` wird bei der Erstellung des `Job`-Objektes angegeben und kann anschließend nicht mehr verändert werden.

### Parameter:

[out]: pVal (VB-Rückgabewert) Name des Jobs

\$ IOutputFileParameters OutputFileParameters [get]

OutputFileParameters liefert die Collection mit den FileParametern die vom Server nach dem Aufruf eines Jobs bereitgestellt werden

Diese enthält Objekte vom Typ IFileParametern

### Parameter:

[out]: pVal (VB-Rückgabewert) Collection mit den Ergebnisdateien nach einem Serveraufruf.

\$ IOutputParameters OutputParameters [get]

OutputParameters liefert die Collection der Outputparameter zurück

Diese Eigenschaft ist die Standardeigenschaft der IJob-Schnittstelle.

### Parameter:

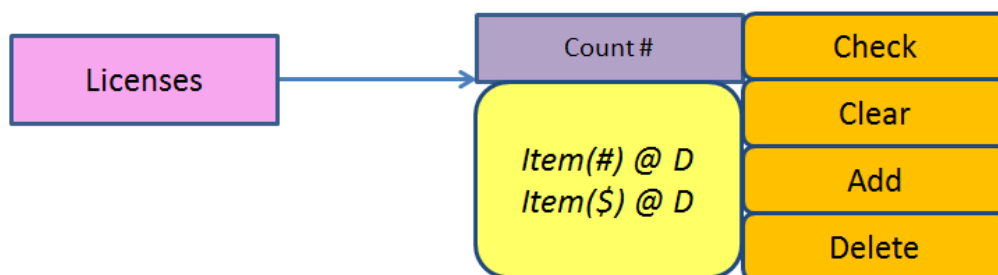
[out]: pVal (VB-Rückgabewert) Collection der Ausgabeparameter des Jobs

## ILicenses

### Beschreibung:

ILicenses ist eine Collection zur Verwaltung der Lizenzen.

import "OxSvrSpt.idl"



### Öffentliche Methoden:

HRESULT Add ([in] BSTR Name)

HRESULT Delete ([in] VARIANT Index)

HRESULT Clear ()

HRESULT Check ([in] BSTR Name)

### Properties:

long Count [get]

BSTR Item([in] VARIANT Index) [get]

### Dokumentation der Elementfunktionen:

\$ HRESULT Add ([in] BSTR Name)

Add fügt eine neue Lizenz-Zeichenkette hinzu.

Wird versucht, eine Lizenzs-Zeichenkette doppelt hinzuzufügen, wird der zweite ignoriert. Es wird kein Fehler geworden. Die übergebene Lizenz-Zeichenkette wird in Großbuchstaben umgewandelt und intern verwaltet.

**Parameter:**

[in]: Name      hinzuzufügende Lizenz-Zeichenkette

§ HRESULT Check ([in] BSTR Name)

Check überprüft die angegebene Lizenz beim Server, ohne sie mittels LicLogin einzuchecken.

Diese Methode arbeitet über den Serverjob "lic.CheckLicense"

Schlägt die Überprüfung fehl, wird der oberste Fehler der Fehler-Collection als COM-Fehler zurückgeliefert.

**Parameter:**

[in]: Name      die zu überprüfende Lizenz

§ HRESULT Clear ()

Clear entfernt alle Elemente der Collection.

§ HRESULT Delete ([in] VARIANT Index)

Delete löscht den Lizenzeintrag anhand der Position in der Liste oder dem Namen.

**Parameter:**

[in]: Index      Name oder Position des zu löschenden Lizenzeintrags

**Dokumentation der Properties:**

§ long Count [get]

Count liefert die Anzahl der Elemente in der Collection zurück.

**Parameter:**

[out]: plNumber      (VB-Rückgabewert) Anzahl der Elemente der Collection

§ BSTR Item([in] VARIANT Index) [get]

Item liefert liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert `errCollectionIndexOutOfRange` zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert `errCollectionItemNotFound` zurückgeliefert.

**Parameter:**

[in]: Index      Position bzw. Name des angeforderten Elements

[out]: pItem      (VB-Rückgabewert) zugehörige Lizenz, die dem Schlüssel entspricht

## ILogger

**Beschreibung:**

ILogger ist eine Schnittstelle für den Zugriff auf den OS-Logger.

```
import "OxSvrSpt.idl"
```

**Öffentliche Methoden:**

```

HRESULT Log ([in] BSTR Message, [in] LogLevelEnum Level, [in,
defaultvalue("")] BSTR FileName, [in, defaultvalue("")] BSTR Function,
[in, defaultvalue(0)] long Line)

HRESULT Error ([in] BSTR Message, [in, defaultvalue("")] BSTR Function,
[in, defaultvalue(0)] long Line)

HRESULT Info ([in] BSTR Message, [in, defaultvalue("")] BSTR Function,
[in, defaultvalue(0)] long Line)

HRESULT MethodEntry ([in] BSTR Message, [in, defaultvalue("")] BSTR
Function, [in, defaultvalue(0)] long Line)

HRESULT Debug ([in] BSTR Message, [in, defaultvalue("")] BSTR Function,
[in, defaultvalue(0)] long Line)

HRESULT Trace ([in] BSTR Message, [in, defaultvalue("")] BSTR Function,
[in, defaultvalue(0)] long Line)

HRESULT Init ([in] BSTR FileName, [in, defaultvalue("")] BSTR Alias)

```

**Dokumentation der Elementfunktionen:**

```

§ HRESULT Debug ([in] BSTR Message, [in, defaultvalue("")] BSTR
Function, [in, defaultvalue(0)] long Line)

```

Debug loggt die übergebene Nachricht im Debug-Level.

**Parameter:**

[in]: Message zu übergebende Nachricht

[in]: FunctionName der Funktion, in der der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: Line zu loggende Quelltextzeile. Wird diese nicht angegeben, erscheint 0 im Logprotokoll.

```

§ HRESULT Error ([in] BSTR Message, [in, defaultvalue("")] BSTR
Function, [in, defaultvalue(0)] long Line)

```

Error loggt die übergebene Nachricht im Error-Level.

**Parameter:**

[in]: Message zu übergebende Nachricht

[in]: FunctionName der Funktion, in der der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: Line zu loggende Quelltextzeile. Wird diese nicht angegeben, erscheint 0 im Logprotokoll.

```

§ HRESULT Info ([in] BSTR Message, [in, defaultvalue("")] BSTR Function,
[in, defaultvalue(0)] long Line)

```

Info loggt die übergebene Nachricht im Info-Level.

**Parameter:**

[in]: Message zu übergebende Nachricht

[in]: `FunctionName` der Funktion, in der der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: `Line` zu loggende Quelltextzeile. Wird diese nicht angegeben, erscheint 0 im Logprotokoll.

```
$ HRESULT Init ([in] BSTR FileName, [in, defaultvalue("")] BSTR Alias)
    Init initialisiert die Loggerinstanz.
```

**Parameter:**

[in]: `FileName` Dateiname für den alle Logeinträge vorgenommen werden sollen

[in]: `Alias` Alias der aktuellen Bibliothek, für die geloggt werden soll

```
$ HRESULT Log ([in] BSTR Message, [in] LogLevelEnum Level, [in,
    defaultvalue("")] BSTR FileName, [in, defaultvalue("")] BSTR Function,
    [in, defaultvalue(0)] long Line)
    Log loggt die übergebene Nachricht im angegebenen Level.
```

**Parameter:**

[in]: `Message` zu übergebende Nachricht

[in]: `Level` anzuwendendes LogLevel

[in]: `FileName` Name der Quelldatei, von der aus der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: `FunctionName` der Funktion, in die der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: `Line` zu loggende Quelltextzeile. Wird diese nicht angegeben, erscheint eine 0 im Logprotokoll.

```
$ HRESULT MethodEntry ([in] BSTR Message, [in, defaultvalue("")] BSTR
    Function, [in, defaultvalue(0)] long Line)
    MethodEntry loggt den Methodeneintritt.
```

**Parameter:**

[in]: `Message` zu übergebende Nachricht

[in]: `FunctionName` der Funktion, in die der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: `Line` zu loggende Quelltextzeile. Wird diese nicht angegeben, erscheint eine 0 im Logprotokoll.

```
$ HRESULT Trace ([in] BSTR Message, [in, defaultvalue("")] BSTR
    Function, [in, defaultvalue(0)] long Line)
    Trace loggt die übergebene Nachricht im Trace-Level.
```

**Parameter:**

[in]: `Message` zu übergebende Nachricht

[in]: `FunctionName` der Funktion, in die der Logeintrag erfolgt ist. Wird diese nicht angegeben, erscheint eine leere Zeichenkette im Logprotokoll.

[in]: Line zu loggende Quelltextzeile. Wird diese nicht angegeben, erscheint eine 0 im Logprotokoll.

## INotifyErrors

### Beschreibung:

INotifyErrors ist eine Schnittstelle für die Verwaltung der Fehler einer Notification. Diese Schnittstelle erweitert die IErrors-Schnittstelle um die Möglichkeit Fehlermeldungen zur Collection hinzuzufügen und zu entfernen.

```
import "OxSvrSpt.idl"
```

### Öffentliche Methoden:

```
HRESULT Add ([in] BSTR SourceName, [in] long ISourceCode, [in] BSTR
FaultString, [in] long FaultCode, [in] BSTR InfoList)
```

```
HRESULT Clear ()
```

```
HRESULT NewResultCode ([in] long newValue)
```

```
HRESULT NewResponseResult ([in] long newValue)
```

### Dokumentation der Elementfunktionen:

```
$ HRESULT Add ([in] BSTR SourceName, [in] long ISourceCode, [in] BSTR
FaultString, [in] long FaultCode, [in] BSTR InfoList)
```

Add fügt eine neue Fehlermeldung zur Collection hinzu.

#### Parameter:

[in]: SourceName Namen der Quelle, in der der Fehler aufgetreten ist

[in]: ISourceCode Quellzeile, in der der Fehler aufgetreten ist.

[in]: FaultString Fehlerbeschreibung

[in]: FaultCode Fehlercode

[in]: InfoList InfoList

```
$ HRESULT Clear ()
```

Clear entfernt alle Fehler-Objekte aus der Collection.

```
$ HRESULT NewResponseResult ([in] long newValue)
```

NewResponseResult setzt den Rückgabewert der Notification.

#### Parameter:

[out]: newVal Rückgabewert der Notification

```
$ HRESULT NewResultCode ([in] long newValue)
```

NewResultCode setzt den Fehlerwert der Notification.

#### Parameter:

[out]: newVal Fehlerwert des Servers



## INotifyInputFileParameters

### Beschreibung:

INotifyInputFileParameters beinhaltet die Eingangs-Fileparameter einer Notification.

```
import "OxSvrSpt.idl"
```

## INotifyInputParameters

### Beschreibung:

INotifyInputParameters ist eine Collection für die Verwaltung der Eingangsparameter einer Notification. Die Methoden entsprechen denen der Schnittstelle IOutputParameters.

```
import "OxSvrSpt.idl"
```

## INotifyJob

### Beschreibung:

INotifyJob beinhaltet die Daten beim Aufruf einer Notification.

```
import "OxSvrSpt.idl"
```

### Properties:

```
INotifyOutputParameters OutputParameters [get]
INotifyErrors Errors [get]
INotifyInputParameters InputParameters [get]
BSTR Name [get]
INotifyOutputFileParameters OutputFileParameters [get]
INotifyInputFileParameters InputFileParameters [get]
long UserData [get, set]
```

### Dokumentation der Properties:

\$ INotifyErrors Errors [get]

Errors liefert die Collection für die Aufnahme der Fehler.

#### Parameter:

[out]: pVal (VB-Rückgabeparameter) INotifyErrors-Collection

\$ INotifyInputFileParameters InputFileParameters [get]

InputFileParameters liefert die Collection der an die Notification übergebenen Fileparameter.

#### Parameter:

[out]: pVal (VB-Rückgabeparameter) INotifyInputFileParameters-Collection

\$ INotifyInputParameters InputParameters [get]

InputParameters liefert die Collection der an die Notification übergebenen Parameter.

#### Parameter:

[out]: pVal (VB-Rückgabeparameter) INotifyInputParameters-Collection

\$ BSTR Name [get]

Name liefert den Namen der Notification.

**Parameter:**

[out]: pVal (VB-Rückgabeparameter) Name der Notification

\$ INotifyOutputFileParameters OutputFileParameters [get]

OutputFileParameters liefert die Collection der Ausgabe-Fileparameter.

**Parameter:**

[out]: pVal (VB-Rückgabeparameter) INotifyOutputFileParameters-Collection

\$ INotifyOutputParameters OutputParameters [get]

OutputParameters liefert die Collection der Ausgabeparameter.

**Parameter:**

[out]: pVal (VB-Rückgabeparameter) INotifyOutputParameters-Collection

\$ long UserData [get, set]

UserData liefert und setzt die Benutzerdaten der Notification.

**Parameter:**

[out]: pVal (VB-Rückgabeparameter) Benutzerdaten der Notification

[in]: newVal Benutzerdaten der Notification

## INotifyOutputFileParameters

**Beschreibung:**

INotifyOutputFileParameters ist eine Collection für die Verwaltung der Ausgangsdateien einer Notification.

```
import "OxSvrSpt.idl"
```

## INotifyOutputParameters

**Beschreibung:**

INotifyOutputParameters ist eine Collection für die Erstellung und Verwaltung der OutputParameters einer Notification.

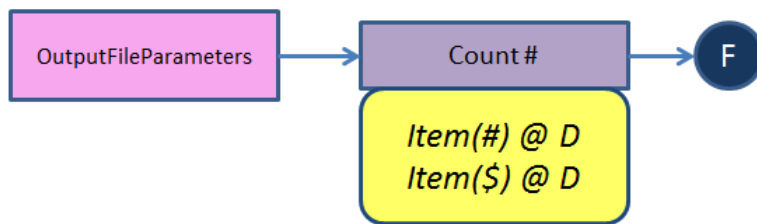
```
import "OxSvrSpt.idl"
```

## IOutputFileParameters

**Beschreibung:**

IOutputFileParameters beinhaltet die Fileparameter nach dem Aufruf eines Server-Jobs.

```
import "OxSvrSpt.idl"
```

**Properties:**

```
long Count [get]
```

```
IFileParameter Item([in] VARIANT Index) [get]
```

**Dokumentation der Properties:**

```
$ long Count [get]
```

Count liefert die Anzahl der Elemente in der Collection zurück.

**Parameter:**

[out]: `plNumber` (VB-Rückgabewert) Anzahl der Elemente der Collection.

```
$ IFileParameter Item([in] VARIANT Index) [get]
```

Item liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert `errCollectionIndexOutOfRange` zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert `errCollectionItemNotFound` zurückgeliefert.

**Parameter:**

[in]: `Index` Position bzw. Name des angeforderten Elements.

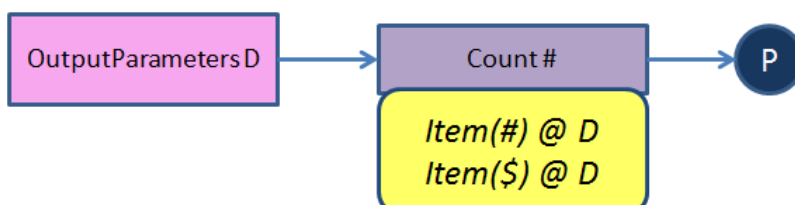
[out]: `ppItem` (VB-Rückgabewert) zugehöriges `FileParameter`-Objekt.

## IOutputParameters

**Beschreibung:**

`IOutputParameters` ist eine Collection für die Verwaltung der Outputparameter eines Jobs.

```
import "OxSvrSpt.idl"
```

**Properties:**

```
long Count [get]
```

```
IFileParameter Item([in] VARIANT Index) [get]
```

**Dokumentation der Properties:**

§ `long Count [get]`

`Count` liefert die Anzahl der Elemente in der Collection zurück.

**Parameter:**

[out]: `plNumber` (VB-Rückgabewert) Anzahl der Elemente der Collection.

§ `IFileParameter Item([in] VARIANT Index) [get]`

`Item` liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert `errCollectionIndexOutOfRange` zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert `errCollectionItemNotFound` zurückgeliefert.

**Parameter:**

[in]: `Index` Position bzw. Name des angeforderten Elements.

[out]: `ppItem` (VB-Rückgabewert) zugehöriges `FileParameter`-Objekt.

## IParameter

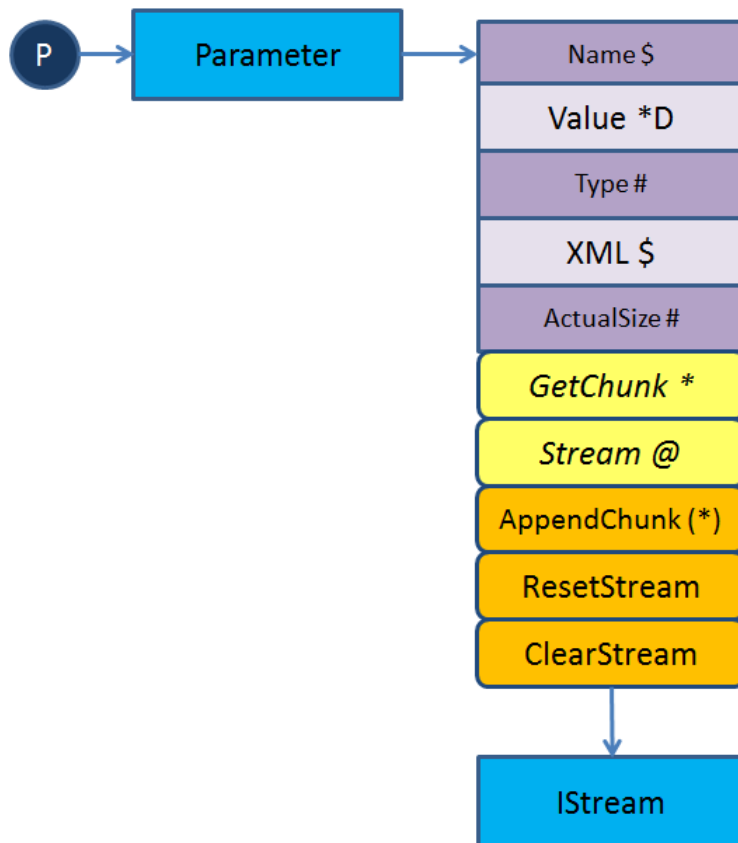
**Beschreibung:**

`IParameter` repräsentiert sowohl einen Eingangs- als auch einen Ausgangs-Parameter eines Jobs.

Die `IParameter`-Schnittstelle stellt Methoden für den Zugriff auf die Eigenschaften eines Parameters zur Verfügung. Diese sind in zwei Gruppen unterteilt. Die erste Gruppe steht bei allen Paramtertypen zur Verfügung. Die zweite nur bei Parametern vom Binär-Typ.

Zur ersten gehören `Name`, `Type` und `Value`. Diese repräsentieren den einfachsten Zugriff auf das Parameterobjekt. `Name` und `Type` stehen als Nur-Lese-Eigenschaften zur Verfügung. Die `Value`-Eigenschaft kann auch im Nachhinein gesetzt bzw. verändert werden. Je nach Parametertyp werden die an diese Eigenschaft übergebenen Werte unterschiedlich behandelt. Dabei erfolgt keine Überprüfung, ob der übergebene Wert dem geforderten Varianttyp entspricht. Anstelle dessen wird versucht, den übergebenen Wert in diesen Typ zu konvertieren. Dies geschieht über die COM-Konvertierungsfunktionen des Variants. Schlägt die Konvertierung fehl wird der betreffende COM-Fehler geworfen. Die verwendeten Zieltypen werden bei der Eigenschaft `Value` aufgeführt. Beim Setzen der `Value`-Eigenschaft

Zur zweiten Gruppe gehören die restlichen Methoden. Diese dienen der Verarbeitung der Binär-Daten des Parameters. Wird eine dieser Methoden bei einem Nicht-Binär-Parameter aufgerufen wird der Fehler `errParameterMethodUnsupported` geworfen.



### Öffentliche Methoden:

```

HRESULT Stream ([out, retval] IStream ** ppStream)
HRESULT AppendChunk ([in] VARIANT Data)
HRESULT GetChunk ([in] long Length, [out, retval] VARIANT * pResult)
HRESULT ResetStream ()
HRESULT ClearStream ()

```

### Properties:

```

VARIANT Value [get, set]
BSTR Name [get]
long ActualSize [get]
BSTR XML [get, set]
ParameterTypeEnum Type [get]

```

### Dokumentation der Elementfunktionen:

§ HRESULT AppendChunk ([in] VARIANT Data)

AppendChunk hängt zusätzliche Bytes an den Stream des Parameters

Diese Methode steht nur bei Parametern vom Typ `ptBinary` und `ptXML` zur Verfügung

Nach dem Aufruf dieser Methode steht der Positionszeiger des internen Streams auf dessen Ende.

**Parameter:**

[in]: Data enthält die Daten, die an den Stream angehängt werden sollen

Die Daten werden über die `OLE32 ChangeType`-Methode nach `VT_ARRAY|VT_UI1` konvertiert und weiterverarbeitet. Wird z. B. ein `BSTR` übergeben, werden die Daten als `WIDECHAR` weiterverarbeitet. Das Schreiben von 8bit-Zeichen in den Stream kann über die Methoden des `Helper-COM-Objektes` erfolgen.

**Beispiele:**

VB

```
Dim abWriteData() As Byte
ReDim abWriteData(0 To 5)
abWriteData(0) = 0
abWriteData(1) = 1
abWriteData(2) = 2
abWriteData(3) = 3
abWriteData(4) = 4
abWriteData(5) = 5
oParameter.AppendChunk abWriteData
```

§ HRESULT ClearStream ()

ClearStream löscht die Daten des Streams.

§ HRESULT GetChunk ([in] long Length, [out, retval] VARIANT \* pResult)

GetChunk liefert die angegebene Anzahl von Bytes aus dem Stream zurück.

Diese Methode steht nur bei Parametern vom Typ `ptBinary` und `ptXML` zur Verfügung.

Diese Methode beginnt ab der aktuellen Position im Stream die Daten zu lesen. Wird das Ende des Streams erreicht, bevor die gewünschte Anzahl an Zeichen gelesen wurde, werden nur die bis dahin gelesenen Zeichen zurückgeliefert. Die Anzahl der gelesenen Zeichen kann anhand der Größe des zurückgelieferten Puffers bestimmt werden (siehe Beispiel).

**Parameter:**

[in]: Length Anzahl der maximal zurückgeliefertten Bytes.

[out]: pResult (VB-Rückgabeparameter) enthält die gelesenen Bytes. Diese werden in einem Variant vom Typ `VT_ARRAY|VT_UI1` geliefert.

**Beispiel:**

VB

```
' der folgende Programm-Ausschnitt entspricht:
' Dim var
' var = oParameter.Value
' dort bleibt jedoch zusätzlich die aktuelle Position des Streams erhalten
' Stream zum Lesen auf die Anfangsposition setzen
oParameter.ResetStream
Dim abReadData() As Byte
...
' Array auf die benötigte Größe anpassen
ReDim abReadData(0 To oParameter.ActualSize - 1)
' Daten lesen
abReadData = oParameter.GetChunk(oParameter.ActualSize)
' Größe der gelesenen Daten anhand der zurückgegebenen Daten ermitteln
Dim nSize As Long
nSize = UBound(abReadData) - LBound(abReadData)

§ HRESULT ResetStream ()
```

ResetStream setzt den Datenstream auf den Anfang zurück.

Diese Methode steht nur bei Parametern vom Typ `ptBinary` und `ptXML` zur Verfügung.

```
$ HRESULT Stream ([out, retval] IStream ** ppStream)
```

`Stream` liefert den Stream der Binärdaten

Diese Eigenschaft steht nur bei Parametern vom Typ `ptBinary` und `ptXML` zur Verfügung

Der Stream enthält die Binärdaten des jeweiligen Parameters. Diese sind durch den Aufrufer nicht MIME-BASE-64 zu kodieren bzw. zu dekodieren. Dies geschieht automatisch durch die `OxSvrSpt-Bibliothek`.

**Parameter:**

[out]: `ppStream` (VB-Rückgabeparameter) enthält den Stream als `IStream`-Schnittstelle.

**Dokumentation der Properties:**

```
$ long ActualSize [get]
```

`ActualSize` liefert die Größe des Datenstreams in Byte zurück.

Diese Methode steht nur bei Parametern vom Typ `ptBinary` und `ptXML` zur Verfügung.

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) enthält die Größe des Datenstreams.

```
$ BSTR Name [get]
```

`Name` liefert den Namen des Parameters zurück.

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) Name der Datei

```
$ ParameterTypeEnum Type [get]
```

`Type` liefert den Typ des Parameters zurück.

Folgende Typen stehen zur Verfügung:

`ptString = 1`

`ptInteger = 2`

`ptBoolean = 3`

`ptDouble = 4`

`ptDateTime = 5`

`ptBinary = 6`

`ptXML = 6`

Die Parameter `ptBinary` und `ptXML` bilden dabei den Server-Parameter Base64 ab. Sie unterscheiden sich lediglich in den Erstellungsmöglichkeiten.

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) Typ des Parameters

```
$ VARIANT Value [get, set]
```

`Value` liefert den Wert des Parameters für nicht Base64-Parameter und setzt den Wert des Parameters.

Diese Eigenschaft steht nicht bei Parametern des Typs Base64 zur Verfügung. Anstatt dess kann über die `Stream`- und `Chunk`-Funktionen direkt auf die dekodierten Binärdaten zugegriffen werden. Weiterhin besteht die Möglichkeit über die XML-Eigenschaft

Bei Parametern des Typs Base64 besteht die Möglichkeit die Daten über die `Stream`- und `Chunk`-Funktionalität

Beim Setzen der Eigenschaft wird versucht, den übergebenen Wert des Variants in den benötigten Typ zu konvertieren. Als Zieltypen werden folgende verwendet:

```
ptString - VT_BSTR ptInteger - VT_I4 ptBoolean - VT_BOOL ptDouble - VT_R8
ptDateTime - VT_DATE ptBase64 - VT_ARRAY | VT_UI1
```

**Parameter:**

[in]: `newVal` neuer Wert des Parameters

Bei Parametern vom Typ Base64 werden die Binärdaten als `Array` im Rückgabe-Variant zurückgeliefert. Der Positionszeiger des `Streams` wird durch den Aufruf dieser Eigenschaft nicht verändert.

Der Typ des zurückgelieferten Variants ist abhängig vom Typ des Parameters.

folgende Typen werden verwendet:

```
ptString - VT_BSTR ptInteger - VT_I4 ptBoolean - VT_BOOL ptDouble - VT_R8
ptDateTime - VT_DATE ptBinary - VT_ARRAY | VT_UI1 ptXML - VT_ARRAY | VT_UI1
```

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) enthält den Wert des Parameters

§ BSTR XML [get, set]

XML liefert den Wert des Base64-Parameters als XML-String und setzt den Wert des Base64-Parameters aus der übergebenen XML-Zeichenkette.

Diese Methode steht nur bei Parametern vom Typ `ptBinary` und `ptXML` zur Verfügung.

Die ausgegebene XML-Zeichenkette wird über den XML-Parsers ausgelesen.

Lässt sich aus den Daten keine valide XML-Zeichenkette generieren, wird ein Fehler geworfen. Die Fehlermeldung richtet sich nach dem zur Validierung verwendeten XML-Parsers (MS-XML4).

Der Positionszeiger des `Streams` wird durch den Aufruf dieser Eigenschaft nicht verändert.

**Parameter:**

[out]: `pVal` (VB-Rückgabeparameter) enthält die als XML-Zeichenkette dekodierten Daten.

**Beispiel:**

VB



**Option explicit**

```

Dim oServer, oSession, oJob, oInputParameters, oByteParameter
Set oServer = CreateObject("OxsvrSpt.Server")
Set oSession = oServer.Login()
set oJob = oSession.NewJob("med.ObservationValues")
Set oInputParameters = oJob.InputParameters
set oByteParameter = oInputParameters.AddNewByteParameter("Parametername")
oByteParameter.XML = "<?xml version='1.0' encoding='utf-8'?>" + _
"<med>irgendein Test mit Umlauten äöüß</med>"

oByteParameter.ResetStream
' Daten über MSXML wieder auslesen
Dim oDocument, bSuccess
Set oDocument = CreateObject("MSXML.DOMDocument")
bSuccess = oDocument.Load(oByteParameter.Stream)
' Text des XML aus dem DOM auslesen und ausgeben
MsgBox oDocument.XML

```

Diese Eigenschaft steht nur bei Parametern des Typs Base64 zur Verfügung.

Dabei wird die übergebene XML-Zeichenkette in den im Parameterobjekt zur Verfügung gestellten Stream übertragen. Es wird automatisch die in der XML-Zeichenkette angegebene Kodierung berücksichtigt.

Die alten Daten des Streams werden durch den Aufruf dieser Eigenschaft überschrieben. Nach dem Aufruf dieser Eigenschaft weist der Positionszeiger des Streams auf den Anfang desselben.

Die übergebenen Daten werden bei der Konvertierung validiert. Werden nicht-XML-konforme Daten übergeben, wird ein Fehler ausgelöst. Der Stream hat in diesem Fall die Länge 0. Die Fehlermeldung richtet sich nach dem zur Validierung verwendeten XML-Parser (MS-XML4).

**Parameter:**

[in]: newVal Basicstring mit den XML-Daten.

**Beispiel:**

VB

```

Dim oParameter As OxSvrSpt.Parameter
...
oParameter.XML = "<?xml version='1.0' encoding='utf-8'?>" + _
"<med>irgendein Test mit Umlauten äöüß</med>"

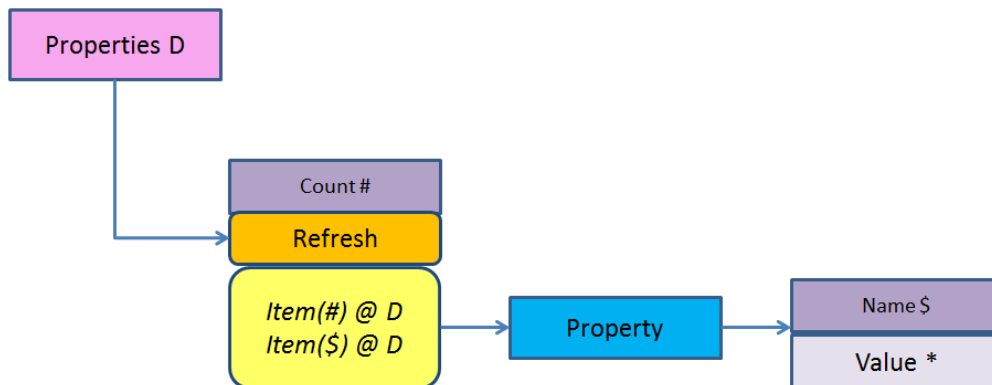
```

## IProperties

**Beschreibung:**

IProperties ist eine Collection für Eigenschaften.

```
import "OxSvrSpt.idl"
```



### Öffentliche Methoden:

HRESULT Refresh ()

### Properties:

long Count [get]

IProperty Item([in] VARIANT Index) [get]

### Dokumentation der Elementfunktionen:

\$ HRESULT Refresh ()

Refresh aktualisiert die Property-Collection. Dabei werden alle Einträge dieser Collection neu erzeugt.

### Dokumentation der Properties:

\$ long Count [get]

Count liefert die Anzahl der Elemente in der Collection zurück.

#### Parameter:

[out]: plNumber (VB-Rückgabewert) Anzahl der Elemente der Collection.

\$ IProperty Item([in] VARIANT Index) [get]

Item liefert das angegebene Element der Collection anhand der Position oder des Schlüssels zurück.

Wird eine Position außerhalb des gültigen Index angegeben, wird ein Fehler mit dem Fehlerwert `errCollectionIndexOutOfRange` zurückgegeben. Kann das Item nicht gefunden werden, wird ein Fehler mit dem Wert `errCollectionItemNotFound` zurückgeliefert.

#### Parameter:

[in]: Index Position bzw. Name des angeforderten Elements.

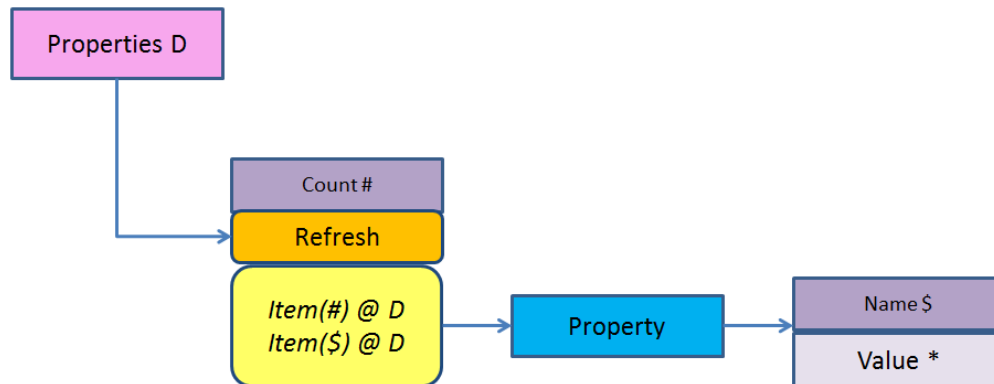
[out]: ppItem (VB-Rückgabewert) zugehöriges FileParameter-Objekt.

## IProperty

### Beschreibung:

IProperty repräsentiert eine Eigenschaft.

```
import "OxSvrSpt.idl"
```



### Properties:

```
VARIANT Value [get, set]
```

```
BSTR Name [get]
```

### Dokumentation der Properties:

```
$ BSTR Name [get]
```

Name liefert den Namen der Eigenschaft.

#### Parameter:

[out]: pVal (VB-Rückgabewert) Name der Eigenschaft.

```
$ VARIANT Value [get, set]
```

Value liefert und setzt den Wert der Eigenschaft. Diese Eigenschaft ist die Standardeigenschaft der IProperty-Schnittstelle.

#### Parameter:

[out]: pVal (VB-Rückgabewert) Wert der Eigenschaft.

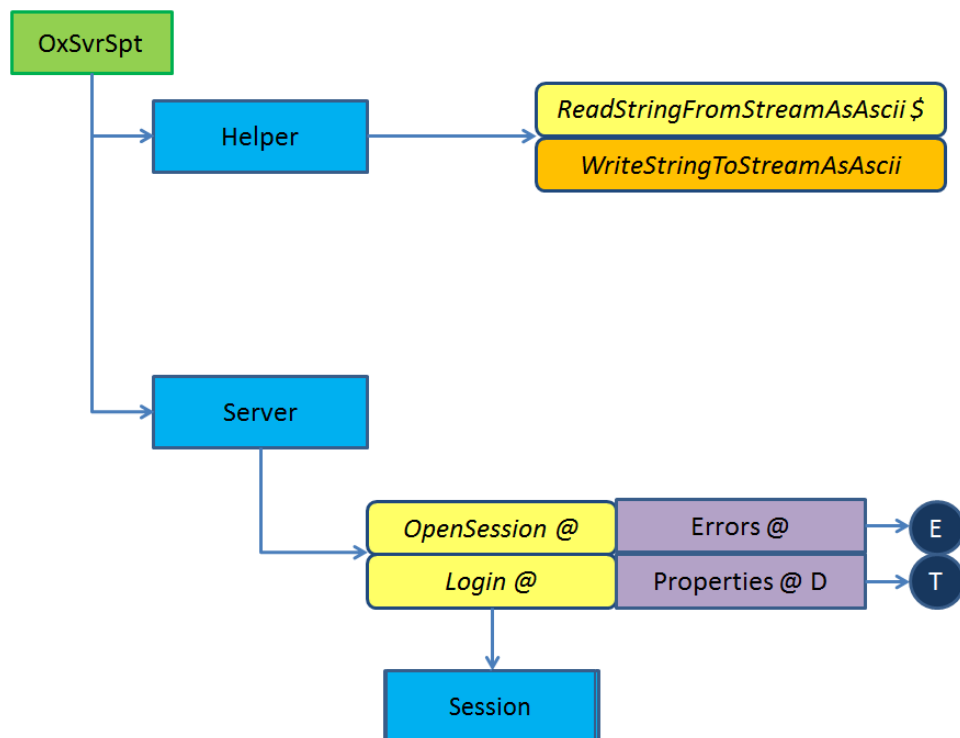
[in]: Val Wert der Eigenschaft.

## IServer

### Beschreibung:

IServer ist der zentrale Einstiegspunkt der Bibliothek. Es ist das einzige, erstellbare Objekt in der Hierarchie für den Zugriff auf den Server.

```
import "OxSvrSpt.idl"
```



### Öffentliche Methoden:

```

HRESULT Login ([in, defaultvalue("")] BSTR User, [in, defaultvalue("")]
BSTR Password, [in, defaultvalue("")] BSTR Server, [in, defaultvalue("")]
BSTR Port, [in, defaultvalue(pwNotEncrypted)] PasswortTypeEnum
PasswortType, [in, defaultvalue()] VARIANT_BOOL DefaultSession, [out,
retval] ISession ** ppSession)

```

```

HRESULT Connect ([in, defaultvalue("localhost")] BSTR Server, [in,
defaultvalue("4000")] BSTR Port)

```

```

HRESULT OpenSession ([in, defaultvalue("")] BSTR SessionGUID, [in,
defaultvalue("")] BSTR Alias, [out, retval] ISession ** ppSession)

```

```

HRESULT LoginBalanced ([in, defaultvalue("")] BSTR User, [in,
defaultvalue("")] BSTR Password, [in, defaultvalue("")] BSTR ServerList,
[in, defaultvalue(pwNotEncrypted)] PasswortTypeEnum PasswortType, [in,
defaultvalue()] VARIANT_BOOL DefaultSession, [out, retval] ISession **
ppSession)

```

```

HRESULT LoginGUID ([in, defaultvalue("")] BSTR GUID, [in,
defaultvalue("")] BSTR Server, [in, defaultvalue("")] BSTR Port, [in,
defaultvalue()] VARIANT_BOOL DefaultSession, [out, retval] ISession **
ppSession)

```

### Properties:

```

IProperties Properties [get]

```

```

IErrors Errors [get]

```

### Dokumentation der Elementfunktionen:

```
$ HRESULT Connect ([in, defaultvalue("localhost")] BSTR Server, [in, defaultvalue("4000")] BSTR Port)
```

Connect stellt eine Verbindung zu dem angegebenen Servern her.

Diese Methode liefert zurzeit den Fehler E\_NOTIMPL zurück!

Der Aufbau einer Verbindung zu einem Server ohne eine Anmeldung ist notwendig, wenn die Eigenschaften des Servers benötigt werden. Diese werden nach einem Connect in den Properties zur Verfügung gestellt. Werden keine dieser Daten benötigt, ist es nicht notwendig erst ein Connect zum Server aufzubauen bevor man sich einloggt. Die Serverdaten können in diesem Fall direkt beim Login mit angegeben werden.

**Parameter:**

[in]: Server (Standardwert ist localhost) IP-Adresse oder Name des Servers, zu dem die Verbindung aufgebaut werden soll

[in]: Port (Standardwert ist 4000) Port des Servers

```
$ HRESULT Login ([in, defaultvalue("")] BSTR User, [in, defaultvalue("")] BSTR Password, [in, defaultvalue("")] BSTR Server, [in, defaultvalue("")] BSTR Port, [in, defaultvalue(pwNotEncrypted)] PasswortTypeEnum PasswortType, [in, defaultvalue()] VARIANT_BOOL DefaultSession, [out, retval] ISession ** ppSession)
```

Login führt eine Anmeldung am Server durch und liefert die betreffende Session zurück.

Wurde vor dem Aufruf von Login bereits eine Verbindung zu einem Server über die Methode Connect hergestellt, werden diese Verbindungsdaten für das Login verwendet, falls keine angegeben wurden.

Werden sowohl der User-, als auch der Password-Parameter leer oder mit einer leeren Zeichenkette übergeben, wird versucht die Anmeldung über das automatische Login durchzuführen. Dieses muss im enaio® administrator aktiviert sein und ist nicht mit NTLM-Authentifikation gleichzusetzen.

**Parameter:**

[in]: User Nutzer-Account

[in]: Password Passwort des zu verwendenden Nutzer-Accounts

[in]: Server IP-Adresse oder Name des Servers, zu dem die Verbindung aufgebaut werden soll

[in]: Port Port des Servers

[in]: PasswortType gibt an, ob das übergebene Passwort bereits verschlüsselt wurde. Wird dieser Parameter nicht angegeben, wird davon ausgegangen, dass das Passwort unverschlüsselt übergeben wurde.

[in]: DefaultSession (Standardwert ist VARIANT\_FALSE) gibt an, ob es sich um die Session handelt, an der sich der Benutzer später mit Hilfe von OpenSession anhängen kann.

[out]: ppSession (VB-Rückgabeparameter) erstellte Session für den Account.

**Ausnahmebehandlung:**

errLoginUnknownUser (602) Der angegebene Benutzer existiert nicht.

errLogin3TimesWrong (603) Der dritte Login-Versuch schlug fehl.

errLoginInvalidPassword (604) Das angegebene Passwort ist falsch.

errLoginLocked (605) Der Benutzeraccount ist gesperrt.

```
§ HRESULT LoginBalanced ([in, defaultvalue("")] BSTR User, [in,
    defaultvalue("")] BSTR Password, [in, defaultvalue("")] BSTR
    ServerList, [in, defaultvalue(pwNotEncrypted)] PasswortTypeEnum
    PasswortType, [in, defaultvalue()] VARIANT_BOOL DefaultSession, [out,
    retval] ISession ** ppSession)
```

LoginBalanced führt eine Anmeldung am Server durch und liefert die betreffende Session zurück.

Werden sowohl der User-, als auch der Password-Parameter leer oder mit einer leeren Zeichenkette übergeben, wird versucht die Anmeldung über das automatische Login durchzuführen.

**Parameter:**

[in]: User      Nutzer-Account

[in]: PasswordPasswort des zu verwendenden Nutzer-Accounts

[in]: ServerList      Liste mit den Servern, auf die sich die Bibliothek verbinden soll. Diese Liste hat folgenden Aufbau: Server1#Port1#Wichtung1;Server2#Port2#Wichtung2#;

Server3#Port3#Wichtung3

[in]: PasswortType      gibt an, ob das übergebene Passwort bereits verschlüsselt wurde. Wird dieser Parameter nicht angegeben, wird davon ausgegangen, dass das Passwort unverschlüsselt übergeben wurde.

[in]: DefaultSession(Standardwert ist VARIANT\_FALSE) gibt an, ob es sich um die Session handelt, an der sich der Benutzer später mit Hilfe von OpenSession anhängen kann.

[out]: ppSession      (VB-Rückgabeparameter) erstellte Session für den Account.

**Ausnahmebehandlung:**

errLoginUnknownUser (602) Der angegebene Benutzer existiert nicht.

errLogin3TimesWrong (603) Der dritte Login-Versuch schlug fehl.

errLoginInvalidPassword (604) Das angegebene Passwort ist falsch.

errLoginLocked (605) Der Benutzeraccount ist gesperrt.

```
§ HRESULT LoginGUID ([in, defaultvalue("")] BSTR GUID, [in,
    defaultvalue("")] BSTR Server, [in, defaultvalue("")] BSTR Port, [in,
    defaultvalue()] VARIANT_BOOL DefaultSession, [out, retval] ISession **
    ppSession)
```

LoginGUID führt eine Anmeldung mit der übergebenen SessionGUID am Server aus und liefert die betreffende Session zurück

Mittels der SessionGUID wird eine Verbindung mit einer bestehenden Session auf dem Server aufgebaut.

**Parameter:**

[in]: GUID      GUID der Server-Session, mit der das Session-Objekt arbeiten soll

[in]: Server IP-Adresse oder Name des Servers, zu dem die Verbindung aufgebaut werden soll

[in]: Port Port des Servers

[in]: DefaultSession (Standardwert ist VARIANT\_FALSE) gibt an, ob es sich um die Session handelt, an der sich der Benutzer später mit Hilfe von OpenSession anhängen kann

[out]: ppSession (VB-Rückgabeparameter) erstellte Session für den Account

#### **Ausnahmebehandlung:**

errLoginLocked (605) Der Benutzeraccount ist gesperrt.

```
$ HRESULT OpenSession ([in, defaultvalue("")] BSTR SessionGUID, [in,
    defaultvalue("")] BSTR Alias, [out, retval] ISession ** ppSession)
    OpenSession erstellt ein neues Session-Objekt und verbindet dies mit der
    DefaultSession.
```

#### **Parameter:**

[in]: SessionGUID GUID der bestehenden Session, mit der die Verbindung aufgebaut werden soll

[in]: Alias beliebiger Name für den Aufruf, um Fehler oder Zustand zuzuordnen

[out]: ppSession (VB-Rückgabeparameter) erstellte Session für den Account

#### **Dokumentation der Properties:**

```
$ IErrors Errors [get]
    Errors liefert die Fehler-Collection mit den Fehlern des Serverzugriffs zurück.
```

#### **Parameter:**

[out]: pVal (VB-Rückgabewert) Fehler-Collection

```
$ IProperties Properties [get]
    Properties liefert die Collection mit den Eigenschaften des Servers zurück
    Derzeit werden bei der Erstellung des Serverobjektes folgende Eigenschaften durch die
    OxSvrSpt-Bibliothek gesetzt:
```

#### **TempDir:**

Beinhaltet das temporäre Verzeichnis, in dem die Dateien der Fileparameter abgelegt werden sollen. Dieses wird mit dem temporären Verzeichnis des Benutzers initialisiert.

#### **NotifyNeeded:**

wird mit 0 initialisiert.

Wird der Wert auf einen Wert ungleich 0 (oder VARIANT\_FALSE) gesetzt, wird die Unterstützung für die Notifications aktiviert. Es besteht dann die Möglichkeit über die Event-Schnittstelle des Server- und des Session-Objekts Notifications zu verarbeiten.

Die angegebenen Bezeichner entsprechen den Namen der Parameter.

#### **Parameter:**

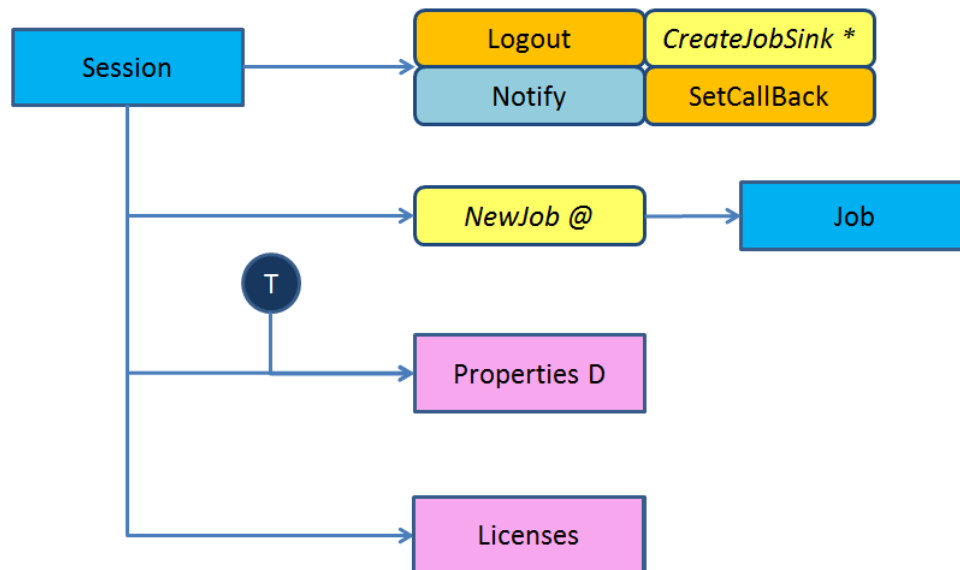
[out]: pVal (VB-Rückgabeparameter) IProperties-Schnittstelle

## ISession

### Beschreibung:

**ISession** wird von der Server-Schnittstelle nach einer erfolgreichen Anmeldung zur Verfügung gestellt. Session repräsentiert eine offene Verbindung zu einem Server. Es stellt die Möglichkeit zum Aufruf von Serverjobs zur Verfügung.

```
import "OxSvrSpt.idl"
```



### Öffentliche Methoden:

```
HRESULT Logout ()
```

```
HRESULT NewJob ([in] BSTR Name, [out, retval] IJob ** ppJob)
```

```
HRESULT CreateJobSink ([out, retval] VARIANT * pJobSink)
```

```
HRESULT SetCallBack ([in] long ICallbackType, [in] IDispatch * pUnkSink,
[in] long IUserData)
```

### Properties:

```
IProperties Properties [get]
```

```
ILicenses Licenses [get]
```

### Dokumentation der Elementfunktionen:

§ HRESULT CreateJobSink ([out, retval] VARIANT \* pJobSink)

CreateJobSink liefert das IJobSink-Interface der zugrundeliegenden OxSvrCom-Bibliothek eine ausführliche Beschreibung ist der Dokumentation der Bibliothek OxSvrCom.dll zu entnehmen.

#### Parameter:

[out]: pJobSink (VB-Rückgabeparameter) erstellte IJobSink-Schnittstelle

§ HRESULT Logout ()



Logout beendet die Arbeit mit dieser Session.

Nach einem Session-Logout lösen alle Zugriffe auf die anderen Methoden und Eigenschaften der Session den Fehler `errNoSession` aus.

### Ausnahmebehandlung:

`errNoSession` (1501) Die zugrunde liegende Session steht nicht mehr zur Verfügung. Diese Session wurde entweder geschlossen oder das Session-Objekt wurde freigegeben.

\$ HRESULT NewJob ([in] BSTR Name, [out, retval] IJob \*\* ppJob)

NewJob erzeugt ein neues Job-Objekt für den Job mit dem übergebenen Namen

### Parameter:

[in]: Name Name des Jobs. Dieser setzt sich aus Namespace.Job zusammen

[out]: ppJob (VB-Rückgabeparameter) enthält das erstellte Job-Objekt

\$ HRESULT SetCallBack ([in] long ICallbackType, [in] IDispatch \* pUnkSink, [in] long IUserData)

SetCallBack setzt die IJobSink-Schnittstelle für Callbacks in der zugrunde liegenden OxSvrCom-Bibliothek

Eine ausführliche Beschreibung ist der Dokumentation der Bibliothek OxSvrCom.dll zu entnehmen.

### Parameter:

[in]: ICallbackType Dokumentation ICallbackType

[in]: pUnkSink Dokumentation pUnkSink

[in]: IUserData Dokumentation IUserData

### Dokumentation der Properties:

\$ ILicenses Licenses [get]

Licenses liefert die Collection der aktuell angemeldeten Lizenzen zurück.

### Parameter:

[out]: pVal (VB-Rückgabewert) liefert eine COM-Collection mit den Lizenzen

\$ IProperties Properties [get]

Properties liefert Server- und Sessioneigenschaften für dieses Objekt zurück.

### Parameter:

[out]: pVal (VB-Rückgabewert) liefert eine COM-Collection mit den betreffenden Eigenschaften zurück. Die Collection enthält Objekte vom Typ `IProperty`

## Klassenhierarchie

\$ \_INotifiactionEvent

\$ IError

\$ IErrors

\$ INotifyErrors

- § IFileParameter
- § IHelper
- § IInputFileParameters
- § INotifyOutputFileParameters
- § IInputParameters
- § INotifyOutputParameters
- § IJob
- § ILicenses
- § ILogger
- § INotifyJob
- § IOutputFileParameters
- § INotifyInputFileParameters
- § IOutputParameters
- § IInputParameters
- § IParameter
- § IProperties
- § IProperty
- § IServer
- § ISession

# Index

- abn.Add 12
- abn.AddRevisit 22
- abn.ChangeRevisitUser 22
- abn.CheckOsrevisit 13
- abn.GetAboGrpList 13
- abn.GetDocList 14
- abn.GetRequestList 16
- abn.GetUserList 17
- abn.NotifyAbonnement 18
- abn.NotifyRequestAbo 18
- abn.RemoveAboIdent 19
- abn.RemoveAllObjAboNotifyFromUser 20
- abn.RemoveObjAboNotifyFromUser 20
- abn.RemoveObjRevisitNotifyFromUser 21
- abn.SetObjRevisitOpen 21
- abn.UpdateReqAboGrp 19, 21
- abn.UpdateRevisit 23, 28
- adm.CleanUpConfig 305
- adm.CleanUpLog 306
- adm.EnumServerGroups 306
- adm.EnumServers 307
- adm.GetServerFamilyInfo 307
- adm.GetServersActivity 307
- adm.GetSystemFile 308
- adm.GetUserProfile 162
- adm.LogdirDeleteFiles 308
- adm.LogdirDownloadFiles 309
- adm.LogdirGetInfo 309
- adm.StoreSystemFile 310
- adm.StoreUserProfile 165
- Administration 258
- Administration und Historienverwaltung 259
- ado.ExecuteSQL 30
- BatchJobs 302
- Batch-Verwaltung 312
- cnv.GetExifData 35
- cnv.GetIcons 35
- cnv.GetPageCount 36
- cnv.GetPictureInfos 36
- cnv.GetRendition 37
- cnv.AddAnnotations 34
- cnv.ConvertDocument 31
- cnv.CreateSlide 34
- CreateLaboratoryReport 141
- Datumsformate 107
- DMS - Anfragetypen und Ergebnisformate 71
- DMS.AddPortfolio 121
- DMS.AddRel 116
- DMS.AddRelText 117
- DMS.AddRelTextLang 117
- DMS.CheckInDocument 52
- DMS.CheckOutDocument 53
- DMS.CheckPermission 109, 110
- DMS.CopySD 111
- DMS.CreateSD 112
- DMS.DeleteSD 112
- DMS.DeleteUserData 125
- DMS.DelPortfolio 122
- DMS.DelRel 117
- DMS.DelRelText 118
- DMS.GetObjDef 24, 26, 67, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104
- DMS.GetUserData 125
- DMS.GetUserDataNames 125
- DMS.GetXMLJobOptions 53, 54
- DMS.GetXMLSchema 126
- DMS.IsUserData 126
- DMS.ModPortfolio 122
- DMS.ModRel 118
- DMS.ModRelText 118
- DMS.ModRelTextLang 119
- DMS.ReadSD 113
- DMS.RetrievePortfolios 123
- DMS.RetrieveRelations 119
- DMS.RetrieveRelTexts 120
- DMS.SetSD 113
- DMS.SetUserData 126
- DMS.UndoCheckOutDocument 53
- DMS.XMLCopy 60
- DMS.XMLDelete 54
- DMS.XMLInsert 55, 63
- DMS.XMLMove 58
- DMS.XMLUpdate 61, 65
- Erweiterte Informationen 89
- Historienverwaltung 282
- Interne Jobs 184
- krn.AppsEventsEnum 315
- krn.AppsEventsSubscribe 316
- krn.BatchAdd 312
- krn.BatchChange 313
- krn.BatchEnum 313
- krn.BatchGetStatistic 314
- krn.BatchRemove 315
- krn.CheckCrashedServers 316
- krn.CheckDiskSpace 332
- krn.CheckServerConnection 316

krn.EnumJobs 326  
 krn.EnumModules 329  
 krn.EnumNameSpaces 326  
 krn.GetCounter 335  
 krn.GetFileVersionList 332  
 krn.GetNameSpaceParams 327  
 krn.GetNextIndex 333  
 krn.GetServerInfo 317  
 krn.GetServerInfoEx 318  
 krn.JobThreadBreak 330  
 krn.JobThreadGetInfo 330  
 krn.LoadExecutor 327  
 krn.MakeBeatPing 319  
 krn.NameSpaceEnum 328  
 krn.NameSpaceGetInfo 328  
 krn.NameSpaceGetJobsInfo 328  
 krn.ProcessGetInformation 335  
 krn.QueueEnum 330  
 krn.QueueGetParams 331  
 krn.QueueGetStatistic 331  
 krn.REBackup 311  
 krn.RefillServerList 319  
 krn.REGetCurrentSchema 311  
 krn.REGetRegValue 311  
 krn.RELoad 311  
 krn.ReloadExecutor 329  
 krn.RESave 312  
 krn.RESetRegValue 312  
 krn.RunScript 333, 334  
 krn.SendAdminMail 334  
 krn.SendMail 334  
 krn.SendMessageToClients 335  
 krn.SessionAttach 320  
 krn.SessionDeleteLost 320  
 krn.SessionDrop 320  
 krn.SessionDropDB 321  
 krn.SessionEnum 321  
 krn.SessionEnumDB 321  
 krn.SessionEnumResourcesDB 321  
 krn.SessionGetInfo 322  
 krn.SessionLogin 322  
 krn.SessionLogout 323  
 krn.SessionPropertiesEnum 323  
 krn.SessionPropertiesGet 323  
 krn.SessionPropertiesSet 324  
 krn.ShutDown 319  
 krn.UnloadExecutor 329  
 krn.UserSessionCreate 325  
 krn.UserSessionDelete 325  
 lic.CheckLicense 337  
 lic.LicCopyDefault 337  
 lic.LicFreeResource 337  
 lic.LicGetGlobalInfo 338  
 lic.LicGetGlobalInfoEx 338  
 lic.LicGetModuleInfo 339  
 lic.LicGetQueueStatus 339  
 lic.LicLogin 340  
 lic.LicLoginEx 340  
 lic.LicLogout 341  
 lic.LicLogoutEx 341  
 lic.LicResetData 341  
 lic.LicSetGlobalInfo 342  
 LoincObservations 135  
 LoincResults 132  
 LoincUnits 135  
 LoincViewSets 136  
 Medizin - Engine (Namespace med) 128  
 mng.GetGroupAttributes 153, 154, 155, 156, 157  
 mng.GetGroupList 158  
 mng.GetGroupMembers 159  
 mng.GetUserAttributes 159  
 mng.GetUserGroups 161, 163, 164  
 mng.GetUserList 161  
 ObservationInsert 142  
 ObservationRequestHistory 144  
 ObservationResultHistory 143  
 ObservationValues 145  
 ocr.DoDocOCR 166  
 ocr.DoOCR 166  
 OxSvrSpt  
   \_INotificationEvents 360  
   IError 360  
   IErrors 361  
   IFileParameter 363  
   IHelper 367  
   IInputFileParameters 369  
   IInputParameters 371  
   IJob 377  
   ILicenses 380  
   ILogger 381  
   INotifyErrors 384  
   INotifyInputFileParameters 385  
   INotifyInputParameters 385  
   INotifyJob 385  
   INotifyOutputFileParameters 386  
   INotifyOutputParameters 386  
   IOutputFileParameters 386  
   IOutputParameters 387  
   IParameter 388  
   IProperties 393  
   IProperty 394  
   IServer 395  
   ISession 400  
 Parametrisierung von Anfragen 87  
 PatientData 136  
 Registry-Verwaltung 310  
 SaveMedicalRecord 146  
 ServerCommunicationJobs 303  
 Serverinterne Jobs 302  
 Server-Verwaltung 315  
 Session-Verwaltung 319  
 Sonstige Jobs 291  
 Standard - Engine (Namespace std) 167

std.AdjustRetentions 184  
 std.CalcDocumentDigest 174  
 std.CheckSource 185  
 std.CleanUpCache 167  
 std.ClearFromCache 168  
 std.ConfigVarc 186  
 std.DeleteDocument 175  
 std.DeleteDocumentVersion 174  
 std.DeleteObject 174  
 std.DeleteRemark 175  
 std.DiskSpace 186  
 std.DoArchive 168  
 std.DoPrefetch 168  
 std.FileTransfer 186  
 std.GetDocStatistics 176  
 std.GetDocStream 176  
 std.GetDocumentDigest 173, 175, 177  
 std.GetDocumentPage 177  
 std.GetDocumentStream 177  
 std.GetDocVersion 178, 179  
 std.GetObjectInfo 179  
 std.GetRemark 180  
 std.GetSignedDocument 180  
 std.IndexDataChanged 187, 188  
 std.MergeDocuments 181  
 std.MergeFolder 181  
 std.MoveToCache 169  
 std.ObjectTransfer 184  
 std.PackDirectory 188, 189  
 std.RestoreDocVersion 181  
 std.RestoreObject 182  
 std.SetHistory 182  
 std.SetPlannedRetention 184  
 std.StoreInCache 170  
 std.StoreInCacheByID 170  
 std.StoreInCacheDirect 171  
 std.StoreInWork 172  
 std.StoreRemark 182  
 std.StoreSignedDocument 183  
 std.TransformIndexData 189  
 std.UndoArchive 172  
 std.Unknown2Known 183  
 Suchbedingungen 79  
 Systemfelder 105  
 UpdatePatientId 141  
 UpdateVisitId 141  
 Volltext - Engine (Namespace vtx) 191  
 vtx.CleanupClient 191  
 vtx.CloseQuery 192  
 vtx.GetDocument 192  
 vtx.GetEngineName 192, 193, 194  
 vtx.OpenObjectQuery 195  
 vtx.OpenWordListQuery 196  
 wfm.AdminDeleteProcesses 259  
 wfm.AdminGetActivityVariables 260  
 wfm.AdminGetLockInfo 275  
 wfm.AdminGetProcessActivities 261  
 wfm.AdminGetProcessList 263  
 wfm.AdminGetProcessListByRole 264  
 wfm.AdminGetProcessListByUser 266  
 wfm.AdminGetProcessLocks 277  
 wfm.AdminGetProcessReport 278, 279, 280, 281  
 wfm.AdminGetRoleProcesses 268  
 wfm.AdminGetUserProcesses 269  
 wfm.AdminGetWorkerqueue 276  
 wfm.AdminGetWorkflowList 269, 270, 271  
 wfm.AdminReleaseLock 278  
 wfm.AdminRollbackProcess 271  
 wfm.AdminSaveActivityVariables 272, 273  
 wfm.AdminSuspendActivity 273  
 wfm.AdminSuspendProcess 274, 275  
 wfm.CancelWorkItem 225  
 wfm.ChangeWorkflowState 206  
 wfm.CheckJob 303  
 wfm.CompleteWorkItem 226, 298, 300  
 wfm.ConfigUserAbsence 197  
 wfm.ConvertExportFile 292  
 wfm.CopyWorkflow 206  
 wfm.CreateProcessInstance 228  
 wfm.DBCommands 302  
 wfm.DeleteEvent 248  
 wfm.DeleteMasks 249  
 wfm.DeleteOrganisation 198  
 wfm.DeleteScript 249  
 wfm.DeleteSysClienttypes 297  
 wfm.DeleteWorkflow 207  
 wfm.Export 292  
 wfm.GetAbsentUsers 198  
 wfm.GetActivityPerformers 229  
 wfm.GetEvents 249  
 wfm.GetEventTypes 251  
 wfm.GetGlobalScripts 252  
 wfm.GetHistActivitiesByProcess 282  
 wfm.GetHistEntries 283  
 wfm.GetHistProcessList 284  
 wfm.GetHistTimerEntries 285  
 wfm.GetHistTimersByProcess 286  
 wfm.GetHistVariablesByHistEntry 287  
 wfm.GetHistWorkflowList 287  
 wfm.GetHistWorkItemRelActivitiesByProcesses 288  
 wfm.GetHistWorkItemRelEntriesByActivity 289  
 wfm.GetHistWorkItemRelEntriesByUser 290  
 wfm.GetHistWorkItemRelUsersByProcess 290  
 wfm.GetOrganisationClasses 199  
 wfm.GetOrganisationObjects 200  
 wfm.GetOrganisations 202  
 wfm.GetProcessList 230, 231  
 wfm.GetProcessProtocol 232  
 wfm.GetProcessResponsibles 232  
 wfm.GetRunningActivities 233

wfm.GetSubstitutes 202, 203  
wfm.GetSysClienttypes 296, 297, 298  
wfm.GetVersionInfo 293  
wfm.GetWFInfo 293  
wfm.GetWorkflow 207  
wfm.GetWorkflowData 208  
wfm.GetWorkflowInfo 209  
wfm.GetWorkflowList 209  
wfm.GetWorkflowListByFamily 210  
wfm.GetWorkItem 242  
wfm.GetWorkItemList 234  
wfm.GetWorkItemParams 236  
wfm.Import 294  
wfm.LoadMasks 252  
wfm.LoadScript 255  
wfm.SaveEvent 255  
wfm.SaveMasks 256  
wfm.SaveOrganisation 204  
wfm.SaveScript 257  
wfm.ServerNotifyClients 303  
wfm.ServerUpdateWorkflowModels 304  
wfm.ServerUserAbsent 304  
wfm.SetActiveOrganisation 204  
wfm.SetActivityPerformers 240  
wfm.SetEventScriptRelation 257  
wfm.SetProcessResponsibles 240  
wfm.SetSubstitutes 205  
wfm.StartProcess 240  
wfm.StartWorkItem 242  
wfm.StoreWorkflow 211  
wfm.ValidateWorkflow 212  
wfm.WorkerJob 303  
wfm.WorkItemNoti 303  
Workflow - Engine (Namespace wfm) 197