Handbuch

OMS-ReportWriter 5.7

profiforms
ProfiForms Projekt GmbH Hausener Str. 6/1 D-71296 Heimsheim +49/7033 3012-0 www.profiforms.de

Rechtliche Hinweise

Der Inhalt dieses Handbuches ist das geistige Eigentum der ProfiForms Projekt GmbH.
Bei der Erstellung der Texte und Abbildungen dieses Handbuches wurde mit größter Sorgfalt vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden. Die ProfiForms Projekt GmbH übernimmt keinerlei Gewähr für die Aktualität, Korrektheit und Vollständigkeit der bereitgestellten Informationen.

Die ProfiForms Projekt GmbH behält sich das Recht vor, den Inhalt dieses Handbuches ohne vorherige Ankündigung zu verändern oder ergänzen und übernimmt keine Haftung für Fehler in diesem Handbuch oder daraus resultierende mögliche Schäden.

Für Verbesserungsvorschläge und Hinweise auf Fehler sind Herausgeber und Autoren dankbar.

Diese Software wird gebündelt mit den Schriftarten des DejaVu-Projekts ausgeliefert. DejaVu ist eine Sammlung von verschiedenen, unter freier Lizenz stehenden Schriftarten, die ihren Ursprung in der Schriftartensammlung Bitstream Vera haben. Die Schriftarten stehen unter dem Bitstream Vera Fonts Copyright und dem Arve Fonts Copyright, welche als Lizenzverträge mit installiert werden. DejaVu ist kein preislicher Bestandteil dieses Produkts. ProfiForms übernimmt für diese Schriften weder Garantie noch Wartung. Alle Rechte bezüglich dieser Schriften liegen bei Bitstream und dem DejaVu-Projekt.

SAP, SAP R/3, SAPScript, SmartForms, BC-RDI, BC-XFP und andere sind eingetragene Warenzeichen der SAP AG, Walldorf.
Java ist eine eingetragene Marke der Oracle Corporation.
Adobe, Abobe Present, Adobe Central, Adobe Designer, PostScript, PDF, XDP und weitere Warenzeichen sind eingetragene Warenzeichen der Adobe Systems Incorporated.
Hewlett Packard, HP-PCL sind eingetragene Warenzeichen der Hewlett-Packard Company.
Unix ist ein Warenzeichen der Open Group.
Windows ist ein eingetragenes Warenzeichen der Microsoft Corporation.
TBarcode ist ein eingetragenes Warenzeichen der TEC-IT Datenverarbeitung GmbH.

Alle anderen Firmennamen und Produktbezeichnungen sind Warenzeichen oder eingetragene Warenzeichen der jeweiligen Firmen und unterliegen im Allgemeinen warenzeichen-, marken- und/oder patentrechtlichem Schutz.

WHAT'S NEW

1. **HTML und XHTML Textsupport**

   Wenn Textfelder mit bereits vorformatierten Texten gefüllt werden sollen, so geschah das bis jetzt immer über Inline-Sequenzen. In der letzten Zeit sind datenliefernde Systems aber immer mehr in der Lage fertig formatiertes HTML zu erzeugen. Aus diesem Grund wurde Textinterpretation so erweitert, dass Texte für Textfelder auch HTML und XHTML darstellen können. Ebenfalls unterstützt ist das Markup-Format des Adobe Textlayout-Frameworks aus dem Flash-Player.

2. **AXTE-Support**

   Die Adobe XDP Text Engine (AXTE) ist der Formatierer, der Texte im Adobe LiveCycle-Designer darstellt. Um bessere Ergebnisse bei der Darstellung von Texten aus dem Adobe LiveCycle-Designer zu erreichen, wurde die Setzmethode nun implementiert und ist der neue Standard.

3. **Seitenrotation, Bundsteg und Ankerpunkte**

   Sollen PDFs direkt für den Druck erstellt werden, so ist es sinnvoll eine Seite so eindrehen zu können, dass diese optimal produziert werden kann. Dies ist jetzt möglich. Da es aber häufig Designs gibt, die nicht mit gedreht werden sollen, hat diese Technik Auswirkungen auf die Designs und deren Ankerpunkte. Lesen Sie dazu den neuen Abschnitt Designs und Ebenen. Weiter gibt es jetzt einen Bundsteg für Rückseiten, die ebenso Auswirkungen auf Designs und Ebenen haben.

4. **Materialbeschreibung für Pages und Insertions**

   Jede Seite die erzeugt und jede Beilage benutzt wird besitzt jetzt eine Materialbeschreibung. Damit ist es internen und externen Logistikprozessen möglich immer sinnvoll zu arbeiten. Die Materialbeschreibung ist Voraussetzung für die kommende Produktionssteuerung. Die neue Technik hat erste Auswirkungen im EnvelopeSortSystem, da dort Beilagen mit vollständiger Materialbeschreibung definiert werden können.

5. **UTF16 Support für TCI- und INI-Dateien und für JFND- und RDI-Dateien**

   Der Unicode-Standard setzt sich immer weiter durch. Nach einer Übergangszeit mit UTF-8 basierten Texten können nun immer stärker auch UTF-16 Texte. An allen Stellen, an denen Text-Dateien gelesen werden, sind jetzt UTF-16 BE und UTF16-LE als Encoding unterstützt. Zur Erkennung dieser Formate benötigt die Text-Datei eine Byte Order Mark (BOM) an Anfang der Datei.

6. Redaktionssystem

Die Erstellung von Dokumenten und Dokumentabschnitten aus einem IT-fernen Umfeld findet über OMS-Redaktionssystem statt. Dort werden Textsequenzen, Werbung und Beilagen definiert, die im Prozess der Dokumenterzeugung ausgewertet und umgesetzt werden müssen. So werden aus dem Redaktionssystem nun LayoutAreas, WhiteSpace-Werbung, logische und physikalische Beilagen umgesetzt.

7. Datenstruktur Job

In viele Projekten wurde gewünscht eine Datenstruktur zu haben, die neben dem Dokument existiert und für einen ganzen Job oder Lauf verfügbar ist. Eine solche Datenstruktur gibt es jetzt als Datenstruktur Job. Diese kann über das Prefix $job angesprochen werden.

8. Charts

Besonders getrieben durch die Anforderungen des EnWG kam der Wunsch auf, Diagramme (Charts) automatisch berechnen zu lassen. Da es im Adobe LiveCycle momentan keine Charts gibt, wurden diese als grafische Objekte im XDP in Anlehnung an den Standard von uns eingeführt. Das hat den Nachteil, dass die Konfiguration nicht grafisch im Adobe LiveCycle-Designer geschehen kann, sondern manuell als Text geschehen muss. Da dieses Handbuch keine XDP-Referenz ist, enthält es auch nicht die Syntax-beschreibung von Charts. Hierzu gibt es eine separate Dokumentation und den Service von ProfiForms dies im Kundenauftrag umzusetzen.

9. XTF-Support

In Prozessketten, in den die Erzeugung von Dokumenten nur einen Schritt darstellt, wird zumeist neben dem Dokument noch eine Information benötigt, die dokumentiert, was erzeugt wurde. Bisher hat diese Aufgabe die DocRef- und/oder SAPRef-Datei wahrgenommen. DocRef und SAPRef sind aber technisch am Ende ihrer Leistungsfähigkeit angekommen und so wurde ein neues Format, das XML Ticketing Format (XTF) eingeführt, das diese Aufgabe in Zukunft erfüllen soll. Die Formatbeschreibung selbst ist nicht Bestandteil dieser Dokumentation und kann separat von uns angefordert werden. Über einen Kommandozeilenparameter -xtf kann die Ausgabe von XFT-Dateien konfiguriert und angestoßen werden.

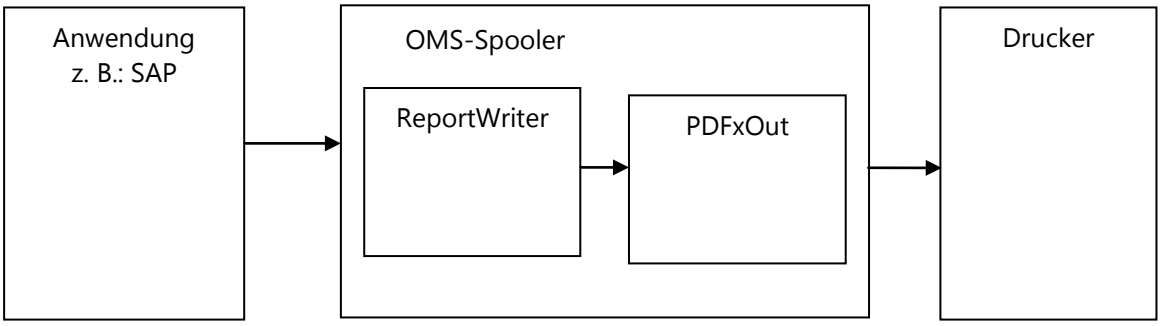
Inhalt

EINLEITUNG

Der OMS-ReportWriter ist ein Programm zur Umformung von Daten, die von ihrer Struktur her noch nicht für den Druck aufgearbeitet wurden, so dass diese typischen Drucklogiken entsprechen. Die Erzeugung von Druckjobs, Dokumenten, Seiten und dynamischen Absätzen steht im Mittelpunkt dieses Programms. Weiterhin fließen viele unterschiedliche Logiken aus dem Drucksektor, der Archivierung und der Papiernachverarbeitung ein. Das Ineinandergreifen von Logiken zur Generierung, Sortierung, Designsteuerung, Papiersteuerung und Archivierung führt zu einer hoch komplexen Applikation, die über Parameter gesteuert wird.

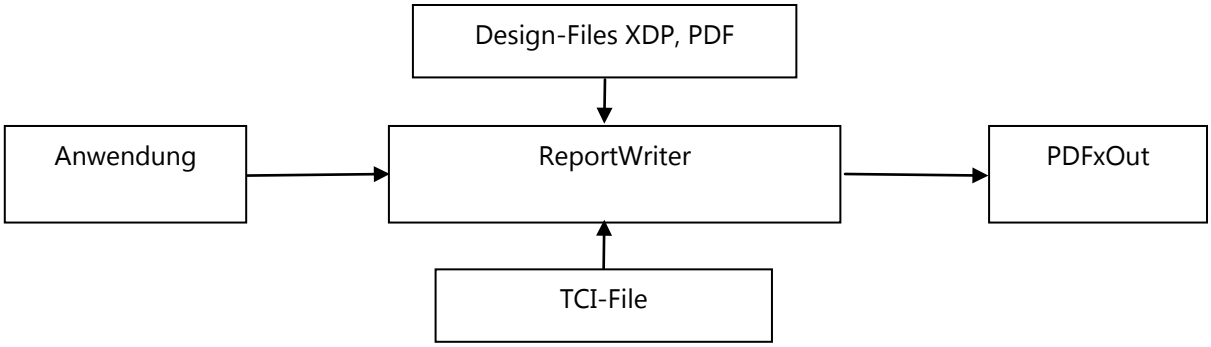
In einem Punkt unterscheidet sich der OMS-ReportWriter von klassischen SQL-ReportWritern.
Die Quelle des OMS-ReportWriters ist immer eine Datei, nie eine SQL-Datenbank. Als Designer verwendet der OMS-ReportWriter den Adobe Output-Designer und den Adobe Designer. Das Dokument wird vollständig im OMS-ReportWriter erstellt und als PDF + Ticketing ausgegeben.

Der OMS-ReportWriter läuft als Programm im OMS-Spooler. Sollen nur PDFs erzeugt werden, so kann der OMS-ReportWriter auch „stand alone" betrieben werden. Für die Ausgabe auf PostScript- oder PCL-Drucker benötigt der OMS-ReportWriter einen OMS-Spooler mit PDFxOut.



Die zu verarbeitenden Daten der Anwendung können vom OMS-ReportWriter in unterschiedlichen Formaten entgegengenommen werden. Alle diese Formate werden über Interpreter in ein einheitliches OMS-ReportWriter-Format überführt und verarbeitet. Zur Wahl steht heute eine an das Adobe Present/Format angelehnte Schnittstelle, die SAP-RDI-Schnittstelle sowie eine XML-Schnittstelle mit fester DTD. Überdies implementieren wir für unsere Kunden auf Wunsch auch firmenspezifische Schnittstellen. Zum Aufarbeiten der Anwenderdaten benötigt der OMS-ReportWriter genaue Informationen, die er aus der TCI erhält. Die TCI ist ein objektbasierter Konfigurations-File, der alle Einstellungen zur Erstellung der Dokumente aus dem Anwenderdatenstrom enthält. Große Teile der Einstellungen entnimmt der OMS-

ReportWriter den Design-Files, die im XDP-Format vorliegen. Neben diesen Konfigurationen gibt es für einige Module im OMS-ReportWriter eigene INI-Dateien, die das modulspezifische Verhalten beeinflussen.



Der OMS-ReportWriter arbeitet hauptsächlich mit den Begriffen Job, Dokument, Seite und Absatz. Die zentrale Verarbeitungslogik ist im Wechselverhältnis von Dokument und Seite implementiert.
Der OMS-ReportWriter erkennt eingangsseitig Dokument für Dokument und ordnet diese einer Verarbeitung zu. Dabei kann die Verarbeitung für jedes einzelne Dokument verschieden sein in Bezug auf die seines Vorgängers sowie Nachfolgers. Dies gilt für den Bereich der internen Verarbeitungslogik und in Bezug auf äußere Merkmale, wie Drucker oder Formular-File.

Ein erkanntes eingangsseitiges Dokument wird in der TCI durch den Begriff des DocDef repräsentiert. Gegenspieler hierzu ist das WorkItem. Ein WorkItem ist eine konkrete Ausgabe oder Verarbeitung des unter DocDef definierten Dokuments. Ein DocDef kann mehrere WorkItems zur Folge haben. Ein Beispiel dazu: Aus einem umfangreichen Datenstrom einer Anwendung sollen eine Rechnung, ein Zahlschein, ein Lieferschein und ein Etikett für den Spediteur gedruckt werden. Eingangsseitig ist das ein DocDef. Ausgangsseitig sind dies vier unterschiedliche WorkItems.

Ein durch ein WorkItem veranlasster Formulardruck kann ein statisches oder ein dynamisches Formular erzeugen. Handelt es sich um ein dynamisches Formular, so bedient sich der OMS-ReportWriter der Position der TCI.

Der OMS-ReportWriter unterscheidet strikt nach zentralen und dezentralen Ausdrucken. Als zentrale Ausdrucke werden alle Drucke angesehen, die weder im DocDef noch im WorkItem einen Drucker definieren bzw. den Schalter CENTRAL im WorkItem gesetzt haben. Dezentrale Ausdrucke sind alle anderen WorkItems.

Zentrale Drucke verfügen über eine erweiterte Ausgabelogik.
Dazu zählen:

- Separieren nach Druckkanälen (Ansteuern mehrkanaliger Kuvertiermaschinen)
- Separieren nach Portoklassen

- Separieren nach max. Seitenanzahl
- Separieren nach Anwenderwerten
- Separieren nach Dokumenttypen
- Generieren von OMR-Marken (Ansteuern von Kuvertierstraßen)
- Deutsche Post DV-Freimachung (EDV-Abrechnung von Sendungen der Deutschen Post)

AUFRUFKONVENTIONEN

Verwendung

Der OMS-ReportWriter kann zwei Parameter und eine Liste mit Optionen als Übergabeparameter verarbeiten:

reportw [OptionenListe] InputFile [TCI-File]

Erklärung

InputFile

InputFile bezeichnet den Namen der zu verarbeitenden Datei.

TCI-File

Der TCI-File ist ein optionaler Parameter und gibt den Namen der zu verwendenden TCI-Datei an. Fehlt dieser Parameter, so wird standardmäßig die Datei reportw.tci geladen.

OptionenListe

Die OptionenListe enthält einzelne Optionen, die mit dem Zeichen ‚-' beginnen.

-aap PATH: Pfad zum Archiv-System

-aip PATH: Pfad zu den Ini-Dateien

-aic CP: Codepage für Konfigurationsdateien, die über den LineReader gelesen werden (siehe CodePages im Kapitel Internationalisierung)

-all LogFile: Der FileName wird zum Logging aller Aktivitäten verwendet

-awp PATH: Setzt das aktuelle Arbeitsverzeichnis beim Programmstart

-aop PATH: Output Pfad

-asp PATH: SourceCopy Pfad

-atp PATH: Pfad für temporäre Dateien

-afp PATH: Pfad zu den Formularen (XDP- und PDF-Dateien) und den Bildern (TIF, JPG usw.) Dieser Pfad ist ein Suchpfad beim Laden von Ressourcen, wie Formularen und Bildern. Der Ressourcen-Pfad kann mehrfach angegeben werden, so dass in unterschiedlichen Pfaden nach den Ressourcen gesucht wird.

-alp PATH: Pfad zu den Logos und Bitmap-Dateien

-arp PATH: Ersatzpfad zu den Fonts-Dateien

-adv: Mit -adv werden alle globalen Variablen definiert, die in allen Dokumenten des Datenstroms gesetzt werden. Existiert in dem Dokument bereits eine Variable gleichen Namens, so wird diese überschrieben, andernfalls wird die Variable neu angelegt. Der Parameter -adv kann in der Kommandozeile mehrfach vorkommen, so dass unterschiedliche Variablen definiert werden können.

Syntax

-adv VARNAME=VOLUME

Beispiel:

```
reportw –adv PRN=OFFICEPRN1 –adv PRNSERVER=RW1 x.rdi
```

-ass CP: Der Datenstrom, der durch das Adobe Present (JetForm) Interface gelesen wird, benutzt die angegebene CodePage. Eine Liste möglicher CodePages finden Sie im Kapitel Internationalisierung.

-deb: Debug PDF. In das PDF-File werden zusätzliche Debug-Angaben mit gedruckt.
Um alle grafischen Objekte wird ein Rahmen gemalt, der zusätzlich den Namen des Objektes darstellt. Im Inhaltsverzeichnis des PDFs werden alle für das Dokument relevanten Werte als Strukturbaum ausgegeben. Darunter befinden sich auch alle Felder des Dokuments. Nicht existierende Fontressourcen oder Images werden ignoriert.

Vorsicht: Im Debug-Fall befinden sich Header-Pages am Ende des PDFs und nicht am Anfang.

-**stp**: Start-PDF. Unter dem Betriebssystem Windows ist es möglich, die erzeugten PDFs gleich mit einem PDF-Anzeigeprogramm darzustellen. Der Programmlauf wird erst beendet, wenn die Ergebnis-PDFs im aktuellen PDF-Reader angezeigt werden. Unter allen anderen Plattformen ist dieser Schalter ohne Wirkung.

-lan LAN: Reporting-Language: Die Log- und Fehler-Informationen können in Deutsch oder Englisch ausgegeben werden. Beim Start des Programms wird ermittelt, ob die Sprache des Betriebssystems Deutsch ist. In diesem Fall ist auch die Reporting-Language Deutsch, in allen anderen Fällen Englisch. Mit -lan kann die Reporting-Language willkürlich gesetzt werden.

 Syntax

-lanGER oder -lanENG

-lmt MT: Log-Message-Type: Um die Log-Informationen einzuschränken, können mit -lmt bestimmte Log-Message-Typen zur Ausgabe ausgewählt werden. Zulässige Typen sind:

TCI	Messages beim Lesen der TCI
GEN	Messages bei der Generierung aus den Nettodaten
LOG	Messages bei der Aufarbeitung im Logistics-Modul
PDF	Messages bei der Ausgabe von PDFs
OUT	Messages bei der Aufarbeitung für die Ausgabe
FRM	Messages beim Lesen von PDF und XDP Formularen und Designs
USR	Messages, die vom User erzeugt werden in Calc-Objekten
STA	Messages mit Statistik-Informationen

 Syntax

-lmtMT1[+MT2[...]]

```
Beispiel:

reportw –lmt FRM+STA x.rdi

Es werden nur Messages zum Lesen von Formularen und Statistik-
Informationen ausgegeben.
```

-rsp FILETYPE:

Type der Response-Datei: Mit dem Beenden des Programms kann eine Datei ins aktuelle Arbeitsverzeichnis geschrieben werden, die Informationen über den Programmlauf und dessen Erfolg enthält. Dabei sind folgende Werte einstellbar:

NONE	Es wird keine Response-Datei erzeugt (Standardwert)
XML	Es wird eine XML-Response-Datei im ProfiForms-Format erzeugt
JETFORM	Es wird ein Response-Datei im JetForm-Format erzeugt

-ftr CLASSES:

Der Parameter dient dazu zu bestimmen, welche Files in die XML-basierte Response-Datei aufgenommen werden und welche nicht. Der Kommandozeilen-Parameter wird gefolgt von einer Buchstabenfolge. Jeder Buchstabe steht für eine bestimmte Kasse von Files, die ausgegeben werden. Eine leere Buchstabenfolge steht für den Standardwert ALLE Files. Folgende Buchstaben bezeichnen folgende Klassen an Dateien:

X	XTF-Dateien
O	Output, zumeist PDF oder XML-Dateien mit Dokumenten
D	DVFreimachungsdateien für die AM-Exchange-Anmeldung und EABs
R	DocRef/SAPRef-Dateien
I	Archiv-Dateien

-nds: Unterbindung der Dokument-Sortierung (no document sort).

-sno SNO: Alternativ zur Angabe der Seriennummer in der reportw.ini kann diese auch als Kommandozeilenparameter übergeben werden. Die Seriennummer in der reportw.ini wird dann ignoriert.

-v: Bei der Angabe von -v erfolgt nur die Ausgabe der Versionsnummer auf dem Bildschirm. Danach beendet der OMS-ReportWriter seine Arbeit mit dem Return-Code 0.

-sft FILETIME:

Willkürliche Angabe der FileTime für das TransPromo-Modul im ISO-Format. Wird keine FileTime angegeben, so wird die Startzeit des Programms als FileTime gesetzt. Die FileTime steuert die Gültigkeit von TransPromo-Ressourcen.

```
Beispiel:

reportw –sft 2009-12-24T16:00CET
```

-vol: Für jede Ausgabedatei generiert der OMS-ReportWriter zusätzlich noch eine *.vol-Datei. Eine *.vol-Datei ist eine xml-Datei, welche die Ausgabedatei näher beschreibt. Die Option –vol ist von besonderem Interesse bei der Integration in den OMS-Spooler.

-xtf OPTIONS: Erstellung von Statistik- und Abrechnungsdateien als XTF (XML Ticketing Format). Eine XTF-Datei beschreibt die Ausgabe-Dateien aus Sicht wesentlicher Verarbeitungsmerkmale (siehe dazu XTF-Dokumentation). Die Erstellung kann durch Optionen gesteuert werden. Die Optionen werden als Buchstabenreihenfolge angegeben, bei der jeder Buchstabe für eine Option steht. Folgende Optionen sind möglich:

M	MetaData (Es werden Metadaten mit ausgegeben)
A	Archive (Es werden Archivindexdaten ausgegeben)
T	TransPromo (Es werden TransPromo-Resultate ausgegeben)
J	Join (Für alle Ausgaben wir nur eine einzige XTF-Datei erzeugt)
O	Optimize (Die Ausgabe wird optimiert)

-iap FILE: Filename mit optionalem Pfad, der den Filenamen reportw.ini ersetzt.

-ixm FILE: Filename mit optionalem Pfad, der den Filenamen xml.ini ersetzt.

-ird FILE: Filename mit optionalem Pfad, der den Filenamen rdi.ini ersetzt.

-idp FILE: Filename mit optionalem Pfad, der den Filenamen dpdv.ini ersetzt.

-ifo FILE: Filename mit optionalem Pfad, der den Filenamen fonts.ini ersetzt.

SYNTAX DER TCI-OBJEKTE

Alle Zeilen und Objekte werden über den LineReader als Preprozessor gelesen und unterstützen Includes und das Lesen aus Sections.

Basisobjekte

DocRec

Verwendung

Jedes hereinkommende Dokument muss durch den OMS-ReportWriter erkannt und einer Verarbeitung zugeführt werden. Das DocRec-Objekt ist eine Liste, die zur Erkennung hereinkommender Dokumente verwendet wird. In ihr werden Verweise auf DocDef-Objekte und dazugehörige Erkennungskriterien definiert. Weiterhin kann zu einem DocDef noch angegeben werden, in welcher TCI das nachzuladende DocDef zu finden ist. Die Liste der DocDefs wird von oben nach unten abgearbeitet. Dabei wird geprüft, ob die Erkennung eines DocDefs zu den hereinkommenden Daten passt. Das erste DocDef, bei dem die Erkennung den Wert wahr zurückliefert, ist das DocDef, das die Verarbeitung übernehmen muss.

Die Angabe eines zu prüfenden DocDefs kann in zwei Syntax-Varianten angegeben werden.
In der verkürzten Syntax befindet sich die komplette Definition auf einer Zeile.

Syntax

```
DocRec {
  ...
  DocDefName Field=Value [Field!Value [...]] [(TCIFileName)]
  ...
}
```

Jede Zeile der Liste enthält einen Verweis auf eine DocDef und definiert danach Feld-Inhalts-Paare, die mit einem Vergleichsoperator verknüpft sind. Als Operatoren stehen ein Gleich-Operator (=) und ein

Ungleich-Operator (!) zur Wahl. Die einzelnen Vergleichskriterien sind dabei Und-verknüpft. Optional kann in Klammern ein TCIFile angegeben werden, der zum Ausführen des DocDefs nachgeladen werden muss.

In der erweiterten Syntax befinden sich die Angaben zum zu prüfenden DocDef in einem strukturierten Objekt.

Syntax

```
DocRec {
  ...
  DocDefName {
    Recognition bzw. Rec {
      ...
    }
    TCIFileName
  }
}
```

Recognition bzw. **Rec**

Die Recognition steuert, ob ein Objekt ausgeführt wird oder nicht. Die Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben. Die Angabe ist optional.

TCIFileName

Der TCIFileName definiert den Filenamen einer nachzuladenden TCI, in der sich das entsprechende DocDef befindet. Die Angabe ist optional.

```
Beispiel:

DocRec {
 ...
 Invoice_SE     aprfk=R  rgspr=E
 Delivery_note {
  Rec {
   IsEqualS(aprfk,“D“)
   IsEqualS(rgspr,“E“)
  }
 }
 Credit_note_SF aprfk!R  rgspr=E (Credit_note.tci)
 ...
}
```

Die verkürzte und die erweiterte Syntax lassen sich in einem DocRec-Objekt miteinander mischen. Der Vorteil der verkürzten Syntax ist die kürzere Schreibweise mit dem Nachteil, dass nur sehr einfache Vergleichsoperatoren (= und !) zugelassen sind. In der erweiterten Syntax ist im Recognition-Objekt jeder Calc-Befehl zulässig, der auf Dokument-Ebene ausgeführt werden kann.

Da DocDef-Objekte selbst auch eine Recognition definieren können, wird die Erkennung eines DocDefs aus der Erkennung des DocDefs im DocRec und dem DocDef zusammengesetzt. Beim Aufbau der TCI können Sie sich entscheiden, ob die Erkennung der Dokumente ganz oder teilweise im DocRec oder im DocDef definiert wird.

DocDef

Verwendung

Ein DocDef-Objekt repräsentiert inputseitig ein Dokument und legt wichtige Verarbeitungsmerkmale fest, die für das Dokument über alle WorkItems hinweg von Bedeutung sind. Über ein WorkListVariant-Objekt verweist ein DocDef-Objekt auf ein oder mehrere WorkItems.

Syntax

DocDef Name {
  FormFileName Name
  Copy No
  Printer Name
  DocRef Name
  Archive Value
  Print Value
  SourceCopy Value
  WorkItems Value
  WorkListVariant Name
  DocRefFields List
  JoinKey Value
  SortKey Value
  Calc {
    ...
  }
  Grouping {
    ...
  }
  AutoPositioning Value
  Positioning {
    ...
  }
  Recognition bzw. **Rec** {
    ...
  }
  Qualifier Value
}

Erklärung

FormFileName

FieldOrValue
FormFileName setzt für die WorkItems einen Standardfile, welcher die physikalischen Seiten und SubForms definiert. Der FormFileName kann im WorkItem überschrieben werden.

Copy No

FieldOrValue
Der Copy-Faktor im DocDef setzt unabhängig vom Copy-Faktor der WorkItems die Anzahl der zu druckenden Exemplare fest.

Anmerkung zur Kuvertierstraßensteuerung

Die Kuvertierstraßensteuerung ist duplexfähig. Es werden die tatsächlich benötigten Seitenzahlen verwendet. Die Kopieangaben aus dem WorkItem werden ausgewertet. Das Zusammenfassen unterschiedlicher Kanal-1-WorkItems bzw. unterschiedlicher Kanal-2-WorkItems ist möglich. Die Anzahl der Kopien von DocDef und WorkItem werden multipliziert.

Printer Name

FieldOrValue
Mit der Angabe des Printers kann bereits im DocDef-Objekt der Drucker für alle WorkItems gesetzt werden. Der hier gesetzte Drucker kann aber im WorkItem überschrieben werden.

DocRef Name

FieldOrValue
DocRef verweist im Allgemeinen auf ein Feld des Input-Dokuments, welches eine eindeutige Referenznummer des Dokuments beinhaltet. Diese Referenznummer kann die Rechnungs- oder Belegnummer sein. Verwendet wird diese Nummer zum Logging.

```
Beispiel:

DocRef @STRCV_DOCNO
```

Archive Value

FieldOrValue
Archive ist ein Schalter mit den Werten 0 und 1. Steht Archive auf 1, so erfolgt die Auswertung des Archive-Parameters der zum Dokument gehörenden WorkItems. Andernfalls wird das Archivieren unterbunden (unabhängig vom Schalter Print).

```
Beispiel:

Archive @GETSUBSTITUTE(“ArchivTab“, JF_REPRINT)
```

Print Value

FieldOrValue
Print ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Steht Print auf 1, so erfolgt die Druckausgabe. Andernfalls wird das Drucken unterbunden (unabhängig vom Schalter Archiv).

```
Beispiel:

Print @GETSUBSTITUTE(“PrintTab“, JF_REPRINT)
```

SourceCopy Value

FieldOrValue
SourceCopy ist ein Schalter mit den Werten 1 und 0. Steht SourceCopy auf 1, werden die Dokumente nach dem Passieren des Input-Interfaces im internen OMS-ReportWriter-Format in eine Datei gespeichert. Andernfalls gibt es keine Copy des internen Dokuments. Das SourceCopyFile dient vor allem der besseren Lesbarkeit des Input-Dokuments und der Kontrolle des Input-Interfaces.

WorkItems List

FieldOrValue
WorkItems ist eine kommaseparierte Liste mit Namen von WorkItems, die zur Ausgabe abgearbeitet werden sollen. Ist das Schlüsselwort WorkItems angegeben, so wird die Angabe einer WorkListVariant ignoriert. Das Schlüsselwort kann mehrfach vorkommen. Die Werte der jeweiligen Schlüsselwörter werden aneinander gehängt.

WorkListVariant Name

Der Parameter WorkListVariant verweist auf ein WorkListVariant-Objekt. Darin wird ermittelt, welche WorkItems zu diesem Dokument ausgeführt werden sollen. Ist das Schlüsselwort WorkItems angegeben, so wird die Angabe einer WorkListVariant ignoriert.

DocRefFields List

DocRefFields ist eine kommaseparierte Liste von Variablen, die in die DocRef-Liste des Dokuments kopiert werden. Die DocRef-Liste wird in der weiteren Verarbeitung noch durch Standard-DocRef-Fields erweitert und kann dann über die im Schlüsselwort OutFileDocRefFields des CommonSettings-Basis-Objektes ausgegeben werden.

Als Offset für die Seitenzähler C1FROMSHEET, C1TOSHEET bzw. C2FROMSHEET, C2TOSHEET kann die Variable C1PAGEOFFSET bzw. C2PAGEOFFSET mitgegeben werden. Das Offset sollte der Seitengröße der Headerpage entsprechen.

JoinKey Value

FieldOrValue
Alle Dokumente, die den gleichen JoinKey besitzen und den gleichen Qualifier miteinander teilen, werden noch vor der DV-Freimachung und den anderen Dokument-Logistischen-Verfahren zu einem Dokument zusammengefasst. JoinKey entspricht damit der Aufgabenstellung der Porto-Optimierung, bei der möglichst alle Sendungen an einen Empfänger zu einer Sendung zusammengefasst werden. Der JoinKey sollte die Adresse des Sendungsempfängers oder dessen Kundennummer sein.

SortKey Value

FieldOrValue
Alle Dokumente, die nicht DV-freizumachen sind, werden vor der Ausgabe anhand des SortKeys sortiert. Der Sortiertyp ist dabei immer alphanumerisch aufsteigend.

Calc

Auch im DocDef gibt es das Objekt Calc. Die Syntax bleibt unverändert, die Verwendung wird gesondert erläutert (siehe Interne Objekte).


Syntax

```
DocDef Name {
  Calc {
    ...
  }
}
```

AutoPositioning

FieldOrValue
AutoPositioning ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standard ist. Wenn AutoPositioning eingeschaltet ist, dann ist jede andere Positioning-Angabe ohne Wirkung. Ist AutoPositioning eingeschaltet, so erfolgt die Zuordnung der Felder der globalen Tabelle zu Positionen über die aus der XDP-Datei gelesenen Informationen. AutoPositioning arbeitet nur, wenn auf DocDef-Ebenen ein FormFileName angegeben wurde. Aus der angegebenen XDP-Datei werden alle SubForms gelesen und die Felder der globalen Tabelle mit den Feldern der SubForms verglichen. Passt ein Feld zu einer SubForm, so wird eine neue Position im OMS-ReportWriter eröffnet und das Feld dort hinein bewegt. Sind die folgenden Felder auch in der SubForm enthalten, so werden diese ebenfalls dorthin bewegt. Feldwiederholungen führen auch zur Wiederholung der Position. Passt ein Feld nicht in das zuletzt gefundene SubForm, so beginnt die Suche nach einem passenden SubForm erneut.

Positioning

Positioning wandelt Kopffelder in Positionen um. Es wird eine Liste erzeugt, in der die Schlüsselfelder und der Positionsname definiert sind. Schlüsselfelder sind die Namen der Felder, die eine Position erzeugen. Alle nachfolgenden Felder eines Schlüsselfeldes werden mit in die neue Position übernommen. Trifft der Algorithmus auf ein weiteres oder ein gleich benanntes Schlüsselfeld, so wird eine neue Position eröffnet.


Syntax

```
Positioning {
  FieldName1 Position1
  FieldName2 Position2
  ...
}
```

Grouping

Das Sub-Objekt Grouping realisiert das Gruppieren einzelner Positionen zu Groups. Diese Funktion ist dann sinnvoll, wenn die Eingangsdaten nicht im Sinne logischer Gruppierungen sortiert sind, sondern in Druck-Positionen. Zur Sortierung oder WorkItem-Erkennung werden die zu kaufmännischen Positionen zusammengefassten Groups jedoch benötigt.

Grouping beinhaltet ein oder mehrere Group-Objekte. Jedes Group-Objekt ist eine konkrete Anweisung dafür, wie nachfolgende Positionen zu einer Group zusammengefasst werden. Ein Group-Objekt definiert für eine solche Klammer die Anfangs- und Endpositionen. Für die StartPosition kann optional in Klammern noch ein Feld und dessen Wert angegeben werden. Beim Grouping wird die Position nur als StartPosition erkannt, wenn der Name der Position übereinstimmt und das Feld in der Position den vorgegebenen Inhalt hat.

 Syntax

```
Grouping {
  ...
  Group GROUPNAME {
    StartPos          POSITIONNAME[(FIELD=VALIUE)][,POSITIONNAME[(FIELD=VALIUE)][...]]
    [StopBeforePos  POSITIONNAME[,POSITIONNAME[...]]]
    [StopAfterPos     POSITIONNAME[,POSITIONNAME[...]]]
  }
  ...
}
```

Recognition bzw. **Rec**

Die Recognition steuert, ob ein Objekt ausgeführt wird oder nicht. Die Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben.

Qualifier Value

FieldOrValue
Qualifier ist ein Textstring, der als FieldOrValue ausgelegt ist. Mit Hilfe der Qualifier und des Schalters SeparateCentralQualifiers können zentrale Dokumente nach einem Qualifier separiert werden. Es entstehen für jeden vorhandenen Qualifier separate Ausgabefiles.

WorkListVariant

Verwendung

Ein WorkListVariant-Objekt ist ein Recognition-Objekt. Über die WorkListVariant wird festgelegt, welche WorkItems (ausgehende Dokumente) zu einem DocDef (eingehendes Dokument) erzeugt werden sollen. Das WorkListVariant-Objekt ist eine Liste, die zur Bestimmung auszugebender WorkItems verwendet wird. In ihr werden Verweise auf WorkItem-Objekte und dazugehörige Erkennungskriterien definiert. Weiterhin kann zu einer WorkItem-Liste noch angegeben werden, in welcher TCI diese nachzuladenden Objekte zu finden sind. Die Liste der WorkLists wird von oben nach unten abgearbeitet. Dabei wird geprüft, ob die Erkennung einer WorkList zu den aktuellen Daten passt. Die erste WorkList, bei der die Erkennung den Wert wahr zurückliefert, ist die WorkList, die abgearbeitet wird. Die Angabe der zu prüfenden WorkList kann in zwei Syntax-Varianten angegeben werden. In der verkürzten Syntax befindet sich die komplette Definition auf einer Zeile.

Syntax

```
WorkListVariant Name {
  ...
  Field=Value [Field!Value [...]] (WorkItem[,WorkItem[...]])
  ...
}
```

Eine Liste besteht aus einer oder mehreren Zeilen, in der eine Liste von Feld-Inhalts-Paaren und die dazugehörigen WorkItems angegeben werden. Als Operatoren stehen ein Gleich-Operator (=) und ein Ungleich-Operator (!) zur Wahl. Die einzelnen Vergleichskriterien sind dabei Und-verknüpft.

In der erweiterten Syntax befinden sich die Angaben zu den WorkLists in einem strukturierten Objekt.

Syntax

```
WorkListVariant Name {
  ...
  WorkList {
    Recognition bzw. Rec {
      ...
    }
```

```
    WorkItems LISTOFWORKITEMS
    TCIFileName TCIFILENAME
  }
  ...
}
```

Recognition bzw. **Rec**

Die Recognition steuert, ob ein Objekt ausgeführt wird oder nicht. Die Funktionsweise ist im Kapitel „Interne Objekte" beschrieben. Die Angabe ist optional.

WorkItems List

WorkItems ist eine kommaseparierte Liste mit Namen von WorkItems, die zur Ausgabe abgearbeitet werden sollen.

TCIFileName

Der TCIFileName definiert den File-Namen einer nachzuladenden TCI, in der sich das entsprechende DocDef befindet. Die Angabe ist optional.

Beispiel:

```
WorkListVariant Invoice_Name {
 WorkList {
   Rec {
      IsEqualS(StxRDI_TDForm,"zrd1_RVInvoice01")
      IsEqualS(client,"473")
   }
   WorkItems Invoice473,Delivery_Note473
   TCIFileName client473.tci
 }
 StxRDI_TDForm=zrd1_RVInvoice01 (Invoice,Delivery_Note)
}
```

Die verkürzte und die erweiterte Syntax lassen sich im WorkListVariant-Objekt miteinander mischen. Der Vorteil der vekürzten Syntax ist die kürzere Schreibweise mit dem Nachteil, dass nur sehr einfache Vergleichsoperatoren (=) und (!) zugelassen sind und das Nachladen einer TCI nicht möglich ist. In der erweiterten Syntax ist im Recognition-Objekt jeder Calc-Befehl zulässig, der auf Dokument-Ebene ausgeführt werden kann.

WorkItem

Verwendung

Ein WorkItem-Objekt ist eine konkrete Realisierung eines Dokuments zum Druck oder Archiv. Das WorkItem generiert ein Formular aus dem Input-Dokument.

Syntax

WorkItem Name {
  FormFileName Name
  PDFImportType Value
  PDFImportDocuments Value
  FirstPage No
  NextPage No
  DesignsFirst Value
  DesignsNext Value
  ArchiveDesignsFirst Value
  ArchiveDesignsNext Value
  PaperTypeFirst Value
  PaperTypeNext Value
  PageDescriptorFirst Value
  PageDescriptorNext Value
  IntrayFirst Value
  IntrayNext Value
  OuttrayFirst Value
  OuttrayNext Value
  FinishingFirst Value
  FinishingNext Value
  BackPageFirst Value
  BackPageNext Value
  RotateToFirst Value
  RotateToNext Value
  GutterFirst Value
  GutterNext Value
  Copy No
  CopyText Value
  Printer Value

Print Value
Archive Value
ArchiveText Value
ArcRefFields Value
PageReverser Value
JoinBefore Value
Calc {
...
}
EnvelopeSortSystem {
...
}
Positions Value
BackPageDesignsFirst Value
BackPageDesignsNext Value
DuplexFirst Value
DuplexNext Value
Recognition bzw. **Rec** {
...
}
Central Value
HeaderWorkItem Name
TrailerWorkItem Name
IndicantWorkItem Name
HeaderFields Value
TrailerFields Value
IndicantFields Value
VOLFields Value
WIFields Value
PDFFields Value
PDFProfile Value
ContinuationTextPage Value
ContinuationTextBackPage Value
Layout Value
LayoutElement Value
DataBind Value
EmbeddedFiles Value
}

FormFileName Name

FieldOrValue
Der FormFileName bestimmt den File, in dem die physikalischen Seiten und SubForms definiert sind. Fehlt die Angabe, so wird auf die Eingabe vom DocDef zurückgegriffen.

PDFImportDocuments Value

FieldOrValue
Ist eine semikolonseparierte Liste von PDF-FileNamen, die in der angegebenen Reihenfolge mit dem WorkItem verbunden werden sollen. Es ist auch möglich nur einzelne Seiten zu importieren. Dazu muss nach dem FileNamen das Pipe-Symbol („|“) gefolgt von der Seitennummer der zu importierenden Seite angegeben werden. Auch XDP- und Images-Dateien lassen sich auf diese Weise importieren. In diesem Fall steht hinter dem Pipe-Symbol der Seitenname des XDPs oder die Bildnummer in der Image-Datei. In welchem Modus die Seiten mit WorkItem verbunden werden, regelt der Parameter PDFImportType.

PDFImportType Value

FieldOrValue
PDFImportType ist ein Schalter mit den Werten von 0 bis 4, wobei 0 der Standardwert ist und bestimmt, in welcher Form die unter PDFImportDocuments angegebenen PDF-Files mit dem WorkItem zusammengeführt werden:

0 *FirstOverlayAll*
als Overlay beginnend auf der ersten Seite des WorkItems, alle PDF-Seiten

1 *NextOverlayAll*
als Overlay beginnend auf der zweiten Seite des WorkItems, alle PDF-Seiten

2 *Add*
als Anhang nach den positionsbedingten Seiten des WorkItems, alle PDF-Seiten

3 *FirstOverlayDoc*
als Overlay beginnend auf der ersten Seite des WorkItems, nur soviel Seiten, wie das WorkItem positionsbedingt besitzt

4 *NextOverlayDoc*
als Overlay beginnend auf der zweiten Seite des WorkItems, nur soviel Seiten, wie das WorkItem positionsbedingt besitzt. Der FormFileName bestimmt den File, in dem die physikalischen Seiten und SubForms definiert sind. Fehlt die Angabe, so wird auf die Eingabe vom DocDef zurückgegriffen.

FirstPage Name

FieldOrValue
Gibt den Namen der Grundseite einer vom WorkItem generierten FirstPage wieder.

NextPage Name

FieldOrValue
Name der Grundseite einer vom WorkItem generierten NextPage. Für den Duplex-Betrieb können auch zwei unterschiedliche Grundseiten angegeben werden. Eine, wenn die NextPage eine Vorderseite ist, und eine andere, wenn die NextPage eine Rückseite ist.

Syntax

NextPage Name oder
NextPage NameVorderseite#NameRückseite

DesignsFirst Value
DesignsNext Value

FieldOrValue
DesignsFirst und DesignsNext definieren eine Folge an Design-Subformularen und ihren Zielpositionen, so dass die Designs stufenweise auf die Grundseite gebracht werden. Siehe dazu den Abschnitt Designs.

```
Beispiel:

DesignsNext ME_COMPANY_DESIGN,SubForm10;[30mm,120mm],SubForm5+
```

ArchiveDesignsFirst Value
ArchiveDesignsNext Value

FieldOrValue

ArchiveDesignsFirst und ArchiveDesignsNext definieren eine Folge von Design-Subformularen und deren Zielpositionen. Im Gegensatz zu DesignFirst und DesignNext werden diese aber nur auf den Archivkopien angebracht. Siehe dazu den Abschnitt Designs.

PageDescriptorFirst Value
PageDescriptorNext Value

FieldOrValue
Mit den PageDescriptoren kann je Seite eine globale Variable PageDescriptor definiert werden, die mit im Formular ausgegeben wird.

 Syntax

PageDescriptorFirst Value
PageDescriptorNext Value

```
Beispiel:

PageDescriptorFirst test print
PageDescriptorNext  test print
```

PaperTypeFirst Value
PaperTypeNext Value

FieldOrValue
Der Papiertyp beschreibt die Art des Papiers, welches in den Papierschacht eingelegt werden muss, der durch das Intray-Kommando angesprochen wird. Aus den Werten von PaperType und Intray wird eine Papierbelegungstabelle erstellt, anhand derer sich widersprechende Papierbelegungen ausgeschlossen werden können.

 Syntax

PaperTypeFirst Value
PaperTypeNext Value

```
Beispiel:

PaperTypeFirst letterhead
PaperTypeNext payment form
```

IntrayFirst Value
IntrayNext Value

FieldOrValue
IntrayFirst und IntrayNext definieren den Papiereinzugsschacht für die FirstPage und die NextPage.


Syntax

IntrayFirst Value
IntrayNext Value

```
Beispiel:

IntrayFirst lower_tray
IntrayNext  upper_tray
```

OuttrayFirst Value
OuttrayNext Value

FieldOrValue
OuttrayFirst und OuttrayNext definieren den Papierausgabeschacht für die FirstPage und die NextPage.


Syntax

OuttrayFirst Value
OuttrayNext Value

```
Beispiel:

OuttrayFirst backside_bin
OuttrayNext  frontside_bin
```

FinishingFirst Value
FinishingNext Value

FieldOrValue
FinishingFirst und FinishingNext definieren Nachverarbeitungsoptionen, das sogenannte Paperfinishing wie Stapeln (Heften) oder Jogging (versetzte Ablage), für die FirstPage und die NextPage.

BackPageFirst Value
BackPageNext Value

FieldOrValue
Mit BackPageFirst und BackPageNext wird die Seite im Design angegeben, die auf die Rückseite der FirstPage bzw. der NextPage gedruckt werden soll. Ist kein Seitenname angegeben, so erfolgt der Ausdruck Simplex (ohne Rückseite).

Syntax

BackPageFirst FormFilePageName
BackPageNext FormFilePageName

```
Beispiel:

BackPageFirst Backpage1
```

RotateToFirst Value
RotateToNext Value

FieldOrValue
RotateToFirst und RotateToNext bestimmen ob die Seite in der Ausgabe gedreht werden soll.
Folgende Werte sind möglich:

NONE	keine Drehung (Standardwert)
PORTRAIT	querformatige Seiten werden gegen die Uhr gedreht, hochformatige Seiten werden nicht gedreht
LANDSCAPE	hochformatige Seiten werden gegen die Uhr gedreht, querformatige Seiten werden nicht gedreht
LEFT	Drehungen gegen die Uhr
RIGHT	Drehungen mit der Uhr
HEADSTAND	Seiten werden auf den Kopf gestellt

RotateToFirst und RotateToNext haben ein Pendant in dem PDFProfile der Ini-Datei. Die Einstellungen im WorkItem haben allerdings Vorrang.

GutterFirst Value
GutterNext Value

FieldOrValue
GutterFirst und GutterNext bestimmen den Bundsteg als Längenangabe für Rückseiten die nicht statisch sind. Damit kann die Rückseite eines Blattes um die angegebene Längeneinheit orthogonal zur Bindekante versetzt werden, wobei nur positive Werte eine Auswirkung haben. Der Standardwert 0.0mm.
Es wird die gesamte Seite verschoben. Nur Designs, die sich den Positionsbezeichnung ROOT beziehen bleiben an der Originalstelle stehen und werden nicht versetzt.

Copy No

FieldOrValue
Mit Copy wird die Anzahl der auszugebenden Exemplare des WorkItems definiert. Es werden dabei mehrere Kopien des WorkItems in den Datenstrom generiert. Mit CopyText kann auf jede Kopie noch ein Text aufgebracht werden, der Organisationsmerkmale enthält.

CopyText Value

FieldOrValue
Für jede auszugebende Kopie, die über Copy bestimmt wird, kann ein Organisationsmerkmal mitgegeben werden. CopyText unterstützt dabei zwei Methoden. Ist der Wert von Copy auf DocDef-Ebene größer als 1, so wird der CopyText auf Basis der DocDef-Kopien erzeugt.

1. CopyText wird ein kommaseparierter String übergeben. Jeder Token des Strings entspricht einem Organisationstext für eine Kopie.

```
Beispiel:

CopyText Original, Kopie 1, Kopie 2, Kopie Buchhaltung
```

2. CopyText wird ein String mit einer Variablen und ohne Komma übergeben. Enthält der CopyText ein %P, so erfolgt die Ausgabe des CopyTextes ab der ersten Kopie. Enthält CopyText kein %P, so wird CopyText erst ab der zweiten Kopie ausgegeben. Enthält der CopyText den Platzhalter %C oder %P so werden diese beim Generieren ausgetauscht:

   %C Nummer der Kopie-1
   %P Nummer der Kopie

```
Beispiel:

CopyText Kopie %C
```

Printer Value

FieldOrValue
Printer definiert den Ausgabedruck für das WorkItem. Ist bereits ein Drucker im DocDef , so wird dieser für dieses WorkItem überschrieben.

Print Value

FieldOrValue
Print ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Steht Print auf 1, so erfolgt die Druckausgabe. Andernfalls wird das Drucken unterbunden (unabhängig vom Schalter Archiv).

Archive Value

FieldOrValue
Archive ist ein Schalter mit den Werten 1 und 0 und bestimmt, ob dieses WorkItem auch zu archivieren ist. Voraussetzung für das Archivieren ist, dass im DocDef das Archivieren für dieses Dokument erlaubt wurde, indem Archive auf eins gesetzt wurde. Wenn ein WorkItem archiviert wird, so enthält das Archivdokument weniger Angaben als das zu druckende Dokument. Angaben zur Portoklassenberechnung, OMR-Steuerung und einige Seitenzähler werden nicht archiviert.

```
Beispiel:

Archive @GETSUBSTITUTE(“ArchivTab“, JF_REPRINT)
```

ArchiveText Value

FieldOrValue
ArchiveText bewirkt, dass der angegebene Text als globales Feld mit dem Namen ArchiveText in einem zu archivierenden Dokument erscheint. Die dazugehörige Druckausgabe enthält dieses Feld auch, aber der Inhalt des Feldes bleibt leer.

```
Beispiel:

ArchiveText Archive output
```

ArcRefFields Value

ArcRefFields definieren in einem kommaseparierten String Kopffelder, die zum Archivieren als Referenzvariablen mitgegeben werden. Die erste Variable sollte eine eindeutige Belegnummer sein, da diese im Log des Archivers wiedergefunden werden kann. Jedes WorkItem, das eine eigene ArcRefFields-Liste besitzt, eröffnet im Ausgabedatenstrom für den Archiver ein eigenständiges Archiv-Dokument. Sollen mehrere WorkItems zu einem Archiv-Dokument zusammengefasst werden, so darf nur das erste WorkItem eine eigene ArcRefFields-Liste besitzen.

```
Beispiel:

ArcRefFields JFA_DOCNUM,JFA_DOCDAT,JFA_DOCTYPE
```

Einige Archivsysteme arbeiten auf Basis eines Volltextindexes. Um diese Funktionalität zu unter-stützen, gibt es ein spezielles Konstrukt, das alle Nettodateninhalte in einem Feld zusammenfasst.

```
Beispiel:
```

```
ArcRefFields JFA_DOCNUM,JFA_DOCDAT,JFA_DOCTYPE,ANF=JFA_COMPLETE

Zusätzlich zu den normalen Indexfeldern wird das Feld JFA_COMPLETE
in den Index generiert. JFA_COMPLETE enthält die Nettodaten des
gesamten Dokuments.
```

PageReverser Value

FieldOrValue
Einige Kuvertierstraßen benötigen das gedruckte Dokument in umgekehrter Reihenfolge. PageReverser ist ein Schalter mit den Werten 1 und 0, wobei 0 der Standardwert ist. Ist der PageReserver aktiviert, wird die letzte Seite eines Dokuments zuerst gedruckt.

Calc

Das Calc-Objekt dient zur Berechnung von Feldern. Die Funktionsweise ist im Kapitel „Interne Objekte" beschrieben.

EnvelopeSortSystem

Das EnvelopeSortSystem beschreibt alle Werte, die im Zusammenhang mit einem Kuvertiersystem der Portoklassenberechnung bzw. der DV-Freimachung stehen. Die Funktionsweise ist im Kapitel „Interne Objekte" beschrieben.

Positions

Positions definiert die Reihenfolge und die Auswahl von TCI-Positionen, die im dynamischen Bereich des WorkItems angedruckt werden. Bei der Angabe der Positionen werden folgende Einstellungen festgelegt:

1. Kommt die Position aus dem Datenstrom oder ist diese fix anzudrucken?
2. Aus welcher Tabelle kommen die Daten der Position?
3. Wird die Reihenfolge der Positionen aus dem Datenstrom bestimmt oder ist diese fest vorgegeben?
4. Müssen die Positionen sortiert werden?

Syntax

Positions Section[,Section[...]]

Die Sektionen werden in der Reihenfolge von links nach rechts abgearbeitet. Sektionen sind bestimmte Bereiche von Positionen, die Sortierung, Datentabelle und die Entscheidung über fix oder dynamisch miteinander teilen. Eine Sektion kann sein:

1. eine fixe Position
2. die dynamische Auswahl von Positionen aus der Positions-Tabelle
3. die dynamische Auswahl von Positionen aus der Group-Tabelle

Die fixe Position

Eine fixe Position ruft unabhängig vom Datenstrom eine TCI-Position auf. Die gerufene Position hat aus diesem Grund auch keine lokalen Daten, die im SubForm-Objekt der Position mit angedruckt werden können.

 Syntax

PosName

Die dynamischen Positionen aus der Positions-Tabelle

Diese Sektion besteht aus der Aufzählung einer oder mehrerer TCI-Positionen, die in der Positions-Tabelle gesucht werden. Dabei werden alle Positionen der Positionstabelle, die der Aufzählung entsprechen, in eine temporäre Liste überführt. Diese Liste kann sortiert und unsortiert ausgegeben werden. Die Werte des Sortierfeldes können numerisch oder alphanumerisch sein. Treten numerische und alphanumerische Werte gemischt auf, so werden die numerischen Werte vor die alphanumerischen Werte sortiert.

 Syntax

PosName[|PosName [...]]([SortField])

Es werden alle Positionen TERMS, ITEM_LINE_1, ITEM_LINE_3F und ITEM_LINE_3F_PRICE gesucht und in der Reihenfolge gedruckt, in der diese in der Positionstabelle standen.

```
Beispiel:

TERMS|ITEM_LINE_1|ITEM_LINE_3F|ITEM_LINE_3F_PRICE()
```

Es werden alle Positionen ITEM gesucht und nach dem in den ITEM-Positionen vorhandenen Feld ITEM_NO aufsteigend sortiert und gedruckt.

Beispiel:

```
ITEM(ITEM_NO)
```

Als Alternative zur genauen Angabe der Positionen kann für dynamische Positionen auch ein Wildcard angegeben werden. Es werden alle Positionen zum Druck angestoßen, die als dynamische Position verfügbar sind.

Beispiel:

```
POSITIONS *()
```

Im nachfolgenden Beispiel werden eine Kopfposition HEADERPOS, alle dynamischen Positionen sowie eine Schlussposition TRAILERPOS angedruckt.

Beispiel:

```
POSITIONS HEADERPOS, *(), TRAILERPOS
```

Die dynamischen Positionen aus einer Group-Tabelle

Diese Sektion besteht aus der Aufzählung einer oder mehrerer TCI-Positionen, die in der Group-Tabelle gesucht werden. Dabei werden alle Positionen der Positionstabelle, die der Aufzählung entsprechen, in eine temporäre Liste überführt. Diese Liste kann sortiert und unsortiert ausgegeben werden. Die Werte des Sortierfeldes können numerisch oder alphanumerisch sein. Treten numerische und alphanumerische Werte gemischt auf, so werden die numerischen Werte vor die alphanumerischen Werte sortiert.

Syntax

GroupName([SortField])[PosName|…]

Es werden alle Groups mit dem Namen DELIV_GROUP in der Reihenfolge des Vorkommens in der Group-Tabelle angezogen und deren Positionen ITEM_LINE und ITEM_LINE_1 gedruckt.

Beispiel:

```
DELIV_GROUP()[ITEM_LINE|ITEM_LINE_1]
```

Es werden alle Groups mit dem Namen DELIV_GROUP nach dem Feld POSSORT sortiert und deren Positionen ITEM_LINE und ITEM_LINE_1 gedruckt.

Beispiel:

```
DELIV_GROUP(POSSORT)[ITEM_LINE|ITEM_LINE_1]
```

Als Alternative zur genauen Angabe der Positionen kann für dynamische Positionen und für Group-Namen auch ein Wildcard angegeben werden. Es werden alle Positionen zum Druck angestoßen, die als dynamische Position verfügbar sind.

Zieht alle Groups mit dem Namen BILL an und druckt daraus alle Positionen.

```
Beispiel:

BILL()[*]
```

Zieht alle Groups an und druckt daraus alle Positions.

```
Beispiel:

*()[*]
```

BackPageDesignsFirst Value
BackPageDesignsNext Value

FieldOrValue
BackPageDesignsFirst und BackPageDesignsNext definieren eine Folge von Design-Subformularen und Positionen für die Rückseite. Siehe dazu den Abschnitt Designs.

DuplexFirst Value
DuplexNext Value

FieldOrValue
DuplexFirst bezieht sich hierbei auf die erste Seite eines WorkItems und DuplexNext auf alle Folgeseiten. Bei Seiten, die eine Rückseite haben, wird die Duplex-Eigenschaft <1 ignoriert und stattdessen die Rückseite mit der Duplex-Einstellung 1 gezogen. DuplexFirst und DuplexNext kennen jeweils vier Zustände.

Syntax

DuplexFirst n
-1 = Duplex aus
0 = keine Angaben (Default)
1 = Duplex LongEdgeBinding
2 = Duplex ShortEdgeBinding

DuplexNext n
-1 = Duplex aus
0 = keine Angaben (Default)
1 = Duplex LongEdgeBinding
2 = Duplex ShortEdgeBinding

Recognition bzw. **Rec**

Die Recognition steuert, ob ein Objekt ausgeführt wird oder nicht. Die Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben.

Central

FieldOrValue
Central ist ein Schalter mit den Werten 0 und 1 und als FieldOrValue ausgelegt. Mit ihm lässt sich bei der Ansteuerung für mehrere zentrale Drucker zwangsweise die zentrale Verarbeitung festlegen. Bei der Nutzung dieses Schalters muss der Drucker für zentrale Drucke nicht unbedingt leer sein. Ist er leer, so erhält der erzeugte Job als Druckernamen den BaseNamen des Ausgabefiles, andernfalls wird der Druckername wie bei dezentralen Druckern gesetzt. Damit ist diese Erweiterung aufwärtskompatibel zu Konfigurationen aus älteren Installationen. Ist kein Drucker gesetzt, so wird Central für das WorkItem automatisch eingeschaltet.

```
Beispiel:

WORKITEM payment form {
  ...
  Central 1
  ...
}
```

HeaderWorkItem Name

Ist dieses WorkItem das erste WorkItem eines Druckstapels, so wird das hier definierte HeaderWorkItem im Sinne einer HeaderPage vor den Druckstapel gestellt. Die mit HeaderFields definierten Felder werden dem HeaderWorkItem als globale Felder mitgegeben ebenso wie alle Statusinformationen des Druckjobs.

TrailerWorkItem Name

Ist dieses WorkItem das erste WorkItem eines Druckstapels, so wird das hier definierte TrailerWorkItem im Sinne einer TrailerPage dem Druckstapel hinten angestellt. Die mit TrailerFields definierten Felder werden dem TrailerWorkItem als globale Felder mitgegeben ebenso wie alle Statusinformationen des Druckjobs.

IndicantWorkItem Name

Ist dieses WorkItem das erste WorkItem eines Druckstapels, so wird das hier definierte IndicantWorkItem als separater Druckjob zur Beschreibung des eigentlichen Druckjobs generiert. Die mit IndicantFields definierten Felder werden dem IndicantWorkItem als globale Felder mitgegeben. Ein IndicantWorkItem muss im Gegensatz zu Header- oder TrailerWorkItems einen Drucker definieren. Indicant-Druckjobs kündigen einen Druckjob an oder stellen eine Art Auftragsbegleitzettel oder Produktionsabrechnungszettel dar, da alle Statusinformationen des eigentlichen Druckjobs als globale Felder mit hinein generiert werden.

HeaderFields Value

HeaderFields ist eine Aufzählung von Feldnamen. Die hier angegebenen Felder werden im aktuellen Dokument gesucht und in das HeaderWorkItem transferiert.

Syntax

HeaderFields FieldName[,FieldName[..]]

TrailerFields Value

TrailerFields ist eine Aufzählung von Feldnamen. Die hier angegebenen Felder werden im aktuellen Dokument gesucht und in das TrailerWorkItem transferiert.


Syntax

TrailerFields FieldName[,FieldName[..]]

IndicantFields Value

IndicantFields ist eine Aufzählung von Feldnamen. Die hier angegebenen Felder werden im aktuellen Dokument gesucht und in das IndicantWorkItem transferiert.


Syntax

IndicantFields FieldName[,FieldName[..]]

VolFields Value

VolFields ist eine Aufzählung von Feldnamen. Die hier angegebenen Felder werden im aktuellen Dokument gesucht und in die VOL-Datei transferiert. Dort tragen die Variablen dann noch das Präfix RW_Doc_.


Syntax

VolFields FieldName[,FieldName[..]]

WIFields Value

WIFields oder WorkItemFields ist eine Aufzählung von Feldnamen. Die hier angegebenen Felder werden im aktuellen Dokument gesucht und als Beschreibungsvariablen auf Seitenebene in das zu erzeugende PDF eingefügt.

PDFFields Value

PDFFields ist eine Aufzählung von Feldnamen. Die hier angegebenen Felder werden im aktuellen Dokument gesucht und an das Modul zur PDF-Erzeugung weitergeleitet. Die so gesammelten Felder können im Inhaltsverzeichnis des PDFs verwendet werden. Ein anderer Einsatzzweck ist die Steuerung des PDFProfiles über diese Variablen.

Syntax

PDFFields FieldName[,FieldName[..]]

PDFProfile Value

FieldOrValue
PDFProfile ist eine Namensreferenz auf ein in der reportw.ini unter PDF/Profiles definiertes Eröffnungsprofile. Mit einem solchen Profile können die Grundeigenschaften wie Rechte, Titel, Autor usw. des anzulegenden PDFs gesteuert werden. Mit PDFFields ist es möglich, zusätzlich dazu Variablen aus dem Dokument weiterzugeben, die in den gewählten Profilen über FieldOrValue verwendet werden können. Die Archiv-Ausgabe des WorkItems erfolgt unabhängig von dem hier gesetzten Profile über das Profile Archive.

ContinuationTextPage Value
ContinuationTextBackPage Value

FieldOrValue
ContinuationTextPage ist ein Text, der auf Seiten ausgegeben wird, die nicht die letzte Seite sind, und dient zur Spezifizierung, auf welcher Seite das Dokument fortgesetzt wird. Der Textstring kann dabei einen Platzhalter %1 beinhalten, der durch die richtige Seitenzahl ersetzt wird.

ContinuationTextBackPage funktioniert ebenso, nur dass keine Seitenzahl generiert wird, sondern nur darauf verwiesen wird, dass es auf der Rückseite weitergeht.
Je nachdem, ob es auf der Rückseite oder einer neuen Seite weitergeht, wird der Text ContinuationTextBackPage oder ContinuationTextPage in das globale Feld ContinuationText ausgegeben. Die Ausgabe erfolgt nicht im Archiv-Lauf.

```
Beispiel:

 WORKITEM Invoice {
  ...
  CONTINUATIONTEXTPAGE       Weiter auf Seite %1
  CONTINUATIONTEXTBACKPAGE   Weiter auf der Rückseite
  ...
  }
```

LayoutElement Value

FieldOrValue
WorkItems, die das Schlüsselwort LayoutElement definieren, sind nicht zur direkten Ausgabe bestimmt. Vielmehr werden die Seiten solcher WorkItems in einen Puffer hinein generiert, der den Namen des LayoutElementes trägt. Andere WorkItems im gleichen DocDef, die mit dem Schlüsselwort Layout ein bestimmtes Layout anfordern, ziehen dann die Seiten aus bestimmten Puffern auf das Blatt. WorkItems mit dem Schlüsselwort LayoutElement können archiviert werden. Gedruckt werden sie nur auf dazugehörigen WorkItems mit dem Schlüsselwort Layout. Der Name des LayoutElementes muss zu dem aufrufenden Layout passen. Da Seiten eines WorkItems mit dem Schlüsselwort LayoutElement logische Seiten sind, werden für diese Seiten alle Einstellungen wie Duplex, Tray und PaperType ignoriert.

Layout Value

FieldOrValue
WorkItem mit dem Schlüsselwort Layout platzieren eine oder mehrere logische Seiten auf den Seiten des aktuellen WorkItems. Ziel der Funktionalität ist es, mehrere logische Seiten (LayoutElement-Seiten) auf einer physikalischen Seite unterzubringen bzw. N-Up-Printing oder logische Seiten nach ihrem LayoutElement-Namen auf Ausgabeseiten zu verteilen. Ein WorkItem mit Layout bringt logische Seiten in den Floating-Bereich (ContentArea) der aktuellen Seite. Dabei werden keine Positionen oder SubForms ausgerechnet oder platziert. Der unter Layout angegebene Name entspricht einem fest vorprogrammierten Layout-Typ. Folgende Layout-Typen sind realisiert:

Layout	Gesuchte LayoutElemente	Arbeitsweise
UP	Spielt keine Rolle	Realisiert N-Up Printing. Die logischen Seiten werden unabhängig von ihrem LayoutElement-Namen der Reihenfolge nach in die ContentAreas hinein platziert. Sind alle ContentAreas der Seite gefüllt, so erfolgt ein Seitenwechsel.
LR	LEFT und RIGHT	Realisiert 2-Up Printing so, dass Layoutelemente mit LEFT in der ersten ContentArea und Layoutelemente RIGHT in der zweiten ContentArea der physikalischen Seite ausgegeben

werden. Die physikalische Seite muss exakt zwei ContentAreas besitzen.

WorkItems mit dem Schlüsselwort Layout können archiviert und gedruckt werden.

DataBind Value

DataBind ist ein optionaler Wert und gibt eine Position im Daten-Baum des Dokuments an, aus der die Positionen des WorkItems erstellt werden sollen. Existiert die angegebene Position nicht, so wird das WorkItem übersprungen.

```
Beispiel:

+---Pos1
|     +---Pos11
|     |     +---Fieldx
|     +---Pos12
|           +---Pos121
|           |     +---Fieldx
|           +---Pos122
|                 +---Fieldx
+---Pos2
      +---Fieldy

DataBind Pos1.Pos12

Es werden die Positionen Pos121 und Pos122 abgearbeitet.
```

EmbeddedFiles Value

FieldOrValue
EmbeddedFiles ist ein Schlüsselwort zum Einbetten von Files in das zu erzeugende Druckdokument. Dabei wird ein oder mehrere Files von einer Festplatte gelesen und auf die erste Seite des zu erzeugenden PDFs eingebettet. Der Empfänger des PDFs kann diesen File dann auspacken und separat verarbeiten. EmbeddedFiles kann mehrfach definiert werden und enthält eine semikolonseparierte Liste von Filereferenzen. Eine Filereferenz ist wiederum eine kommaseparierte Liste: ein Wertepaar. Der erste Eintrag des Wertepaars ist der FileName. Enthält der FileName keinen Pfad, so gilt folgende Suchreihenfolge:

Lokales Verzeichnis, FormsPath, ImagePath

Der zweite Eintrag ist der optionale Kommentar.

```
Beispiel:

c:\hello.xml                              Ein File
c:\hello.xml,Kommentar                    Ein File mit Kommentar:
c:\hello.xml,Kommentar;d:\world.pdf       Mehrere Files
```

JoinBefore Value

FieldOrValue
JoinBefore ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. JoinBefore hängt direkt mit dem DocDef-Schlüsselwort JoinKey zusammen. Beim Joinen entsteht aus mehreren Einzeldokumenten ein Gesamtdokument. Im Standardfall werden die Seiten der Teildokumente immer hinten an das Gesamtdokument angehängt. Wenn das nicht gewünscht ist, dann kann mit JoinBefore in einem WorkItem festgelegt werden, dass die Stelle zum Einfügen weiterer Seiten in das Gesamtdokument vor den Seiten diese WorkItems liegt. JoinBefore wirkt sich nicht nur im ersten Teil-Dokument, dem Master-Dokument, eines GesamtDokuments aus, sondern in jedem anderen außer dem letzten.

Table

Verwendung

Table definiert das Tabellenverhalten für die SubForms in Positionen, die den gleichlautenden Tabelleneintrag haben. Es werden bis zu vier SubForms definiert: TopSubForm, BottomSubForm, PageTopSubForm und PageBottomSubForm. Die Angabe eines jeden SubForms ist optional.
Das Objekt Table hat folgenden Aufbau:

Syntax

```
Table Name {
  Protect Value
  HeaderSubForm Name {
    ...
  }
  TrailerSubForm Name {
    ...
  }
  PageTopSubForm Name {
    ...
  }
  PageBottomSubForm Name {
    ...
  }
  PageBottomJoinedPosition Value
  PageTopJoinedPosition Value
}
```

Protect

Protect ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Achtung: Protect ist kein FieldOrValue! Bei 1 versucht der Algorithmus des Seitenumbruchs, die gesamte Tabelle vor einem Seitenumbruch zu bewahren. Ist die Tabelle zu groß, wird trotzdem umgebrochen. Die gleichzeitige Angabe von PageTopSubForm, PageBottomSubForm und Protect macht also Sinn.

HeaderSubForm, TrailerSubForm

HeaderSubForm leitet die Tabelle ein. TrailerSubForm steht am Ende der Tabelle. Die genaue Syntax der SubForm-Objekte entnehmen Sie bitte der Beschreibung des SubForm-Objektes der Positions.

PageBottomSubForm, PageTopSubForm

PageBottomSubForm steht am Ende der Seite, wenn innerhalb der Tabelle umgebrochen werden muss. Die genaue Syntax der SubForm-Objekte entnehmen Sie bitte der Beschreibung des SubForm-Objektes der Positions.

PageBottomJoinedPosition

PageBottomJoinedPosition ist eine Zahlenangabe von 0 bis n und definiert die Anzahl von SubForms und Subtabellen, die bei einem Seitenumbruch innerhalb einer Tabelle wenigstens am unteren Ende der Seite stehen müssen. Der Standardwert ist 0. HeaderSubForm, TrailerSubForm, PageBottomSubForm spielen bei der Berechnung keine Rolle. Diese Funktion ist auch als „Schusterjungenregel“ bekannt.

PageTopJoinedPosition

PageTopJoinedPosition ist eine Zahlenangabe von 0 bis n und definiert die Anzahl von SubForms und Subtabellen, die bei einem Seitenumbruch innerhalb einer Tabelle wenigstens am oberen Anfang der Seite stehen müssen. Der Standardwert ist 0. HeaderSubForm, TrailerSubForm, PageTopSubForm spielen bei der Berechnung keine Rolle. Diese Funktion ist auch als „Schusterjungenregel“ bekannt.

Positions

Verwendung

Das Positions-Objekt enthält SubForm-Objekte, Calc-Objekte sowie SubPositions-Einträge. In einem Positions-Objekt können mehrere SubForm-Objekte, SubPositions-Einträge und Calc-Objekte enthalten sein, die der Reihenfolge nach von oben nach unten abgearbeitet werden. Der Name einer Position kann um den Namen eines WorkItems erweitert werden. Eine solche Position gilt als WorkItem-abhängig und kann nur von diesem WorkItem benutzt werden. Im Suchalgorithmus der Postionen werden WorkItem-abhängige Positionen vor der gleichnamigen Position ohne WorkItem gefunden. Das Objekt Positions hat folgenden Aufbau:

Syntax

```
Position Name[#WorkItemName] {
  ...
  Recognition bzw. Rec {
    ...
  }
  Calc {
    ...
  }
  SubForm {
    ...
  }
  SubPositions Section[,Section[...]]
  ...
}
```

Erklärung

Recognition bzw. **Rec**

Die Recognition steuert, ob ein Objekt ausgeführt wird oder nicht. Die Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben. Die Angabe ist optional.

Calc

Das Calc-Objekt ist sowohl direkt im Objekt Positions möglich als auch unter SubForm-Objekten. Es dient zur Berechnung von Feldern. Die Funktionsweise von Calc ist im Kapitel „Interne Objekte“ beschrieben.

SubForm

Nach dem Schlüsselwort “SubForm“ steht der Name des Subformulars im Adobe Present/Central.


Syntax

Position Name{
...
SubForm {
Designs Value
Name Value
Fields Value
Height Value
Reserve Value
TopGap Value
PageBreak Value
DoPrint Value
PageTopPosition Value
PageBottomPosition Value
Recognition bzw. Rec Value
Protect Value
WrapTable Value
Table Value
TableControl Value
Label Value
Calc {
...
}
PageCalc {
...
}
Concalc {
...
}
...
}
...
}

Designs Value

FieldOrValue
Designs definieren eine Folge an Design-Subformularen und ihren Zielpositionen, so dass die Designs stufenweise auf die Grundseite gebracht werden. Siehe dazu den Abschnitt Designs.

Name

FieldOrValue
Name des SubForms. Wenn kein Name definiert ist, so wird das SubForm beim Ausdruck ignoriert.

Fields

Unter Fields werden alle Felder aufgeführt, die in diesem SubForm vorkommen. Ist keines dieser Felder im Datenstrom enthalten, wird das SubForm nicht gedruckt.


Syntax

SubForm {
  Name Name
  Fields Field1, Field2, ...,Fieldn
  Fields *FeldName** (* und :n sind ODER-gesetzt und
  Fields *FeldName:n* schließen sich somit aus)
  Height Nmm
}

FeldName
Felder aus dem lokalen Bereich oder über ein Prefix auf einen anderen Bereich angezogen.

```
Beispiel:

FIELDS $global.STXRDI_TDFORM
```

FeldName*
Für alle Felder, die mit * enden, wird ein Subformular solange angezogen, bis alle Felder mit dieser Spezifikation abgearbeitet sind. Beachte ODER-Setzung zu :n.

```
Beispiel:

FIELDS VK310_ITX*
```

FeldName:n
Für alle Felder, die mit :n enden, ist ein wortweiser Umbruch möglich. N ist hier die Umbruchlänge. Bei Überschreiten der maximalen Anzahl der Zeichen in diesem Feld wird ein neues Subformular angezogen, das dann die restlichen Zeichen verarbeitet. Beachte ODER-Setzung zu .*.

```
Beispiel:

VK950_STXT:64
```

Height

FieldOrValue
Height repräsentiert die Höhe des SubForms. Die Angabe besteht aus zwei Teilen: einer Fließkommazahl und der Einheit, die ohne Leerzeichen aneinandergehängt werden. Folgende Einheiten sind zulässig: cm, in, mm und pt. Standardeinheit ist cm.

Reserve

FieldOrValue
Reserve enthält genauso wie Height eine Höhenangabe und hat dieselbe Syntax. Reserve reserviert vor dem Seitenumbruch einen Bereich unterhalb des aktuellen SubForms, der von folgenden Subformularen beansprucht werden kann:

```
Beispiel:

Position Item_Line {
      SubForm {
            Name ItemLine
            Fields VBDPR_POSNR, VBDPR_MATNR, VBDPR_ARKTX
            Height 0,5cm
            Reserve 44mm
      }
}
```

Calc

Das Objekt Calc dient zur Berechnung neuer Felder bzw. zur Erstellung von Global-Variablen und ist nicht vom Druck des SubForms abhängig. Die konkrete Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben. Das Calc-Objekt wird immer unabhängig von der Recognition ausgeführt.

Syntax

```
SubForm {
  Name nn
  Calc {
    ...
  }
}
```

PageCalc

Das Objekt PageCalc dient zur Berechnung neuer Felder und von Global-Variablen und wird erst zu dem Zeitpunkt aufgerufen, an dem das SubForm tatsächlich auf der Seite positioniert wird.
Da ein PageCalc einer PageBottomSubForm oder PageTopSubForm einer Table zwischen den PageCalcs der dynamischen SubForms aufgerufen wird, ist PageCalc zur Berechnung von Überträgen bestens geeignet.


Syntax

SubForm {
  Name nn
  PageCalc {
    ...
  }
}

Beispiel:

```
Position ITEM {
 SubForm ITEM {
  PageCalc {
   $global.SUBTOTAL=Addr($global.SUBTOTAL,AMOUNT)
   Table TABLE_CARRYOVER:ON
  }
 }
}

 Table TABLE_CARRYOVER {
  PageBottomSubForm CARRYOVER {
   PageCalc {
    CARRYOVER=$global.SUBTOTAL
   }
  }
 }
```

Concalc

Das Objekt Concalc dient zur Berechnung neuer Felder bzw. zur Erstellung von Global-Variablen und ist nicht vom Druck des SubForms abhängig. Die konkrete Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben. Das Concalc-Objekt wird immer dann ausgeführt, wenn das Ergebnis der Recognition wahr ist.

 Syntax

```
SubForm {
  Rec {
    ...
  }
  Name nn
  Concalc {
  ...
  }
}
```

TopGap

FieldOrValue
Nachdem alle SubForms für die Seite berechnet wurden, bleibt im dynamischen Bereich (zwischen NextPosPage-Y und BottomMargin-Y) eine Restlücke, welche jedoch für die folgenden Positionen nicht ausreichend ist. Diese Restlücke wird von TopGap eingefügt. Als Standard wird die Lücke unterhalb der Positionen gesetzt. Eine Abweichung vom Standard könnte z. B. eine Abschlussposition sein, die der Anwender am Fuß des dynamischen Positionsbereiches sehen will. Dann wird für dieses Subformular TopGap 1 definiert. Die Anzahl der möglichen Positionen auf einer Seite und der Zeitpunkt für den Seitenumbruch werden in keinem Fall verändert. Es ändert sich lediglich die Stelle, an welcher der übrigbleibende Leerplatz platziert wird.

 Syntax

```
TopGap n     // 0 oder 1   ( 0 = Standard)
```

PageBreak

FieldOrValue
PageBreak steuert willkürliche Seitenumbrüche, die von Subformularen abhängen. Im Standard lösen Subformulare keine Seitenumbrüche aus. Erfolgt der Seitenumbruch nach dem Subformular, so ist dies das letzte SubForm auf dieser Seite. Eine Ausnahme ist das Subformular für eine eventuell vorhandene SubTotalPosition. Folgt dieser SubTotalPosition dann eine normale Position, wird automatisch eine neue Seite erzeugt. Erfolgt der Seitenumbruch vor dieser Position, so wird auf die vorhandene Seite nur noch eine eventuell zu druckende SubTotalPosition gedruckt und dann eine neue Seite gezogen.


Syntax

PageBreak n

Werte für n:
0: keine Angaben (0=Standard)
1: vor der Position
2: nach der Position
3: vor und nach der Position

DoPrint

FieldOrValue
DoPrint erkennt 4 Werte:

Wert 0
Das Subformular wird in Abhängigkeit des Vorhandenseins der angegebenen Felder gedruckt. Der Wert 0 ist Standard, wenn Felder definiert wurden.

Wert 1
Das Subformular wird immer gedruckt. Die Feldangaben werden ignoriert. Der Wert 1 ist Standard, wenn kein Feld angegeben wurde.

Wert 2
Das Subformular wird gar nicht abgedruckt.

Wert 3
Das Subformular wird angezogen, jedoch wird kein Feld hinein generiert.

PageTopPosition
PageBottomPosition

PageTopPosition bezeichnet eine Position in der TCI, die immer vor dem SubForm in den dynamischen Bereich gedruckt wird, wenn dieses SubForm nach einem Seitenumbruch das erste SubForm auf dieser Seite ist. PageBottomPosition bezeichnet eine Position der TCI, die immer nach dem SubForm in den dynamischen Bereich gedruckt wird, wenn dieses SubForm das letzte auf der Seite ist.


Syntax

PageTopPosition	PositionName
PageBottomPosition	PositionName

Anmerkung

Positionen, die bei PageTopPosition oder PageBottomPosition angedruckt sind, ignorieren DoPrint- und Recognition-Angaben. In den SubForms der PagePositions werden alle Felder ignoriert, die mit * enden. Alle Felder mit BreakLength werden nicht umgebrochen. Diese Einschränkung ist notwendig, damit die Höhe der PagePosition konstant bleibt. Als lokale Variable bekommen die Positionen nur ein Übertragsfeld mit, falls ein solches Feld vorhanden ist. Die globalen Variablen sind mit vorangestelltem $global. erreichbar.

Recognition bzw. **Rec**

Die Recognition ist eine erweiterte DoPrint-Anweisung, die diese jedoch nicht aufhebt. Die Recognition steuert, ob ein Objekt ausgeführt wird oder nicht. Die Funktionsweise ist im Kapitel „Interne Objekte“ beschrieben.

Protect

FieldOrValue
SubForms, die mit überlaufenden Feldern arbeiten, können mehrere Ausgabe-SubForms erzeugen. Erzeugt der OMS-ReportWriter mehrere reale SubForms in den Ausgabedatenstrom, so sind diese durch ein Protect zusammengehalten. Um diese Eigenschaft verändern zu können, gibt es den Schalter Protect mit den Werten 1 und 0 (Standardwert ist 1).

WrapTable

FieldOrValue
SubForms, die mit überlaufenden Feldern arbeiten, können mehrere Ausgabe-SubForms erzeugen. Sollen diese realen SubForms als Tabelle verstanden werden, so definiert WrapTable den Namen der Tabelle. Die Eigenschaften und das Verhalten der Tabelle werden im separat zu definierenden Table-Objekt definiert.

Table

FieldOrValue
Table definiert einen Tabellennamen und ein Tabellen-Kommando

Syntax

Table Name:Command

Zum Zeitpunkt der Erstellung der SubForms befinden diese sich einfach in einer Liste hintereinander. Durch das Table Kommando wird diese Liste umgewandelt in eine Baumstruktur, in der jeder Knotenpunkt eine Tabelle darstellt. Der Tabellenname kann frei definiert werden. Optimal ist es, wenn zu dem Tabellennamen noch ein Basisobjekt Table mit demselben Namen gibt, dann werden die Eigenschaften der Tabelle durch dieses Basisobjekt gesteuert. Existiert das Basisobjekt Table nicht, so werden die Eigenschaften der Tabelle durch Standardwerte gesteuert.

Folgende Kommandos können verwendet werden:

ON:

Start einer Tabelle mit dem angegebenen Tabellennamen. Sich wiederholende ON Kommandos mit denselben Tabellennamen werden ignoriert. Kommt ein ON Kommando innerhalb einer offenen Tabelle, so wird im Baum ein neuer Tabellenzweig eröffnet, sprich die Schachtelungstiefe erhöht sich um eine Ebene. Sie bewegen sich dann in einer Untertabelle der Tabelle.

OFF:

Ende der angegeben Tabelle nach diesem SubForm. Das nachfolgende SubForm ist nicht mehr Bestandteil der Tabelle. Ist kein Tabellenname angegeben, so wird die aktuelle Tabelle geschlossen.

OFFBEFORE:

Ende der angegeben Tabelle vor diesem SubForm. Dieses SubForm ist nicht mehr Bestandteil der Tabelle. Ist kein Tabellenname angegeben, so wird die aktuelle Tabelle geschlossen.

ONOFF:

Dieses SubForm eröffnet eine Tabelle, in der nur dieses SubForm geschrieben wird. Das aktuelle SubForm eröffnet und schließt die Tabelle. Dieses Kommando wird vor allem dann benutzt, wenn die Einstellung dieser Tabelle aus dem Basisobjekt Table definiert wird.

RESTART:

Ende der aktuellen Tabelle vor diesem SubForm und Start einer neuen Tabelle mit dem angegebenen Namen. Da RESTART die aktuelle Tabelle abschließt, kann dieses Kommando keine Untertabellen erzeugen.

RESTARTOFF:

Ende der aktuellen Tabelle vor diesem SubForm, Start einer neuen Tabelle mit dem angegebenen Namen und Stop dieser Tabelle nach dem aktuellen SubForm. Da RESTARTOFF die aktuelle Tabelle abschließt, kann dieses Kommando keine Untertabellen erzeugen.

RESTARTAFTERCHANGE:

Ende der aktuellen Tabelle vor diesem SubForm und Start einer neuen Tabelle mit dem angegebenen Namen. Allerdings wird der Restart nur ausgeführt, wenn zwischen dem letzten Tabellen-Eröffnungs-Kommando und den RESTARTAFTERCHANGE-Kommando mindestens ein SubForm mit einem anderen Tabellennamen vorkommt. Da RESTARTAFTERCHANGE die aktuelle Tabelle abschließt, kann dieses Kommando keine Untertabellen erzeugen.

RESET:

Ende der aktuellen Tabelle nach dieser Position und Rückkehr aus allen Verschachtelungsebenen. Die nächste Position beginnt wieder auf dem obersten Level. Die Angabe eines Tabellennamens ist nicht notwendig und wird ignoriert.

RESETBEFORE:

Ende der aktuellen Tabelle vor diesem SubForm und Rückkehr aus allen Verschachtelungsebenen.
Das aktuelle SubForm beginnt wieder auf dem obersten Level. Die Angabe eines Tabellennamens ist nicht notwendig und wird ignoriert.

Beispiel:

```
SubForm {
 Name Start
 Table ProtectPos:ON
}
SubForm {
 Name Content
}
SubForm {
 Name Stop
 Table ProtectPos:OFF
}
```

TableControl

FieldOrValue
TableControl ist ein Schalter mit dem Werten 0 und 1 und dem Standardwert 0. Ist TableControl eingeschaltet, so wird dieses SubForm nicht mit ausgegeben. Der Sinn dieses SubForms besteht in dem gleichfalls zu definierenden TableCommand.

SubPositions

SubPositions definieren die Reihenfolge und die Auswahl von SubPositions, welche mit der aktuellen Position ausgegeben werden. Bei der Angabe der SubPositions werden folgende Einstellungen festgelegt:

Kommt die SubPosition aus dem Datenstrom oder ist diese fix anzugeben?
Aus welcher Tabelle kommen die Daten der SubPosition?
Wird die Reihenfolge der SubPositionen aus dem Datenstrom bestimmt oder ist diese fest vorgegeben?
Ist eine Sortierung der SubPositionen notwendig?

 Syntax

SubPositions Section[,Section[...]]

Sektionen sind bestimmte Bereiche von SubPositionen, die Sortierung, Datentabelle und die Entscheidung über fixe oder dynamische Position miteinander teilen.
Die Sektionen werden in der Reihenfolge von links nach rechts abgearbeitet. Eine Sektion kann eine fixe SubPosition oder eine dynamische Auswahl von SubPositionen aus der aktuellen dynamischen Position sein.

Fixe SubPosition

Eine fixe SubPosition ruft unabhängig vom Datenstrom eine TCI-SubPosition auf. Die gerufene SubPosition hat aus diesem Grund auch keine lokalen dynamischen Daten, die im SubForm-Objekt der SubPosition mit angedruckt werden können.

 Syntax

PosName

Dynamische SubPositionen aus der aktuellen dynamischen Position

Die Sektion besteht aus einer Aufzählung einer oder mehrerer TCI-Positionen, die in den SubPositionen der aktuellen Position gesucht werden. Dabei werden alle SubPositionen, die der Aufzählung entsprechen, in eine temporäre Liste überführt. Diese Liste kann sortiert und unsortiert ausgegeben werden. Die Werte des Sortierfeldes können numerisch oder alphanumerisch sein. Treten numerische und alphanumerische Werte gemischt auf, so werden die numerischen Werte vor die alphanumerischen Werte sortiert.

Syntax

PosName[|PosName [...]]([SortField])

Als Alternative zur genauen Angabe der SubPositionen kann für dynamische SubPositionen auch eine Wildcard angegeben werden. Es werden alle SubPositionen der aktuellen Position zum Druck angestoßen, die als dynamische Position verfügbar sind.

```
Beispiel:

SUBPOSITIONS *()
```

Label Value

FieldOrValue
Label ist eine Art Kennzeichnung, ähnlich einem Keyword, das dazu dient, der Seite mitzuteilen, welche Art von Subforms sie enthält. Die Label aller Subforms der Seite werden verdichtet, so dass jede Wiederholung ausgeschlossen ist. Die Liste aller gefunden Label wird dann als kommaseparierter String als Page_Label geführt und bei den Metadaten an der Seite mit ausgegeben. Page_Label ist damit auch ein weiteres feststehendes Schlüsselwort in den Metadaten, das nicht anders gesetzt werden kann. Durch diese Funktionalität ist ein nachfolgendes Analyseprogramm, wie der OMS-PDFxPlore, in der Lage, einzelne Seiten nach ihrem Inhalt zu bewerten. So kann z.B. die Seite mit der Entgeltabrechnung der DV-Freimachung ein entsprechendes Label erhalten, das es erlaubt, diese Seite in einem Spooljob zu finden und zu extrahieren.

Zusätzliche Basisobjekte

CommonSettings

Verwendung

CommonSettings beinhaltet Schalter, die das globale Verhalten des OMS-ReportWriters beeinflussen. Alle Schalter sind vom Typ FieldOrValue. Da diese Schalter aber nicht Dokument-bezogen sind, werden die mit dem Kommandozeilen-Parameter –adv übergebenen Variablen als Fields genutzt.

Syntax

CommonSettings {
  SeparateCentralChannels Value
  SeparateCentralQualifiers Value
  SeparateCentralIncompatiblePaperTypes Value
  SeparateCentralMaximumPages Value
  SeparateCentralSimplex Value
  SeparateArchiveMaximumDocs Value
  SeparateArchiveDocType Value
  SeparateArchiveFileSize Value
  ArchiveBackPages Value
  ArchiveSinglePages Value
  ArchiveFileType Value
  StrictlyCentralCheck Value
  TrimFields Value
  PoststalService Value
  AutoProtectTables Value
  RemoveEmptyTables Value
  JetFormGroupCompatible Value
  PageBreakSpaceEfficiency Value
  SortAllOutputDocuments Value
  IgnoreUndefinedPositions Value
  OutFileDocRefFields List
  OutFileSAPRefFields List
  OutFileDocRefHeader List
  OutFileSAPRefHeader List

ResolveEntities Value
ResolveInlineEntities Value
DecentralRefFiles Value
MaxJoinWeight Value
UseJoinKeyAsDocRef Value
IndicantOnlyForFirstSpoolInSeries Value
DuplexReduction Value
CancelOnInsertionMismatch Value
}


Erklärung

SeparateCentralChannels Value

FieldOrValue
Die zentralen Druckfiles werden nach Kanälen getrennt. Der Schalter hat als Werte 0 und 1, wobei 0 der Standardwert ist.

SeparateCentralQualifiers Value

FieldOrValue
Die zentralen Druckfiles werden nach dem im DocDef definierten Qualifier getrennt. Mit Hilfe der Qualifier und des Schalters SeparateCentralQualifiers können zentrale Dokumente nach einem Qualifier separiert werden. Es entstehen für jeden vorhandenen Qualifier separate Ausgabefiles. Der Schalter hat als Werte 0 und 1, wobei 0 der Standardwert ist.

SeparateCentralIncompatiblePaperTypes Value

FieldOrValue
SeparateCentralIncompatiblePaperTypes ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Wurde der Schalter aktiviert, wird überprüft, ob die PaperType/Intray-Kombinationen eines OutputFiles widersprüchlich sind. Ist dies der Fall, werden so viele OutputFiles angelegt, bis die Widersprüche aufgelöst sind.

SeparateCentralMaximumPages Value

FieldOrValue
Die Druckstapel des zentralen Druckes werden bei Benutzung dieser Option nur bis zu der dort angegebenen Anzahl an physikalischen Seiten aufgefüllt. Die restlichen Dokumente kommen in den nächsten Druckstapel. Der Wert 0 bedeutet, dass die Option ausgeschaltet ist. Gleichzeitig ist 0 der Standardwert.

```
Beispiel:

CommonSettings {
   SEPARATECENTRALMAXIMUMPAGES 2000
}
```

SeparateCentralSimplex Value

FieldOrValue
SeparateCentralSimplex ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so werden Simplex-Dokumente in einen separaten Druckstapel generiert. Auf diese Art werden Simplex- und Mixplex-Dokumente voneinander getrennt. Dieser Schalter ist für Endlosrollendrucker im Tandem-Betrieb wichtig. Die Simplex-Datei enthält in ihrem Namen die Zeichenfolge X_. Weiterhin werden zusätzliche Variablen in Header- , Trailer und Indicant-Dokument und in die Meta- und Vol-Daten erzeugt

SeparateArchiveMaximumDocs Value

FieldOrValue
Um die maximale Anzahl von Dokumenten, die in einen Archiv-File generiert werden, zu begrenzen, kann SeparateArchiveMaximumDocs verwendet werden. Der Wert "0" bedeutet keine Separierung. Alle anderen positiven Werte beschreiben die maximale Anzahl von Dokumenten.

SeparateArchiveDocType Value

FieldOrValue
Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist SeparateArchiveDocType eingeschaltet, so wird in jedem Archiv-Dokument die ArchiveRef-Variablen-DocType gesucht und für jeden dort gefundenen DocType eine oder mehrere separate Archiv-Files erzeugt. Der DocType wird als Qualifier für den Archiv-File gesetzt und erscheint sowohl im Namen als auch in der Vol-Datei.

SeparateArchiveFileSize Value

FieldOrValue
Dieser Schalter wirkt sich nur auf die ArchiveType META und XML aus und separiert die ArchiveFiles, sobald die Größe des ArchiveFiles überschritten wurde. Der Parameter ist ungenau, da vor dem Schreiben nicht geprüft werden kann, sondern erst danach. Nach der Zahl kann noch eine Maßeinheit stehen: KB für Kilobyte, MB für Megabyte und GB für Gigabyte. Standardwert ist 1.5 GB.

ArchiveBackPages Value

FieldOrValue
Der Schalter hat als Werte 0 und 1, wobei 0 der Standardwert ist. Wird der Schalter auf 1 gesetzt, so werden fixe Rückseiten, die im WorkItem unter BackPageDesignsFirst bzw. BackPageDesignsNext angegeben wurden, mit archiviert. Dies ist von Bedeutung, wenn die Rückseite rechtsrelevante Texte beinhaltet.

ArchiveSinglePages Value

FieldOrValue
Der Schalter hat als Werte 0 und 1, wobei 0 der Standardwert ist. Wird der Schalter auf 1 gesetzt, so wird für jede einzelne Seite eines zu archivierenden Dokuments ein separater PDF-File erzeugt. Andernfalls wird pro Archiv-Dokument ein PDF-File erzeugt.

ArchiveFileType Value

FieldOrValue
Der Schalter ArchiveFileType mit erweiterten Funktionen ersetzt den Schalter ArchiveBase64. ArchiveFileType bestimmt, in welchem Format die Archivdaten angelegt werden. ArchiveFileType ist ein Schalter mit folgenden Werten:

METAPDF	Klassisches MetaFormat mit externen PDFs
METABASE64	Klassisches MetaFormat mit Base64 kodierten internen PDFs
XMLPDF	XML-File mit externen PDFs
XMLBASE64	XML-File mit Base64 kodierten internen PDFs
ARCHIVEFILE	PDF-Archivdatei zusätzlich zu den Druckdateien. Die Archiv-Informationen sind im seitenbezogenen XMP enthalten. (Standardwert)
PRINTFILE	Die Druckdokumente erhalten die Archiv-Informationen als seitenbezogenes XMP.

XMLPDF und XMLBASE64 werden ab dem Archiver 5.0 unterstützt, ebenso ARCHIVEFILE und PRINTFILE. Das XML-Format für die Archivierung entspricht auch dem XML, welches für OutputType XML verwendet wird und von vielen OMS-Produkten gelesen wird. Alle Werte außer PRINTFILE führen zu einer von der Druckdatei separaten Archivdatei. Das bedeutet, dass Schalter wie ArchiveText, ArchiveDesignFirst usw. zur Anwendung kommen und Archiv-dokumente anreichern können.

Im Gegensatz hierzu wird bei PRINTFILE die fertige Druckdatei verwendet und mit unsichtbaren Archivinformationen angereichert. Dies führt dazu, dass ArchiveText, ArchiveDesignFirst usw. nicht ausgewertet werden, dafür aber Duplex-Informationen, OMR-Marken, Copy-Texte usw. enthalten sind. Weiterhin lassen sich über PRINTFILE nur die Dokumente archivieren, die auch gedruckt werden. Soll ein Dokument nicht gedruckt, aber archiviert werden, so ist PRINTFILE nicht die richtige Einstel-lung. Da PRINTFILE keine extra Archivdatei erzeugt, ist es die schnellste Art, Archivdaten zu erzeugen.

StrictlyCentralCheck Value

FieldOrValue
In vielen Kundenprojekten hat es sich als notwendig erwiesen, dass zentrale und dezentrale Drucke unterschiedlich behandelt werden. Die Trennung zwischen zentralen und dezentralen Drucken ließ sich zwar durch den Schalter Central im WorkItem definieren, jedoch wurden einige Funktionen des zentralen Drucks trotzdem ausgeführt. Der Schalter StrictlyCentralCheck mit dem Standardwert 0 sorgt nun dafür, dass die Abgrenzung zwischen zentralen und dezentralen Drucken vollständig ist.

TrimFields Value

FieldOrValue
TrimFields sorgt dafür, dass Feld-Inhalte vor der Ausgabe um führende Leerzeichen bereinigt werden. Ebenfalls werden nachfolgende Zeilenumbrüche (Leerzeilen) und Leerzeichen beseitigt. TrimFields ist ein Schalter mit den Werten 1 und 0, wobei 0 der Standardwert ist.

```
Beispiel:

COMMONSETTINGS {
  ...
  TRIMFIELDS 0
  ...
}

In diesem Beispiel wird die Funktionalität der Trim ausgeschaltet.
```

PoststalService Value

FieldOrValue
Um die DPDV-Freimachung für die Deutsche Post AG und andere Postdienstleister zu aktivieren, muss in der TCI im Basis-Objekt CommonSettings der Schalter PostalService auf 1 gesetzt werden.

AutoProtectTables Value

FieldOrValue
AutoProtectTables ist ein Schalter mit den Werten 0 und 1. Der Standardwert ist 0. AutoProtectTables definiert den Standardwert für das Schlüsselwort Protect des Objektes SubForm.

RemoveEmptyTables Value

FieldOrValue
Im direkten Zusammenhang mit dem Schlüsselwort TableControl des Objektes SubForm steht der Schalter RemoveEmptyTables. Da nach dem Entfernen von SubForms, die nur zum Aufbau des Tabellenbaumes dienten, leere Tabellen übrig bleiben können, ist die Entscheidung zu treffen, wie mit diesen leeren Tabellen umzugehen ist. Die Verfahrensweise wird durch den Schalter RemoveEmptyTables mit den Werten 0 und 1 geregelt. Standardwert ist 1.

JetFormGroupCompatible Value

FieldOrValue
JetFormGroupCompatible ist ein Schalter mit den Werten 1 und 0, Standard ist 0.
Ist JetFormGroupCompatible angeschaltet, so werden ^GROUP-Befehle als ^POSITION und ^ENDGROUP-Befehle als ^ENDPOSITION gelesen. Dieser Modus ist vor allem dann hilfreich, wenn der Datenstrom des liefernden Verfahrens bereits ^GROUP Befehle für JetForm enthielt und nun auf OMS-ReportWriter umgestellt wird.

PageBreakSpaceEfficiency Value

FieldOrValue
Im OMS-ReportWriter arbeitet ein Algorithmus, der versucht, so viele SubForms wie irgend möglich auf die Seite zu ziehen. Das spart Papier- und Portokosten. Um die Optimierungsrate einstellen zu können, gibt es den Schalter PageBreakSpaceEfficiency, mit dem gesteuert werden kann, ob der Algorithmus aktiv wird und wie aggressiv er arbeitet. PageBreakSpaceEfficiency kennt die Werte 0 bis 2, wobei 1 der Standardwert ist.

0 - Algorithmus ist ausgeschaltet:

beachtet alle Reserve-Anweisungen der SubForms und versucht, unsinnige Umbrüche wegzuoptimieren, die von BottomPosition verursacht werden

1 - PageBottom Optimierung

es werden unsinnige PageBottom Reservierungen aufgelöst, wenn durch Verhinderung des Umbruches Platz gespart werden kann

2 - Reserve Optimierung

wie 1, ignoriert aber Reserve-Anweisungen der SubForms , wenn diese in den nachfolgenden Positionen nicht wiederholt werden.

SortAllOutputDocuments Value

FieldOrValue
Das Logistik-Modul im OMS-ReportWriter verarbeitet Dokumente, die nicht DPDV-freigemacht werden sollen, direkt, ohne diese noch einmal auf der Festplatte zwischenzuspeichern. Dokumente, die DPDV-freigemacht werden sollen, werden für die spätere Sortierung noch einmal auf die Festplatte geschrieben. Werden nun Dokumente, die DPDV-freigemacht werden sollen, durch eine Unterschreitung der Mindermengen oder ähnliche Randbedingungen zurückgestuft, so als wären diese nicht DPDV-freizumachen, gibt es eine Verschiebung der Dokumentenreihenfolge in den Ausgabefiles. Am Anfang des Ausgabefiles stehen alle Dokumente, die von Anfang an nicht DPDV-freizumachen waren, gefolgt von den zurückgestuften Dokumenten. Um diesen Verschiebung der Dokumentenreihenfolge zu verhindern, kann mit SortAllOutputDocuments erzwungen werden, dass alle Dokumente in den Sortierlauf der DPDV-Freimachung gelangen und damit die Reihenfolge aller nicht DPDV-freigemachten Dokumente beibehalten wird. SortAllOutputDocuments ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist.

IgnoreUndefinedPositions Value

FieldOrValue
IgnoreUndefinedPositions ist ein Schalter mit den Werten 0 und 1 und dem Standardwert 0. Steht der Schalter auf 1 und eine im Datenstrom befindliche Position wird weder in der TCI noch im XDP-File gefunden, so bricht das Programm nicht ab, sondern ignoriert die Position.

OutFileDocRefFields List

FieldOrValue
OutFileDocRefFields ist eine kommaseparierte Liste von Variablen-Namen oder Fix-Strings, die in der angegebenen Reihenfolge als CSV-Zeile in einen DocRef-File geschrieben werden, der pro Kanal1-File entsteht. Fix-Strings beginnen und enden mit Anführungszeichen. Die hier aufgeführten Variablen sollten entweder mit dem Schlüsselwort DocRef-Fields des DocDef in die DocRef-Liste des Dokuments kopiert worden sein oder aus der Liste der Standard DocRef-Fields kommen, die unter "Zusätzlichen Ausgabevariablen/DocRef" weiter hinten beschrieben sind. OutFileDocRef-Files haben denselben Namen wie die Dat-Files. Die Endung ist allerdings ".docref".

OutFileSAPRefFields List

FieldOrValue
OutFileSAPRefFields ist eine kommaseparierte Liste von Variablen-Namen oder Fix-Strings, die in der angegebenen Reihenfolge als CSV-Zeile in einen SAPRef-File geschrieben werden, der pro Kanal1-File entsteht. Fix-Strings beginnen und enden mit Anführungszeichen.
Die hier aufgeführten Variablen sollten entweder mit dem Schlüsselwort DocRefFields des DocDef in die DocRef-Liste des Dokuments kopiert worden sein oder aus der Liste der Standard DocRef-Fields kommen, die unter "Zusätzlichen Ausgabevariablen/DocRef" weiter hinten beschrieben sind. OutFileSAPRef-Files haben denselben Namen wie die Dat-Files. Die Endung ist allerdings ".sapref".

OutFileDocRefHeader List
OutFileSAPRefHeader List

Beide Schlüsselwörter definieren einen Header für den DocRef- bzw. SAPRef-File.
Dieser Header wird dem eigentlichen Fileinhalt vorangestellt, wenn der File neu angelegt wird.
Der Header darf keine Leerzeichen enthalten.

ResolveEntities Value

FieldOrValue
ResolveEntities bewirkt, dass Character- und Entity-Referenzen in Konfiguration- und Input-Dateien aufgelöst werden. Eine genaue Beschreibung befindet sich im Kapitel Internationalisierung. Standardwert ist 1.

ResolveInlineEntities Value

FieldOrValue
Der Schalter ResolveInlineEntities bewirkt, dass die Entities \n \t \u+ bereits in den Input-Interfaces RDI und JF in Buchstaben umgesetzt werden und so allen folgenden Prozessschritten zur Verfügung stehen. Die Funktionalität ist unabhängig von der Auflösung bei der Generierung über den Schalter reportw.ini\PDF\Texts\AllowInlineCommands. Standardwert ist 0.

DecentralRefFiles Value

FieldOrValue
DecentralRefFiles bewirkt, dass auch für dezentrale Dokumente Einträge in die DocRef- und SAPRef-Dateien getätigt werden. DecentralRefFiles ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist.

MaxJoinWeight Value

MaxJoinWeight ist eine Gewichtsangabe in Gramm, die angibt, bis zu welcher Grammzahl Dokumente gleichen JoinKeys zusammengefasst werden. Standardwert ist 0. Die Grammzahl 0 bedeutet, dass der Schalter ausgeschaltet ist;. es gibt keine Beschränkungen.

UseJoinKeyAsDocRef Value

FieldOrValue
UseJoinKeyAsDocRef ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird für Dokumente, die mit JoinKey zusammengefasst werden, der JoinKey als DocRef gesetzt. Ist der Schalter ausgeschaltet, so wird das DocRef des zusammengefassten Dokuments aus den DocRef aller einzelnen Dokumente zusammengesetzt. Das + Zeichen ist in diesem Fall der Separator.

IndicantOnlyForFirstSpoolInSeries Value

FieldOrValue
IndicantOnlyForFirstSpoolInSeries ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so bekommt bei einer Serienbildung über den Schalter SeparateCentralMaximumPages nur der erste SpoolJob in der Serie ein Indicant-Dokument.

DuplexReduction Value

FieldOrValue
DuplexReduction ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so werden Vorderseiten von Duplex-Seiten, die keine Rückseite besitzen auf Simplex zurück gesetzt.

CancelOnInsertionMismatch Value

FieldOrValue
CancelOnInsertionMismatch ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Trifft die abstrakte Definition des EnvelopeSortSystems auf ein reales EnvelopeSortSystems, so kann es sein, dass z.B. die Anzahl der angezogenen Trays nicht übereinstimmen. Ist der Schalter eingeschaltet, so bricht das Programm ab, wenn eine solche Fehlzuordnung vorliegt. Ist der Schalter ausgeschaltet, so wird eine Fehlermeldung generiert und die Arbeit fortgesetzt.

DocumentSorter

Verwendung

Der DocumentSorter ist eine vorgelagerte Sortierlogik zur Dokument-Sortierung. Der DocumentSorter arbeitet nach den Interfaces und vor der eigentlichen Dokument-Verarbeitung. Aus diesem Grund ist der DocumentSorter auch für alle Interfaces verfügbar und unabhängig vom RDI-, XML- oder dem Adobe Present-Interface. Der DocumentSorter speichert alle Dokumente in einem temporären File auf der Festplatte und merkt sich im Hauptspeicher den Wert der Sortiervariablen und die Position des Dokuments im temporären File. Bedingt durch diese Technik können nur Druckjobs sortiert werden, die kleiner als 2GByte sind.

Syntax

```
DocumentSorter {
  Enabled Value
  SortVars {
    ...
  }
  ...
}
```

Erklärung

Enabled Value

Vor allen anderen Einstellungen ist das Dokument erst einmal zu aktivieren. Dies bewirkt, dass der Umweg über die temporäre Datei aktiviert wird. Enabled ist ein Schalter, der die Werte 1 und 0 kennt.

Syntax

```
DocumentSorter {
  Enabled 1
  ...
```

}

SortVars

Im Objekt SortVars werden die Sortierkriterien definiert.

Syntax

```
DocumentSorter {
  Enabled 1
  SortVars {
  ...
  }
}
```

Die Sortierkriterien, die in SortVars definiert werden, sind nach Ihrer Priorität festgelegt. Jede Zeile im SortVars -Objekt definiert ein Sortierkriterium. Je höher ein Sortierkriterium steht, umso höher ist dessen Wichtung. Die Zeile zur Definition eines Sortierkriteriums hat folgenden Aufbau:

VARNAME1[,VARNAME2[...]] [TYPE [ASC]]

Da in unterschiedlichen Dokumenten das Sortierkriterium auch unterschiedliche Variablen-Namen haben kann, sind mehrere VARNAME erlaubt, wobei der Algorithmus diese Felder in der Reihenfolge sucht, wie sie angegeben sind. Der Algorithmus wird unterbrochen, sobald er eine dieser Variablen mit einem Wert belegen konnte.
Als TYPE sind im Moment zwei Typen erlaubt: INTEGER und STRING.
STRING ist der Default-Wert. ASC bestimmt, ob für dieses Kriterium aufsteigend oder absteigend sortiert wird. ASC ist ein Schalter mit den Werten 0 und 1 (0 für absteigend und 1 für aufsteigend). Defaultwert ist 1 - also aufsteigend.

Beispiel:

```
DOCUMENTSORTER {
  ENABLED 1
  SORTVARS {
    POSTAL CODE INTEGER 1
    STREET STRING 0
  }
}
```

Es wird nach PLZ aufsteigend sortiert und innerhalb einer PLZ nach Straße absteigend.

Zusätzlich zum Enabled-Schalter in der TCI kann die Sortierung auch noch durch einen Parameter in der Kommando-Zeile unterbunden werden: -nds (no document sort).

Substitute

Verwendung

Substitute ist eine Tabelle mit Austauschwerten, die entsprechenden Feldern zugewiesen werden.

Syntax

```
Substitute Name {
  Quelle Resultat1
  Quelle @GetSubstitute (PN-Tabelle,PN-Wert)
  Quelle @Input
  Quelle @Calc(PN)
  ...
  Default Resultat Default
}
```

Erklärung

Substitute

Jedes Substitute-Objekt hat einen eigenen Namen, da mehrere solcher Objekte im TCI-File vorkommen können. Die möglichen Suchwerte sind links in der Tabelle aufgelistet (Quelle). Auf der rechten Seite ist der dazugehörige Rückgabewert definiert. Der Rückgabewert ist immer eine FieldOrValue-Definition, wobei der zusätzliche Befehl @Input erlaubt ist. Das bedeutet, dass alle Befehle, die in FieldOrValue-Angaben erlaubt sind, auch als Rückgabewert einer Substitute-Tabelle erlaubt sind. Die Suche im Substitute-Objekt ist Case-insensitiv.

Resultat1

Ist ein Rückgabewert gefunden, wird der Suchwert vom Rückgabewert ersetzt.

@GetSubstitute

Ein Rückgabewert kann kombiniert werden mit einem weiteren Substitute-Objekt. Der GetSubstitute-Befehl wird durch den Calc-Befehl Substitute realisiert und entspricht dessen Syntax. Dadurch wird der Entscheidungsbaum des Substitute-Objektes weiter vertieft. Mögliche Endlosschleifen müssen vom Ersteller der TCI ausgeschlossen werden.

@Input

Input repräsentiert den Wert, der in die Tabelle eingegeben wurde.

Wird in folgendem Beispiel der Drucker im Feld JF_PRINTER_DEVICE nicht in der Substitute-Tabelle PRINTER gefunden, so erfolgt der Ausdruck über den Namen, der im Feld JF_PRINTER_DEVICE steht, ohne eine Konvertierung.

```
Beispiel:

SUBSTITUTE PRINTER {
      HP5SI     \\SERVER1\PRINTER1
      LEXMARK1  \\SERVER1\PRINTER2
      Default   @INPUT
}
DocDef BUSSI {
      ...
      Printer @GETSUBSTITUTE(“PRINTER“,JF_PRINTER_DEVICE)
      ...
}
```

@Calc

Über den Calc-Befehl wird der Rückgabewert aus den typischen Calc-Befehlen errechnet. Der Übergabeparameter ist eine Polnische Notation, wie sie in allen Calc-Objekten verwendet wird. Weitere Informationen zur Verwendung von Calc-Befehlen finden Sie unter Interne Objekte/Calc.

```
Beispiel:

SUBSTITUTE PRINTER {
      HP5SI     \\SERVER1\PRINTER1
      LEXMARK1  \\SERVER1\PRINTER2
      Default   @CALC(CONCATS(“PRINTER“,ARBEITSPLATZ))
}
DocDef BUSSI {
      ...
      Printer @GETSUBSTITUTE(“PRINTER“,JF_PRINTER_DEVICE)
      ...
```

```
}
```

Resultat Default

Wenn kein Rückgabewert gefunden wird, dann wird Default zugeordnet. Default ist hier der Standardwert.

```
Beispiel:

Substitute VK_ORG {
 100   HO_RE_DATE, 10; HO_RE_COMPANY_DESIGN, 16
      ...
}
Substitute RE_DESIGN {
 0001100   RE_DATE, 11; RE_COMPANY_DESIGN, 17
 0011000   @GetSubstitute("VK_ORG, HO")
 ...
}
```

Es ist jedoch auch möglich, anstelle der FieldOrValue-Angaben fixe Werte anzugeben. Im Beispiel wird das Feld JF_PRINTER1 gelesen und in der Tabelle PrinterTab konvertiert.

```
Beispiel:

Printer @GetSubstitute("PrinterTab", JF_PRINTER1)
```

Der Name XEROXHEADQUARTER1 wird in der Tabelle PrinterTab konvertiert.

```
Beispiel:

Printer @GETSUBSTITUTE("PrinterTab", "XEROXHEADQUARTER1")
```

Interne Objekte

Calc

Verwendung

Das Objekt Calc dient zur Berechnung von Feldern.

Syntax

```
Calc {
  Feld1=PN
  ...
}
```

PN: Variable
PN: FixText
PN: Funktion

Erklärung

PN

PN steht für Polnische Notation (PN). Dabei wird nach den Kriterien Variable, FixText und Funktion unterschieden. Die Übergabeparameter einer Funktion sind ebenfalls Polnische Notationen, welche eine Variable, einen FixText oder eine Funktion beinhalten können. Diese Verschachtelungslogik ist in ihrer Tiefe nicht beschränkt.

Variable

Es wird immer ein Feld dem anderen zugewiesen. Das vorangestellte GLOBAL verweist darauf, dass diese Variable in der Liste der Global-Variablen gesucht bzw. erzeugt werden muss. Existiert die Zielvariable bereits, so wird sie mit dem neuen Wert überschrieben. Bei Angaben eines Variablen-Namens zum Lesen ist auch folgende Syntax zulässig: „*.VARNAME“. Die Variable „VARNAME“ wird dann vom aktuellen Fokus her abwärts gesucht. Erfolgt die Angabe zum Beispiel in einem WorkItem, so kann sich die gesuchte Variable im GLOBAL-Bereich oder aber auch in einer Position befinden. Das erste Auffinden der gesuchten Variable bestimmt den Wert.

```
Beispiel:

Feld1             = Feld2
Feld3             = $global.Feld1
$global.Feld3     = Feld2
```

FixText

Dem Feld wird ein fixer Text zugewiesen. Dies wird deutlich durch die Anführungsstriche vor und hinter dem Wert. Es gibt spezielle Zeichen im FixText, die vom OMS-ReportWriter uminterpretiert werden:

\n Zeilenvorschub
\“ Anführungsstriche im Text
\‘ Hochkomma im Text
\\ Backslash im Text

```
Beispiel:

Feld1             = “Hallo Welt“
$global.Feld3     = “Hallo\nWelt“
```

Funktionen

AddDate

AddDate addiert zu einem als Input gegebenen Tagesdatum weitere Tage hinzu und gibt
als Ausgabe das errechnete Datum zurück. Dabei können die Wochenenden ignoriert werden, so dass nur die Werktage gezählt werden.


Syntax

AddDate(PN1,PN2,PN3)

PN1: InputDatum im Format DD.MM.YYYY oder DD.MM.YY
PN2: Anzahl Tage, die addiert werden sollen
PN3: Wochenende-Schalter mit YES/NO oder 1/0; bei YES/1 werden Tage an Wochenenden mitgezählt

```
Beispiel:

ADDDATE("12.07.2002","3","NO")
Das Ergebnis ist 17.07.2002
```

AddI

AddI addiert die Integerwerte aller resultierenden Strings zueinander.


Syntax

AddI(PN1, PN2,..., PNn)

AddF

AddF addiert reelle Zahlen (Gleitkommazahlen) zueinander.

Syntax

AddF(PN1,PN2,PN3,PN4,PN5,PN6)

PN1: 1. Summand (als lokalisierter und formatierter String)
PN2: 2. Summand (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)
PN5: Picture-Format in dem das Ergebnis dargestellt werden soll (optional, Standard InPicture)
PN6: Unit (Einheit), die beim Formatieren des Ergebnisses verwendet wird (optional, Standard ist leer)

```
Beispiel:

ADDF("1.200,00","1.300,00","de_DE")
Das Ergebnis ist 2.500,00
```

Address

Address dient zum Erstellen eines Adressfeldes aus Einzelwerten. Dieser Adress-Operator arbeitet nach der SAP-Logik und generiert Adressen vom Typ 1 mit 10 Zeilen und dem Absenderland Deutschland.

Syntax

Address(PN1,...,PN17)

Die Parameter haben der Reihenfolge nach folgende Namen.

1 TITLE_TEXT	Anrede
2 NAME1	1. Name
3 NAME2	2. Name
4 NAME3	3. Name
5 NAME4	4. Name

6 NAME_CO	Name beim Senden über c/o
7 STREET	Strasse
8 HOUSE_NUM1	Hausnummer
9 CITY1	Ort
10 CITY2	Ortsteil
11 POST_CODE1	Postleitzahl
12 POST_CODE2	Postleitzahl PO_Box
13 PO_BOX	Postfach
14 PO_BOX_LOC	Ort des Postfaches
15 LOCATION	Gebiet - Erweiterung zum Ort für ausländische Adressen
16 REGION	Region - Erweiterung zum Ort für ausländische Adressen
17 COUNTRY	Land des Empfängers - zweistelliges Kürzel DE für Deutschland

And

Verknüpft alle PNs nach den Regeln des Logischen And.

Syntax

And(PN1,...,PNn)

At bzw. **@**

Mit At kann ein Feldnamen zusammengesetzt und der Inhalt des dazugehörigen Feldes ermittelt werden. Der Befehl arbeitet in zwei Schritten. Im ersten Schritt wird der Feldname durch aneinanderreihen aller Übergabeparameter zusammengesetzt. Im zweiten Schritt wird das Feld mit dem zusammengesetzten Feldnamen gesucht und der Inhalt des Feldes zurückgegeben.

Syntax

At(PN1,...,PNn) bzw.
@(PN1,...,PNn)

```
Beispiel:

@("$global.",FLDNAME,"[",VORKOMMEN,"]")

Wenn FLDNAME den Wert 'Person' hat und VORKOMMEN den Wert '2' hat,
dann ergibt sich die Feldbezeichnung $global.Person[2].
Es wird das Feld $global.Person[2] gesucht und der Inhalt des
Feldes zurückgegeben. Das Ergebnis könnte z.B. 'Marlon Brando'
sein.
```

BuildChartTable

BuildChartTable dient zur Erstellung von Chart-Feldern. BuildChartTable liest Einzelfelder der aktuellen Position zusammen und bildet aus diesen ein Chart-Feld, wie dies vom XDP-Chart-Objekt verstanden wird. BuildChartTable untersucht dabei die lokalen Feldnamen und ordnet diese einer Tabelle zu, die ähnlich aufgebaut ist wie eine MS-Excel-Tabelle. Jeweils einstellbar ist die Interpretation der Feldnamen und die Bildung der Überschriften.

Syntax

BuildChartTable(PN1,...,PN5)

1 ColumnSource	FIELDNAME|FIELDNUMBER|FIELDALPHANUMBER. FIELDALPHANUMBER ist Standard (Pflicht)
2 RowSource	FIELDNAME|FIELDNUMBER|FIELDALPHANUMBER. FIELDNUMBER ist Standard (Pflicht) Es kann genau einer dieser Werte eingestellt sein.

3 Options	MIRROR SEPARATOR (Optional) Es können auch mehrere Werte angegeben werden, wenn diese Leerzeichen separiert sind. MIRROR tauscht die Spalten mit den Zeilen; SEPARATOR definiert ein Separatorzeichen, was im Feldnamen zwischen Column und Row steht
4 ColumnTitles	NAMES|FIRSTENTRY (Optional) Es kann genau einer dieser Werte eingestellt sein. NAMES Es werden die Teilnamen aus den Feldnamen benutzt FIRSTENTRY Die ganze Tabelle wird um eine Position nach oben gerutscht Dadurch wird die obere Zeile zu Spaltenüberschriften
5 RowTitles	NAMES|FIRSTENTRY (Optional) Es kann genau einer dieser Werte eingestellt sein. NAMES Es werden die Teilnamen aus den Feldnamen benutzt FIRSTENTRY Die ganze Tabelle wird eine Position nach links gerutscht Dadurch wird die linke Spalte zu Zeilenüberschriften

```
Beispiel für Source:

Wenn ColumnSource auf dem Wert FIELDALPHANUMBER steht und
RowSource auf dem Wert FIELDNUMBER, so werden in der aktuellen
Position Feldnamen wie A1, A2, B1 usw. gesucht und in die Tabelle
eingetragen.
```

```
Beispiel für Options:

SEPARATOR=_

Dann werden Felder wie A_1, A_2, B_1 usw. gesucht.
```

ClearS

Der angegebene String wird von allen Leerzeichen, Zeilenumbrüchen, Tabulatoren und allen Zeichen bereinigt, die in einem File-Namen nicht vorkommen dürfen.

Syntax

ClearS(“PN“)

ConcatS

ConcatS ist eine Funktion, welche die Werte aller Platzhalter aneinander hängt und den Ergebniswert zurückgibt.

Syntax

ConcatS(PN1, PN2, ..., PNn)

```
Beispiel:

Concats(Feld1, “\n, $global.Feld3, SUB(Feld5, “2“, “3“))
```

ConvertVarsInFile

Die Funktion ConvertVarsInFile ist der Austausch von Variablen über eine gewichtete Konvertierungstabelle, in der mehrere Inputwerte zu einem Outputwert konvertiert werden können. Da in der Tabelle mit Wildcards gearbeitet werden kann, sind Doppeldeutigkeiten nicht zu vermeiden. Der Algorithmus wichtet alle Treffervektoren so, dass der Vektor mit der genauesten Angabe bzw. der Vektor mit der dominantesten Angabe gewinnt. Der Algorithmus bestimmt erst einmal, ob ein Eintrag in der Konvertierungsdatei ein Treffer ist und welche Wichte die Aussage besitzt. Zur Ermittlung der Wichte werden alle InputWerte mit einer Wichtezahl versehen: der Zweierpotenz der Nummer des Inputwertes:

InputWert0=1; InputWert1=2; InputWert3=4; InputWert4=8 usw.

Trifft der InputWert exakt mit dem Eintrag aus der Konvertierungsdatei überein, so wird die Wichte des InputWertes zur Wichte des Vektors hinzugezählt. Wird der InputWert über eine Wildcard gültig, so wird die Wichte nicht zur Wichte des Vektors addiert. Inputvariablen, die leer sind, fallen aus dem Ermitteln der Treffer und Wichte heraus. Sollen diese in die Wichteberechnung mit einfließen, ohne die Treffersuche zu beeinträchtigen, so müssen diese Inputwerte auf * gesetzt werden.

Beispiel:

Suchvektor

InputWert0 (Mandant)	InputWert1 (PC)	InputWert2 (Person)
110	WST03	Müller

Vektoren in der Konvertierungsdatei

InputWert0 (Mandant)	InputWert1 (PC)	InputWert2 (Person)	Treffer	Wichte
120	WST01	Müller	0	
110	*	*	1	1
110	WST03	*	1	3
110	*	Müller	1	5
110	WST03	Müller	1	7
*	*	*	1	0
*	WST03	*	1	2
*	*	Müller	1	4
*	WST03	Müller	1	6

Die Tabelle der Konvertierungsdatei besteht aus einem nicht Treffer und 8 Treffervektoren. Dabei hat jeder Treffervektor eine andere Wichte. Der Algorithmus würde den Vektor

110	WST03	Müller	1	7

heraussuchen, da dieser die größte Wichte hat.

Die Tabelle in der Konvertierungsdatei besteht aus einer Liste von Vektoren mit dem folgenden Aufbau:

OutputWert;InputWert1[;InputWert2[;InputWert3[;...]]]

Dabei ist der OutputWert entweder ein fester Wert oder ein Link auf eine Variable. Der OutputWert, der mit dem Zeichen @ beginnt und dann von einer Zahl gefolgt wird, verkörpert einen Link auf den InputWert mit der entsprechenden Nummer. Dabei ist der erste InputWert der InputWert0 und wird als @0 angegeben. Alle folgenden InputWerte haben die entsprechend größeren Nummern beginnend vom InputWert0.

Die Tabellen-Datei kann mit oder ohne Pfad angegeben werden. Erfolgt die Angabe ohne Pfad, so werden der Reihenfolge nach folgende Pfade durchsucht:

1. Arbeitsverzeichnis
2. Ini-Pfad
3. Alle Ressource-Pfade, die mit -afp -alp usw. definiert wurden

Syntax

ConvertVarsInFile (PN1,PN2[,PNn])

PN1: FileName der Tabelle
PN2: erster Inputwert
PNn: weitere Inputwerte

Beispiel:

```
Calc {
 DPDVFREI=CONVERTVARSINFILE("MANDANT.TBL",MANDANT,TYPE,DEV)
}
```

CopyFields

Die Funktion CopyFields wird zum Kopieren von Feldern einer Position in eine andere Position benutzt. Alle Felder der Quellposition werden in die Zielposition kopiert. Die Funktion hat zwei Parameter:


Syntax

CopyFields (PN1,PN2)

PN1: FromPath (Quellposition)
PN2: ToPath (Zielposition)

Eine leere Pfadangabe meint die aktuelle Position. Das Präfix $global. wird unterstützt.

DirectoryDelete

Erzeugt das angegebenen Verzeichnis. Ließ sich das Verzeichnis anlegen, so wird 1 zurück gegeben, anderenfalls 0


Syntax

DirectoryDelete (PN1)

PN1: Name des Verzeichnisses

DirectoryDelete

Löscht die angegebenen Verzeichnis. Ließ sich das Verzeichnis löschen, so wird 1 zurück gegeben, anderenfalls 0

Syntax

DirectoryDelete (PN1)

PN1: Name des Verzeichnisses

DivideI

DivideI teilt die IntergerZahl von PN1 (Dividend) durch die Integerzahl von PN2 (Divisor).

Syntax

DivideI (PN1, PN2)

DivideF

DivideF teilt die reelle Zahl von PN1 durch die reelle Zahl von PN2.

Syntax

DivideF (PN1,PN2,PN3,PN4,PN5,PN6)

PN1: Diviend (als lokalisierter und formatierter String)
PN2: Divisor (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)
PN5: Picture-Format in dem das Ergebnis dargestellt werden soll (optional, Standard InPicture)
PN6: Unit (Einheit), die beim Formatieren des Ergebnisses verwendet wird (optional, Standard ist leer)

```
Beispiel:

DIVIDEF ("16,5","3,0","de_DE")
Das Ergebnis ist 5,5
```

DPDVFrei

Diese Funktion wird ausführlich im Handbuch DV-Freimachung beschrieben.

EmptyS

EmptyS gibt an, ob Zeichen in dem String stehen.

 Syntax

EmptyS(PN)

```
Beispiel:

Emptys(“Hallo“)   Rückgabewert ist 0.
Emptys(“”)        Rückgabewert ist 1.
```

Exit

Exit beendet das Programm definiert mit dem Fehler -900. Exit hat 0 bis n Parameter, die als Fehlertext mit ausgegeben werden.

FieldEntryExists

FieldEntryExits überprüft, ob ein Feld im Baum vorhanden ist. Der einzige Übergabeparameter ist der Feldname inkl. Pfadangabe und Vorkommen. Zurückgegeben wird der Wert 1 oder 0; 1 wenn das Feld existiert, 0 wenn nicht.

 Syntax

FieldEntryExists (PN1)

PN1: FeldName inkl. Pfad und Vorkommen

```
Beispiel:
```

```
FIELDENTRYEXISTS(“POS1.FIELDDESC[1]“)
```

FormatF

FormatF formatiert eine reelle Zahl (Gleitkommazahl), die im Computer-Zahlen-Format übergeben wird für ein bestimmtes Locale mit einem bestimmten Picture.
Computer-Zahlen-Format meint keine Sonderzeichen nur ein Punkt als Trennzeichen zwischen dem ganzen und dem gebrochenen Teil. Die Vorzeichen + und – können der Zahl vorangestellt sein.


Syntax

FormatF (PN1,PN2,PN3,PN4)

PN1: Zahl im Computer-Zahlen-Format
PN2: Locale (optional, Standard ist das Default Locale)
PN3: Picture-Format in dem das Ergebnis dargestellt werden soll (optional, Standard InPicture)
PN4: Unit (Einheit), die beim Formatieren des Ergebnisses verwendet wird (optional, Standard ist leer)

```
Beispiel:

FORMATF(“-54321.3“,“de_DE“,“zzz,zz9.99s“)
Ergebnis 54.321,30-
```

FileCopy

Kopiert eine Datei. Ließ sich die Datei kopieren, so wird 1 zurück gegeben, anderenfalls 0


Syntax

FileCopy (PN1,PN2)

PN1: FileName der Quell-Datei
PN2: FileName der Ziel-Datei

FileDelete

Löscht die angegebenen Datei. Ließ sich die Datei löschen, so wird 1 zurück gegeben, anderenfalls 0

Syntax

FileDelete (PN1)

PN1: FileName der Datei

FileExists

Prüft ob eine Datei vorhanden ist. Rückgabe ist der Wert 1 wenn die Datei vorhanden ist oder 0 wenn nicht.

Syntax

FileExists (PN1)

PN1: FileName der Datei

FileMove

Verschiebt eine Datei. Ließ sich die Datei verschieben, so wird 1 zurück gegeben, anderenfalls 0

Syntax

FileMove (PN1,PN2)

PN1: FileName der Quell-Datei
PN2: FileName der Ziel-Datei

FillS

FillS ist ein Stringfüller. Strings, die unter der angegebenen Länge sind, werden mit dem ebenfalls angegebenen Füllzeichen gefüllt. Dabei kann das Füllzeichen entweder links oder rechts angesetzt werden. Wie bei der Standardsyntax kann jeder PN-Wert als Variable, FixText oder Funktion angegeben werden.

Syntax

FillS(PN1, PN2, PN3, PN4)

PN1: ist der zu füllende Wert (Zahl oder String)
PN2: ist das Füllzeichen (das erste Zeichen des Strings zählt)
PN3: Orientierung des Befüllens (LEFT oder RIGHT)
PN4: die Sollgröße des Wertes

```
Beispiel:

FILLS(JF_KUNDNUM, “0“, “LEFT“, “8“)
```

GetCityFromAddress

GetCityFromAddress liest aus einer postalischen Adresse die Stadt heraus und gibt diese als Rückgabewert aus.

Syntax

GetCityFromAddress(PN1)

PN1: Adresse

GetCountryFromAddress

GetCountryFromAddress liest aus einer postalischen Adresse das Land heraus und gibt dieses als Rückgabewert aus.

Syntax

GetCountryFromAddress(PN1)

PN1: Adresse

GetHouseNoFromAddress

GetHouseNoFromAddress liest aus einer postalischen Adresse die Hausnummer heraus und gibt diese als Rückgabewert aus.

Syntax

GetHouseNoFromAddress(PN1)

PN1: Adresse

GetPagesInDocument

Gibt die Anzahl der Seiten zurück, die bereits im Dokument erstellt wurden.

 Syntax

GetPagesInDocument ([PN1])

PN1: Name eines Druckers (optional)

Mit PN1 kann optional ein Drucker angegeben werden. Ist ein Drucker angegeben, so wird die Berechnung der Seiten auf WorkItems beschränkt, die diesen Drucker definiert haben.

GetPagesInPDFFile

Gibt die Anzahl der Seiten zurück, die der angegebene PDF-File entält.

 Syntax

GetPagesInPDFFile (PN1)

PN1: FileName des PDF-Files

Existiert der PDF-File nicht oder tritt ein anderer Fehler auf, so wird 0 zurückgegeben.

GetSheetsInDocument

Gibt die Anzahl der Blätter zurück, die bereits im Dokument erstellt wurden.

 Syntax

GetSheetsInDocument ([PN1])

PN1: Name eines Druckers (optional)

Mit PN1 kann optional ein Drucker angegeben werden. Ist ein Drucker angegeben, so wird die Berechnung der Blätter auf WorkItems beschränkt, die diesen Drucker definiert haben.

GetStreetFromAddress

GetStreetFromAddress liest aus einer postalischen Adresse die Straße heraus und gibt diese als Rückgabewert aus.

Syntax

GetStreetFromAddress(PN1)

PN1: Adresse

GetZIPFromAddress

GetZIPFromAddress liest aus einer postalischen Adresse die Postleitzahl heraus und gibt diese als Rückgabewert aus. Uber einen zweiten optionalen Parameter kann eine kommaseparierte Liste von Landesbezeichnungen angegeben werden, für die eine Postleitzahl extrahiert wird.

Syntax

GetZIPFromAddress(PN1,[PN2])

PN1: Adresse
PN2: kommaseparierte Liste von Landesbezeichnungen (optional)

GetSubstitute

GetSubstitute arbeitet mit dem Basisobjekt Substitute zusammen. Der GetSubstitute-Befehl ermittelt den Tabellennamen und den auszutauschenden Wert aus seinen beiden Übergabeparametern, sucht die Substitute-Tabelle und ermittelt den Austauschwert.

Syntax

GetSubstitute (PN1,PN2)

PN1: Name der Substitute-Tabelle
PN2: zu konvertierender Wert

Beispiel:

```
Calc {
    KONTONUMMER=GETSUBSTITUTE("KONTONUMMERN",STXRDI_MANDT)
}
```

GroupExists

Die Funktion testet, ob die angegebene Group existiert. Wird die Gruppe gefunden, so liefert die Funktion den Wert „1“ zurück. Andernfalls den Wert „0“.

Syntax

GroupExists (PN1)

```
Beispiel:

WITHITEMS=GROUPEXISTS(“ITEM_LINE”)
```

HexStrings

Für die Werteübergabe an Barcodes wird oft ein HexString benötigt, der hexadezimal anzugeben ist. Da es viele unterschiedliche Notationen für HexStrings gibt, ist es unter Umständen notwendig, diese in das Format zu überführen, welches intern in der Software erwartet wird. HexString beseitigt aus dem Input-String alle Zeichen außer 0-9 und a-f. Dann werden Byte-Pärchen von Hexzeichen gebildet und mit '\x' vorangestellt in die Ausgabe gegeben

Syntax

HexString (PN1)

```
Beispiel:

HEX=HEXSTRING(“&x09af 8A 9F”)

Das Ergebnis ist: “\x09\xaf\0x8a\0x9f“
```

If

Wenn PN1 den Wert 1 zurückliefert, dann ist der Rückgabewert von If gleich dem Wert PN2. Wenn PN1 den Wert 0 zurückliefert, dann ist der Rückgabewert von If gleich dem Wert PN3.


Syntax

If(PN1,PN2,PN3)

IndexS

IndexS sucht in einem String das Vorkommen eines Matchstrings und gibt die Position des ersten Zeichens des Matchstrings im String zurück. Dabei bedeutet -1 nicht gefunden und 0 ist das erste Zeichen.


Syntax

IndexS(PN1,PN2,PN3)

PN1: String, in dem gesucht wird
PN2: Match- oder Suchstring
PN3: Nummer des Vorkommens des Matchstrings im Suchstring (optional, Standardwert 1)

IsEqualS

Liefert den Wert 1 zurück, wenn der String PN1 gleich dem String PN2 ist. Andernfalls ist der Rückgabewert 0.


Syntax

IsEqualS(PN1,PN2)

IsGreaterS

Liefert den Wert 1 zurück, wenn der String PN1 größer ist als der String PN2. Andernfalls ist der Rückgabewert 0.

Syntax

IsGreaterS (PN1,PN2)

IsLessS

Liefert den Wert 1 zurück, wenn der String PN1 kleiner ist als der String PN2. Andernfalls ist der Rückgabewert 0.

Syntax

IsLessS(PN1,PN2)

IsEqualF

Liefert den Wert 1 zurück, wenn die reelle Zahl PN1 gleich der realen Zahl PN2 ist. Andernfalls ist der Rückgabewert 0.

Syntax

IsEqualF (PN1,PN2,PN3,PN4)

PN1: 1. Zahl (als lokalisierter und formatierter String)
PN2: 2. Zahl (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)

IsGreaterF

Liefert den Wert 1 zurück, wenn die reelle Zahl PN1 größer ist als die reelle Zahl PN2. Andernfalls ist der Rückgabewert 0.

Syntax

IsGreaterF(PN1,PN2,PN3,PN4)

PN1: 1. Zahl (als lokalisierter und formatierter String)
PN2: 2. Zahl (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)

IsLessF

Liefert den Wert 1 zurück, wenn die reelle Zahl PN1 kleiner ist als die reelle Zahl PN2. Andernfalls ist der Rückgabewert 0.

Syntax

IsLessF(PN1,PN2,PN3,PN4)

PN1: 1. Zahl (als lokalisierter und formatierter String)
PN2: 2. Zahl (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)

IsEqualI

Liefert den Wert 1 zurück, wenn die ganze Zahl PN1 gleich der ganzen Zahl PN2 ist.
Andernfalls ist der Rückgabewert 0.

Syntax

IsEqualI(PN1,PN2)

IsGreaterI

Liefert den Wert 1 zurück, wenn die ganze Zahl PN1 größer ist als die ganze Zahl PN2. Andernfalls ist der Rückgabewert 0.

Syntax

IsGreaterI(PN1,PN2)

IsLessI

Liefert den Wert 1 zurück, wenn die ganze Zahl PN1 kleiner ist als die ganze Zahl PN2. Andernfalls ist der Rückgabewert 0.

Syntax

IsLessI(PN1,PN2)

IsNullS

Liefert den Wert 1 zurück, wenn der String PN1 kein Zeichen enthält. Andernfalls ist der Rückgabewert 0.


Syntax

IsNullS(PN1)

IsNullF

Liefert den Wert 1 zurück, wenn die reelle Zahl PN1 den Wert 0,00 hat. Andernfalls ist der Rückgabewert 0.


Syntax

IsNullF(PN1,PN2,PN3)

PN1: Zahl (als lokalisierter und formatierter String)
PN2: Locale (optional, Standard ist das Default Locale)
PN3: InPicture Picture-Format in dem die Zahl formatiert ist (optional, Standard Default Picture)

IsNullI

Liefert den Wert 1 zurück, wenn die ganze Zahl PN1 den Wert 0 hat. Andernfalls ist der Rückgabewert 0.


Syntax

IsNullI(PN1)

LengthS

LengthS gibt die Länge eines Strings zurück.

Syntax

LengthS (PN)

```
Beispiel:

LENGTHS(“Hallo“)
Der Rückgabewert ist 5.
```

MoveFields

Die Funktion MoveFields wird zum Verschieben von Feldern einer Position in eine andere Position benutzt. Alle Felder der Quellposition werden in die Zielposition bewegt. Die Funktion hat zwei Parameter:

Syntax

MoveFields (PN1,PN2)

PN1: FromPath (Quellposition)
PN2: ToPath (Zielposition)

Eine leere Pfadangabe meint die aktuelle Position. Das Präfix $global. wird unterstützt.

ModuloI

ModuloI realisiert die Modulo-Rechnung für Integerwerte. PN1 wird durch PN2 dividiert. Das Ergebnis ist der ganzzahlige Rest der Division von PN1 durch PN2.


Syntax

ModuloI(PN1, PN2)

Modulo10R

Modulo10R realisiert die Modulo-Rechnung für Integerwerte nach dem Verfahren Modulo 10 Rekursiv. Dieses Verfahren wird wird z. B. von Banken verwendet.


Syntax

Modulo10R(PN1)

MultiplyI

MultiplyI multipliziert die Integerwerte der Parameter 2 bis n zum Parameter 1.


Syntax

MultiplyI (PN1, PN2,..., PNn)

MultiplyF

MultiplyF multipliziert zwei reelle Zahlen (Gleitkommazahlen) zueinander.

 Syntax

MultiplyF (PN1,PN2,PN3,PN4,PN5,PN6)

PN1: 1. Faktor (als lokalisierter und formatierter String)
PN2: 2. Faktor (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)
PN5: Picture-Format in dem das Ergebnis dargestellt werden soll (optional, Standard InPicture)
PN6: Unit (Einheit), die beim Formatieren des Ergebnisses verwendet wird (optional, Standard ist leer)

```
Beispiel:

MULTIPLYF("2,5","5,0","de_DE")
Das Ergebnis ist 12,5
```

Nand

Verknüpft alle PNs nach den Regeln des logischen Nand.

 Syntax

Nand(PN1,...,PNn)

Nor

Verknüpft alle PNs nach den Regeln des Logischen Nor.

 Syntax

Nor(PN1,...,PNn)

Not

Negiert das Ergebnis von PN1


Syntax

Not(PN1) oder **!**(PN1)

NormalizeF

NormalizeF wandelt eine lokalisierte und formatierte reelle Zahl (Gleitkommazahl) in das Computer-Zahlen-Format. Die Erkennung arbeitet auf Grundlage eines bestimmtes Locale mit einem bestimmten Picture.
Computer-Zahlen-Format meint keine Sonderzeichen nur ein Punkt als Trennzeichen zwischen dem ganzen und dem gebrochenen Teil. Die Vorzeichen + und – können der Zahl vorangestellt sein.


Syntax

NormalizeF (PN1,PN2,PN3,PN4)

PN1: Zahl im Computer-Zahlen-Format
PN2: Locale (optional, Standard ist das Default Locale)
PN3: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)
PN4: Name des Feldes, in das die Unit geschrieben werden soll

```
Beispiel:

NORMALIZEF(“-76.543,21“,“de_DE“,“szzzz9.99“)
Ergebnis ist -76543.21
```

NowS

Zum Einfügen von Datums- und Zeitangaben wird die Funktion NowS im Calc-Objekt verwendet. NowS ermittelt sowohl die aktuelle Uhrzeit als auch das aktuelle Datum und gibt diese als formatierten String zurück.

Es existiert ein optionaler Übergabeparameter, der den Formatierungsstring enthält. Die Formatierungstechnik wird über Platzhalter realisiert. Dabei stehen folgende Platzhalter für folgenden Inhalt:

Die Uhrzeit

Syntax	Beschreibung	Beispiel
$HOUR$	Stunde, in der die Datei erstellt wurde.	“15“
MIN	Minute, in der die Datei erstellt wurde.	“23“
SEC	Sekunde, in der die Datei erstellt wurde.	“10“

Die Tagesangabe

Syntax	Beschreibung	Beispiel
DD	der Tag als zweistellige Zahl mit Punkt	“17“
WD	der Tag als zweistellige Zahl, Zuweisung erfolgt aufsteigend: 0=So, 1=Mo, ...	“0“
DAY	der englische Tagname als Text	“Monday“
TAG	der deutsche Tagname als Text	“Montag“

Die Monatsangabe

Syntax	Beschreibung	Beispiel
MM	Monatsangabe als zweistellige Zahl.	“03“
$MONTH$	Monatsname in Englisch als Text.	“October“
$MONAT$	Monatsname in Deutsch als Text.	“Oktober“

Die Jahresangabe

Syntax	Beschreibung	Beispiel
$YYYY$	vierstellige Jahreszahl.	“2001“
YY	zweistellige Jahreszahl .	“01“

Y	einstellige Jahreszahl.	"1"

Wenn kein Parameter angegeben wird, ist dies gleichbedeutend mit folgenden Standard-Parametern:

```
Calc {
  JETZT=NOW("$DAY$ $DD$.$MM$.$YYYY$ $HOUR$:$MIN$:$SEC$")
}
```

Beispiel:

```
Calc {
   JETZT=NOWS("$DAY$ $DD$.$MM$.$YYYY$ $HOUR$:$MIN$:$SEC$")
}

Das Ergebnis der Variable JETZT könnte folgenden Inhalt haben:
Thursday 31.01.2002 08:48:12
```

Or

Verknüpft alle PNs nach den Regeln des Logischen Or.

 Syntax

Or(PN1,...,PNn)

PlaceholderS

PlaceholderS nimmt den ersten PN als BasisString und tauscht in diesem String alle %x% gegen die weiteren PNs, wobei x für die Nummer des weiteren Parameters steht.

Beispiel:

```
$global.Anrede=PLACEHOLDERS("Hallo %1% %2%, herzlich willkommen
bei uns %1% %2%","Herr","Müller")

Das Ergebnis ist: "Hallo Herr Müller, herzlich willkommen bei uns
Herr Müller".
```

PositionEntryDelete

PositionEntryDelete löscht das erste erste Vorkommen einer Position im Baum. Der einzige Übergabeparameter ist der Positionsname inkl. Pfadangabe und Vorkommen. Zurückgegeben wird der Wert 1 oder 0; 1 wenn die Position existierte und gelöscht wurde, 0 wenn nicht.

Syntax

PositionEntryDelete (PN1)

PN1: FeldName inkl. Pfad und Vorkommen

```
Beispiel:

POSITIONENTRYDELETE(“POS1.SUBPOS[1]“)
```

PositionEntryExists

PositionEntryExists überprüft, ob eine Position im Baum vorhanden ist. Der einzige Übergabeparameter ist der Positionsname inkl. Pfadangabe und Vorkommen. Zurückgegeben wird der Wert 1 oder 0; 1 wenn die Position existiert, 0 wenn nicht.

Syntax

PositionEntryExists (PN1)

PN1: FeldName inkl. Pfad und Vorkommen

```
Beispiel:

POSITIONENTRYEXISTS(“POS1.SUBPOS[1]“)
```

PositionEntryValue

PositionEntryValue sucht alle Positionen unter dem angegebenen Positionsnamen. Innerhalb der Positionen wird der Wert des Vergleichsfeldes ermittelt und mit dem Vergleichswert verglichen. Die erste Position, für die der Wert des Vergleichsfeldes mit dem Vergleichswert identisch ist, ist der Treffer. Innerhalb der Trefferposition wird das Rückgabefeld gesucht und zurückgegeben.

Syntax

PositionEntryValue (PN1,PN2,PN3,PN4)

PN1: Name einer Position
PN2: Vergleichsfeld innerhalb der Position
PN3: Vergleichswert fürs Vergleichsfeld
PN4: Rückgabefeld

```
Beispiel:

<POS><TYPE>0</TYPE><TEXT>Printer</TEXT></POS>
<POS><TYPE>9</TYPE><TEXT>Phone</TEXT></POS>

Result=PositionEntryValue("POS","TYPE","9","TEXT")

Ergebnis: Result hat den Wert "Phone".
```

PositionExists

PositionExists prüft das Vorhandensein einer Position im Global-Bereich. Rückgabewert ist 1 für gefunden und 0 für nicht gefunden.


Syntax

PositionExists (PN1,PN2,PN3)

PN1: Name der gesuchten Position
PN2: Name eines Feldes (optional)
PN3: Inhalt des gesuchten Feldes (Match-Code) (optional)

```
Beispiel 1:

ITEMLINEEXISTS=POSITIONEXISTS("ITEM_LINE")
Wird die Position ITEM_LINE gefunden, ist das Ergebnis 1, sonst 0

Beispiel 2:

ITEMLINEEXISTS=POSITIONEXISTS("ITEM_LINE","VKORG","20")
Wird die Position ITEM_LINE gefunden, in der es ein Feld mit Namen
VKORG gibt, das den Inhalt 20 besitzt, so ist das Ergebnis 1,
sonst 0.
```

Program

Program startet ein Programm mit der angegebenen Kommandozeile und gibt die Ausgabe von stdout zurück.


Syntax

Program (PN1[,PN2])

PN1: Ist die Kommandozeile zum Aufruf des Programmes
PN2: Arbeitsverzeichnis, in dem gearbeitet wird (optional)

```
Beispiel:

calc {
 Info=PROGRAM(PLACEHOLDERS("c:\\sqlc %1% %2%,%3%",Produkt,Mandant,LKZ))
}
```

RandomI

RandomI liefert einen Zufallswert als Integerzahl. Wird die Funktion ohne Parameter aufgerufen, so liefert sie eine positive Zufallszahl. Die Funktion kann auch wahlweise mit einem oder zwei Parametern aufgerufen werden.

Syntax

RandomI ([PN1[,PN2]])

Bei einem Parameter:

PN1: MaxValue

Für die Zufallszahl n gilt: 0 <= n <= MaxValue

Bei zwei Parametern:

PN1: MaxValue
PN2 MinValue

Für die Zufallszahl n gilt: MinValue <= n <= MaxValue
Die Funktion sortiert die Werte selbstständig richtig, sollten die Parameter vertauscht sein.

RandomS

RandomS liefert einen Zufallswert als String. Als mögliche Rückgabewerte für das Ergbnis werden alle PNs verwendet, die übergeben wurden; also PN1 bis PNn. Weiterhin ist es möglich als PN-Wert eine kommaseparierte Liste von Werten zu übergeben. Leere Elemente werden akzeptiert, so dass auch leere Werte als Ergebnis möglich sind.


Syntax

RandomS (PN1[,..,PNn])

```
Beispiel:

   RandomS("John,Paul","Mary")
Alle drei Werte John, Paul und Mary sind möglich Rückgabewerte.
```

ReadVarsFromFile

ReadVarsFromFile liest Variablen aus einem File in die globale oder lokale Tabelle.


Syntax

ReadVarsFromFile (PN1[,PN2])

PN1: FileName
PN2: Speicherplatz GLOBAL oder LOCAL (Standard ist LOCAL)

```
Beispiel:

   READVARSFROMFILE("c:\\var.tbl","GLOBAL")
```

RoundF

RoundF rundet die über PN1 angegebene reelle Zahl (Gleitkommazahlen).

Syntax

RoundF (PN1,PN2,PN3,PN4,PN5,PN6)

PN1: Quellzahl (als lokalisierter und formatierter String)
PN2: Rundungsrichtung als Zahl
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)
PN5: Picture-Format in dem das Ergebnis dargestellt werden soll (optional, Standard InPicture)
PN6: Unit (Einheit), die beim Formatieren des Ergebnisses verwendet wird (optional, Standard ist leer)

Die Rundungsrichtung wird als Zahl angegeben. Ist die Rundungsrichtung 0, so wird ab einem Restwert von 0,5 auf, sonst aber abgerundet. Ist die Rundungsrichtung kleiner als 0, so wird abgerundet. Ist die Rundungsrichtung größer als 0, so wird aufgerundet.

SepSign

SepSign dient dazu, in einem TextString bei einer Zahl mit Vorzeichen die Zahl und das Vorzeichen voneinander zu trennen.

Syntax

SepSign (PN1, PN2)

PN1: Zahl mit Vorzeichen (Zahl oder String)
PN2: Part 0=Zahlenwert 1=Vorzeichen

```
Beispiele:

SEPSIGN(KOMVD_KBETR, “50,00-“)
SEPSIGN(KOMVD_KBETR, “0“)    //liefert den Zahlenwert ohne
                                Vorzeichen
SEPSIGN(KOMVD_KBETR, “1“)    //liefert das Vorzeichen
```

SubS

Bildet vom Platzhalter PN1 einen SubString, der auf Position PN2 beginnt und PN3 Zeichen lang ist.

Syntax

SubS(PN1, PN2, PN3)

```
Beispiel:

SUBS($global.STXRDI_TDCOVTITLE,"12","5")
```

SubtracDate

SubtracDate subtrahiert von einem als Input gegebenen Tagesdatum weitere Tage und gibt als Ausgabe das errechnete Datum zurück. Dabei können die Wochenenden ignoriert werden, so dass nur die Werktage gezählt werden.

Syntax

SubtracDate(PN1,PN2,PN3)

PN1: InputDatum im Format DD.MM.YYYY oder DD.MM.YY
PN2: Anzahl Tage, die addiert werden sollen
PN3: Wochenende-Schalter mit YES/NO oder 1/0; bei YES/1 werden Tage an Wochenenden mitgezählt

```
Beispiel:

SUBTRACDATE("12.07.2002","3","NO")
Das Ergebnis ist 9.07.2002
```

SubtracI

SubtracI zieht vom Integerwert des Strings PN1 die Integerwerte aller folgenden PNs ab.

Syntax

SubtracI (PN1, PN2,..., PNn)

SubtracF

SubtracF subtrahiert zwei reelle Zahlen (Gleitkommazahlen) voneinander.

Syntax

SubtracF (PN1,PN2,PN3,PN4,PN5,PN6)

PN1: 1. Minuend (als lokalisierter und formatierter String)
PN2: 2. Minuend (als lokalisierter und formatierter String)
PN3: Locale (optional, Standard ist das Default Locale)
PN4: InPicture Picture-Format in dem die erste und die zweite Zahl formatiert sind
(optional, Standard Default Picture)
PN5: Picture-Format in dem das Ergebnis dargestellt werden soll (optional, Standard InPicture)
PN6: Unit (Einheit), die beim Formatieren des Ergebnisses verwendet wird (optional, Standard ist leer)

```
Beispiel:

SUBTRACF("5.000","500,50","de_DE")
Das Ergebnis ist 4.499,5
```

StretchS

StretchS zieht mehrere Zeilen eines Strings zu einer Zeile zusammen. Leerzeilen bleiben allerdings erhalten. Ein solcher Text kann dann mit Fields FeldName:NN wieder wortweise umgebrochen werden.

Syntax

StretchS (PN1)

RemoveAllS

RemoveAllS entfernt jedes Vorkommen der Zeichenkette PN2 im String PN1.

Syntax

RemoveAllS (PN1, PN2)

PN1: Original String
PN2: Match String

ReplaceAllS

ReplaceAllS tauscht jedes Vorkommen der Zeichenkette PN2 im String PN1 gegen PN3.

Syntax

ReplaceAllS (PN1, PN2, PN3)

PN1: Original String
PN2: Match String
PN3: Replacement

TokenS

TokenS liefert den n-ten Token aus einem String zurück, wobei 1 der erste Token ist.

Syntax

TokenS(PN1,PN2,PN3)

PN1: ist der Ursprungs-String
PN2: ist die Nummer des Tokens
PN3: ist der Separator zwischen den Token

```
Beispiel:

$global.Address ="Firma\nHeinz GmbH\nWaldweg 5\n1111 Musterdorf"
Street=TOKENS($global.Address,"3","\n")
Das Ergebnis ist: "Waldweg 5"
```

Tokenizer

Tokenizer zerteilt unter Angaben eines Delimiters einen InputString in mehrere Tokens. Ist der Delimiter ein Leerzeichen, so werden leere Tokens wegoptimiert. In allen anderen Fällen bleiben leere Tokens erhalten. Alle im Calc-Befehl angegebenen Token-Variablen werden tatsächlich gesetzt, das bedeutet, dass wenn der InputString weniger Tokens besitzt als VariablenNamen im Calc-Befehl vorkommen, dann werden die restlichen Variablen mit einem leeren String gesetzt.


Syntax

Tokenizer(PN1,PN2,PN3[,PNn])

PN1: ist der Ursprungs-String
PN2: ist das Delimiter-Zeichen
PN3-PNn: sind Variablennamen der Variablen, die die Teilstrings gesetzt bekommen

ToLowerS

ToLowerS wandelt die Großbuchstaben eines Strings in Kleinbuchstaben. Alle anderen Zeichen bleiben unverändert. Zurückgegeben wird der geänderte String.


Syntax

ToLowerS(PN1)

PN1: zu lesender String

ToUpperS

ToUpperS wandelt die Kleinbuchstaben eines Strings in Großbuchstaben. Alle anderen Zeichen bleiben unverändert. Zurückgegeben wird der geänderte String.


Syntax

ToUpperS(PN1)

PN1: zu lesender String

Trace

Wie der Name der Funktion bereits verrät, werden mit Trace Informationen ausgegeben, die den Ersteller der TCI darüber informieren, wo der OMS-ReportWriter bei seiner Arbeit vorbeikam und welche Werte dort beobachtbar sind. Trace gibt keine Werte zurück.


Syntax

Trace(PN1,PN2[,PNn])

PN1: Name des Ausgabemediums
PN2: auszuschreibende Information
PNn: auszuschreibende Information

Der Name des Ausgabemediums kann drei unterschiedliche Ausgabemedien bedienen.

OUT oder STDOUT: Ausgabe erfolgt auf STDOUT - sprich dem Bildschirm
LOG oder LOGFILE: Ausgabe erfolgt im LOGFILE, dieser wurde mit -aiiLOGFILENAME definiert; alles andere wird als FileName interpretiert.

Der FileName kann zeitabhängige Anteile besitzen, z. B. "ReportW_$YYYY$.MM.DD.trc". Zur Auflösung der zeitabhängigen Variablen wird der Calc-Befehl NowS verwendet. Die genaue Syntax der zeitabhängigen Variablen ist dort dokumentiert.

TrimS

TrimS entfernt nicht benötigte Zeichen vom Anfang und oder vom Ende eines Strings.


Syntax

TrimS(PN1,PN2,PN3) oder
TrimS(PN1,PN2) oder
TrimS(PN1)

PN1: String, aus dem die Zeichen zu entfernen sind.
PN2: Orientierung des Entfernens (BOTH, LEFT oder RIGHT), keine Angabe meint BOTH
PN3: ist das Füllzeichen (nur das erste Zeichen des Strings zählt)

```
Beispiel:

TRIMS(" Hallo Welt ","BOTH"," ")

Das Ergebnis ist dann "Hallo Welt".
```

WriteDocAsJF

WriteDocAsJF schreibt das aktuelle Dokument im JetForm/Adobe Present Format als File.


Syntax

WriteDocAsJF (PN1,PN2,PN3)

PN1: Name des Files
PN2: APPEND oder NONE soll ein bereits bestehender File überschrieben werden (optional)
PN3: CodePage UTF8 LOCAL OEM ISO-8859-1 ISO_10646_UCS_2 ISO_10646_UCS_2B ISO_10646_UCS_2L

WriteDocAsXML

WriteDocAsXML schreibt das aktuelle Dokument im XML-Format als File.

Syntax

WriteDocAsXML (PN1)

PN1: Name des Files

WriteAllGlobalVarsToFile

WriteAllGlobalVarsToFile schreibt alle globalen Variablen in einen File. Dabei kann unterschieden werden, ob dieser File upgedated oder überschrieben wird.

Syntax

WriteAllGlobalVarsToFile (PN1[,PN2])

PN1: FileName
PN2: Modus UPDATE oder REPLACE

```
Beispiel:

   WRITEALLGLOBALVARSTOFILE("c:\\var.tbl","UPDATE")
```

WriteAllLocalVarsToFile

WriteAllLocalVarsToFile schreibt alle lokalen Variablen in einen File. Dabei kann unterschieden werden, ob dieser File upgedated oder überschrieben wird.


Syntax

WriteAllLocalVarsToFile (PN1[,PN2])

PN1: FileName
PN2: Modus UPDATE oder REPLACE

```
Beispiel:

   WRITEALLLOCALVARSTOFILE("c:\\var.tbl","REPLACE")
```

WriteGlobalVarsToFile

WriteGlobalVarsToFile schreibt alle angegebenen globalen Variablen in einen File. Dabei kann unterschieden werden, ob dieser File upgedated oder überschrieben wird.


Syntax

WriteGlobalVarsToFile (PN1,PN2,PN3[,PNn])

PN1 FileName
PN2 Modus UPDATE oder REPLACE
PNn VariablenNamen

```
Beispiel:

  WRITEGLOBALVARSTOFILE("c:\\var.tbl","REPLACE","STXRDI_TDFORM","USER")
```

WriteLocalVarsToFile

WriteLocalVarsToFile schreibt alle angegebenen lokalen Variablen in einen File. Dabei kann unterschieden werden, ob dieser File upgedated oder überschrieben wird.


Syntax

WriteLocalVarsToFile (PN1,PN2,PN3[,PNn])

PN1 FileName
PN2 Modus UPDATE oder REPLACE
PNn VariablenNamen

```
Beispiel:

   WRITELOCALVARSTOFILE("c:\\var.tbl","REPLACE","POSITION","NOTIZ")
```

Recognition bzw. Rec


Verwendung

Alle Objekte, die ein Recognition-Objekt definieren, werden vor ihrer Ausführung von einem übergeordneten Objekt angefragt, ob sie sich für die aktuellen Daten für zuständig befinden. Ist die Recognition wahr, so wird das Objekt ausgeführt. Ist die Recognition falsch, muss das übergeordnete Objekt auf die Ausführung verzichten. Eine Recognition ist eine Verknüpfung von Einzelanfragen zu einem Wert 1 oder 0, wahr oder falsch. Standardwert ist wahr. Für eine Recognition gibt es zwei Syntaxvarianten: eine vereinfachte Syntax, die nur Feldinhalte abprüfen kann, und eine erweiterte Syntax, die komplexe Anweisungen ausführen kann.


Vereinfachte Syntax

Recognition Field=Value [Field=Value [...]]
bzw.
Rec Field=Value [Field!Value [...]]

Die Recognition beinhaltet eine Liste von Feld-Inhalts-Paaren. Wurde das Keyword REC definiert, dann prüft der OMS-ReportWriter, ob die Feld-Inhalts-Paare für die Kopffelder des Dokuments übereinstimmen. Als Operatoren stehen ein Gleich-Operator (=) und ein Ungleich-Operator (!) zur Wahl. Alle Prüfungen sind miteinander AND-verknüpft, so dass ein nicht zutreffendes Pärchen zum negativen Ergebnis der Recognition führt. Ist das Ergebnis negativ, wird dieses WorkItem nicht generiert. Ist die Prüfung positiv oder wurde das Keyword REC nicht definiert, wird das WorkItem ausgeführt.

Erweiterte Syntax

Recognition bzw. **Rec** {
  ...
  Calc-Befehl
  Logische Liste
  ...
}

Das Recognition-Objekt ist eine logische Liste von Calc-Befehlen und/oder logischen Listen, die von oben nach unten abgefragt werden, ob das Ergebnis wahr oder falsch ist. Die Einzelergebnisse werden dabei AND-verknüpft und bilden das Gesamtergebnis der Recognition. Die aufgerufenen Calc-Befehle sollen keine Variable als Rückgabewert definieren, dafür aber 0 oder 1 für falsch oder wahr zurückliefern. Logische Listen haben grundsätzlich denselben Aufbau wie das Recognition-Objekt und dienen dazu, einzelne logische Abfragen zu einem Gesamtergebnis zu verknüpfen. Logische Listen sind beliebig ineinander verschachtelbar.

Folgende logische Listen werden unterstützt:

And: Alle in der Liste enthaltenen Einzelwerte werden AND-verknüpft
Or: Alle in der Liste enthaltenen Einzelwerte werden OR-verknüpft
Nand: Alle in der Liste enthaltenen Einzelwerte werden AND-verknüpft und dann negiert
Nor: Alle in der Liste enthaltenen Einzelwerte werden OR-verknüpft und dann negiert

Syntax der logischen Listen:

And {
  ...
  Calc-Befehl
  Logische Liste
  ...
}
usw.

Beispiel:

```
Rec {
 ISEQUALS(CLIENT,“417“)
 OR {
  ISEQUALS(USER,“Deer“)
  ISEQUALS(USER,“Fox“)
 }
}
```

EnvelopeSortSystem

Verwendung

Das EnvelopeSortSystem beschreibt alle Werte, die im Zusammenhang mit einem Kuvertiersystem der Portoklassenberechnung bzw. der DV-Freimachung stehen.

Syntax

EnvelopeSortSystem {

Type	Value
Channel	Value
Insertion	Value
DoNotClose	Value
Divert	Value
DivertBin1	Value
...	
DivertBin3	Value
DocRef	Value
SheetWeight	Value
EnvelopeWeight	Value
Feature1	Value
...	
Feature4	Value
JobDescriptor	Value
EndOfJob	Value
OMRDesign	Value
PostalServiceProvider	Value
Client	Value
Class	Value
ZIPCode	Value
CountryCode	Value
Action	Value
ITProcessingDate	Value
PostingDate	Value
Dimension	Value
WindowDesign	Value
IndividualData	Value

PremiumAdressID Value
OrderLabel Value
}


Erklärung

Type Value

Type bezeichnet den Namen des Kuvertiersystems. Die Kuvertiersysteme sind fest hinterlegt.

Channel Value oder **Kanal** Value

Channel kann mit den Werten 1 und 2 belegt werden. Kanalbezeichnungen werden dann gebraucht, wenn der OMS-ReportWriter unterschiedliche WorkItems eines Dokuments in separate Spool-Files bringen muss. Dies wird für Endlosdruckanwendungen in Zwei-Kanal-Technik benötigt. Für Einzelblattanwendungen bleibt der Wert auf 1 stehen.

Insertion Value
InsertionName Value

FieldOrValue
Insertion ist ein Schlüsselwort, das eine Materialbeschreibung für eine Beilage definiert und kann mehrfach aufgerufen werden. Der Aufbau und die interne Struktur einer Materialbeschreibung ist im Abschnitt Material näher beschrieben. Da in einer Materialbeschreibung alle Eigenschaften eines Materials definiert werden, erzeugt ein Aufruf von Insertion auch alle Eigenschaften der Beilage, wie z.B. MaterialID, Tray, Gewicht und Größe.
Ist es gewünscht je Eigenschaften des Materials einen separaten Aufruf eines Insertion-Befehls zu schreiben, so erlaubt diese das Schlüsselwort InsertionName. Name ist dabei ein beliebiger Name, der für die Entwurfslogik angebracht scheint. Mit jedem Aufruf von InsertionName mit dem selben Name wird das Insertion-Objekt unter das unter diesem Namen angelegt wurde erweitert.

```
Beispiel:

Insertion M:BeilegerVertrieb07 N:3 F:DINA4 O:ISFOLDED L:3 W:4,2gr

 oder

Insertion1 M:BeilegerVertrieb07
Insertion1 N:3
Insertion1 F:DINA4
Insertion1 O:ISFOLDED L:3
Insertion1 W:4,2gr
```

Eine Beilage arbeitet entweder über den Tray oder über die MaterialID. Eine Beilage ohne Tray und ohne MaterialID ist nicht zulässig. Wird nur über eine MaterialID gearbeitet, so erfolgt die Zuordnung zu einem Tray automatisch.

DoNotClose Value oder **NichtBefeuchten** Value

FieldOrValue
DoNotClose ist ein Schalter mit den Werten 1 oder 0, wobei der Standardwert 0 ist. Das Signal DoNotClose wird als OMR- oder Barcodesteuerung für die Kuvertiermaschine generiert. Es kann sein, dass Ihre spezielle Kuvertierstraße das Signal nicht unterstützt.

Divert Value oder **Aussteuern** Value

FieldOrValue
Divert ist ein Schalter mit den Werten 1 oder 0, wobei der Standardwert 0 ist. Das Signal Divert bzw. Aussteuern wird als OMR- oder Barcodesteuerung für die Kuvertiermaschine generiert. Es kann sein, dass Ihre spezielle Kuvertierstraße das Signal nicht unterstützt.

DivertBin1 Value oder **AussteuernFach1** Value
...
DivertBin3 Value oder **AussteuernFach3** Value

FieldOrValue
DivertBin ist Schalter mit den Werten 1 oder 0, wobei der Standardwert 0 ist. Das Signal DivertBin definiert das Aussteuerfach funktioniert wie das Signal Divert, wobei in das Fach N der Kuvertierstraße ausgesteuert wird.

DocRef Value

FieldOrValue
Mehrkanalige Kuvertierstraßen benötigen zur Dokumentverifizierung ein eindeutiges Kennzeichen. Dieses generiert der OMS-ReportWriter entweder in eigener Verantwortung oder verwendet das unter DocRef angegebene Feld zur Bildung des eindeutigen Kennzeichens.

SheetWeight Value oder **SeitenGewicht** Value

FieldOrValue

SheetWeight definiert das Gewicht der aktuellen Seite in Gramm. Der Dezimalpunkt kann als "." oder "," angegeben werden. Diese Angabe ist sowohl in führenden als auch in folgenden WorkItems Pflicht.

EnvelopeWeight Value oder **KuvertGewicht**

EnvelopeWeightdefiniert das Gewicht des aktuellen Kuverts in Gramm. Der Dezimalpunkt kann als "." oder "," angegeben werden. Das Kuvertgewicht muss nur im führenden WorkItem angegeben werden.

Feature1 Value
...
Feature4 Value

FieldOrValue
Feature dienen zur Ansteuerung spezieller Eigenschaften von Kuvertiermaschinen.

JobDescriptor Value

FieldOrValue
JobDescriptor ist die JobNummer aus Sicht des sendenden Systems. Einige Kuvertiermaschinen benötigen dieses Feld für Rückmeldung von Dokumenten.

EndOfJob Value

FieldOrValue
Der Kuvertiermaschine wird mitgeteilt, dass das Ende des Jobs erreicht ist. Im Allgemeinen wird bei der Kuvertiermaschine das Merkmal Maschine Leerfahren angesprochen.

OMRDesign Value

FieldOrValue
OMRDesign wird nur verwendet, wenn eine Kuvertiersteuerung aufzubringen ist. OMRDesign definiert ein Design-Subformular und dessen Zielpositionen. ArchiveDesignsFirst und Siehe dazu den Abschnitt Designs.

PostalServiceProvider Value oder **PSP** Value

FieldOrValue

PostalServiceProvider beinhaltet den Namen des Postdienstleisters wie dieser in psp.ini definiert ist. Der Name DPAG ist fest vordefiniert und steht für die Deutsche Post AG und die Arbeit mit dpdv.ini. Wird kein PostalServiceProvider angegeben, so erfolgt keine Aufarbeitung für postalische Zwecke.

Client Value oder **Mandant** Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter der Mandant angegeben. Diese Angabe ist Pflicht.

Class Value oder **Sendungsart** Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter die Sendungsart angegeben. Diese Angabe ist Pflicht.

ZIPCode Value oder **PLZ** Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter die Postleitzahl angegeben. Diese Angabe ist Pflicht.

CountryCode Value oder **LKZ** Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter das Länderkennzeichen angegeben. Diese Angabe ist Pflicht. Für Deutschland sind folgende Schreibweisen möglich: " " (Leer), "D", "DE"

Action Value oder **Aktion** Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter die Aktion für die Sendungsarten InfoBrief und InfoPost definiert. Diese Angabe kann für einige Sendungsarten Pflicht sein.

ITProcessingDate Value oder **DVBearbeitungstag** Value

FieldOrValue

Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter der DV-Bearbeitungstag angegeben. Diese Angabe ist optional. Die Angabe hat im Format DD.MM.YYYY zu erfolgen.

PostingDate Value oder **Einlieferungstag** Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter der Einlieferungstag angegeben. Diese Angabe ist optional. Die Angabe hat im Format DD.MM.YYYY zu erfolgen.

Dimension Value

FieldOrValue
Wurde die DVFreimachung über einen Postdienstleister aktiviert, so wird über diesen Schalter die Dimension der Sendung angegeben. Die Angabe ist optional.

```
Beispiel:

Dimension C5
```

WindowDesign Value oder **FensterDesign** Value

FieldOrValue
WindowDesign wird nur verwendet ausgewertet, wenn eine postalische Aufarbeitung erfolgte. WindowDesign definieren ein Design-Subformular und dessen Zielpositionen. Siehe dazu den Abschnitt Designs.

IndividualData Value

FieldOrValue
Bei der DV-Freimachung mit einem DataMatrix-Barcode gibt es die Möglichkeit individuelle Daten pro Sendung mitzugeben. Mit dem Schlüsselwort IndividualData können diese aus dem Nettodaten des Dokuments heraus gesetzt werden.

PremiumAdressID Value

FieldOrValue
Die DV-Freimachung unterstützt PremiumAdress-Produkte der Deutschen Post AG. Mit dem Schlüsselwort PremiumAdressID kann auf Sendungsebene eine im PremiumAdress-Portal definierte PremiumAdressID für die Sendung gewählt werden.

OrderLabel Value oder **Buchungstext** Value

FieldOrValue
Die Anmeldung der DV-Freimachung über AM.Exchange-Nachrichten unterstützt die Angabe von Buchungstexten, die dann im AM.Portal ausgewertet werden können. Die DV-Freimachung sorgt hier dafür, dass nur Sendungen mit identischem Buchungstext in die selbe Entgeltabrechnung gelangen.

SYNTAX DER SCHNITTSTELLENKONFIGURATION

RDI-Interface

Verwendung

Das RDI-Interface ist ein SAP-Standard, mit dem Nettodruckdaten aus SAP R/3 an den OMS-ReportWriter übergeben werden können. Die Aufarbeitung des Druckes geschieht letztendlich im OMS-ReportWriter.

Das RDI-Interface des OMS-ReportWriters verarbeitet sowohl normales RDI als auch SimpleRDI. Die Erkennung der Formate geschieht automatisch. In einem RDI-Datenstrom sind Standardfelder enthalten, die den Druckauftrag beschreiben, wie z. B. das gewählte Formular, den Drucker oder den Mandant. Zusätzlich können RDI-Dokumente auch Standardfelder mit Informationen zum Archiv-System beinhalten. Der OMS-ReportWriter erkennt diese Standardfelder und wandelt sie zu globalen Feldern des entsprechenden Dokuments. Um Namensdoppeldeutigkeiten zu verhindern, erhalten diese Variablen ein Präfix, das ihre Herkunft aus der entsprechenden SAP-Tabelle kennzeichnet.

RDI-Zeile	Präfix
H	“STXRDI_“
I	“STIRDI_“
P	“STPRDI_“

Weiterhin werden aus dem RDI heraus noch zwei zusätzliche Variablen gebildet:

STXRDI_CODEPAGE SAP-CodePage des Formulars
STXRDI_LANGUAGE SAP-Sprache des Formulars

Die Aufgabe des RDI-Interface ist es, die Logik des RDI-Datenformats in die Logik der Datenstruktur des OMS-ReportWriters zu überführen. Um dem Designer des RDI-Datenstroms mehr Möglichkeiten zur Konvertierung des Datenstroms zu geben, wurden sechs zusätzliche Befehle eingeführt.

Befehl	Erläuterung
^FIELD Feldname	Alle folgenden Dateninhalte werden in ein Feld mit dem angegebenen Feld-Namen geschrieben.
^ENDFIELD	Beendet ein mit ^FIELD geöffnetes Feld.
^GLOBAL	Alle folgenden Dateninhalte werden in ein globales Feld mit dem angegebenen Feld-Namen geschrieben.
^ENDGLOBAL	Beendet ein mit ^GLOBAL geöffnetes globales Feld.
^POSITION Positionsname	Startet eine neue Position mit dem angegebenen Namen. Alle folgenden Felder werden dieser Position zugeordnet.
^ENDPOSITION	Beendet eine mit ^POSITION geöffnete Position.
^GROUP Groupname	Startet eine neue Group mit dem angegebenen Namen. Alle folgenden Positionen werden dieser Group zugeordnet.
^ENDGROUP	Beendet eine mit ^GROUP geöffnete Group.

Ein ^ENDGROUP-Befehl schließt nicht nur die Group, sondern auch eventuell noch offene ^POSITION, ^GLOBAL oder ^FIELDS. Ein ^ENDITEM schließt nicht nur die aktuelle Position, sondern auch eventuell noch offene ^GLOBAL und ^FIELDS.

Syntax

Alle Zeilen und Objekte werden über den LineReader als Preprozessor gelesen und unterstützen Includes und das Lesen aus Sections.

RDI.INI:

```
RDIAddress_Language {
  CS Postbox;c/o
  ...
}
```

RDIAddress_Language Zeile:

Sprachkennzeichen

„Text für Postfach“
„Text für CO”

```
RDIAddress_Country {
  ...
  US United States;USA;0;EN;$CITY$ $POST_CODE$
  ...
}
```

RDIAddress_Country Zeile:

Länderkennzeichen
voller internationaler Name
Länderkurzname
Binnenadresse (Domestic Address= Länderkurzname in PLZ) 1=ja 0=nein
Standardsprache
CityLineFormat Frei formatierbare Zeile für die CityLine
StreetFormat US|EU|FR|DE|NL|JP

Das CityLineFormat ist eine Zeile mit Platzhaltern zur individuellen Formatierung der CityLine.
Folgende Platzhalter werden unterstützt:

$TITLE_TEXT$ $NAME1$ $NAME2$ $NAME3$ $NAME4$ $TITLE_PERS$ $NAME_PERS$ $TITLE_COMP$ $DEPARTMENT$ $NAME_CO$ $STREET$ $HOUSE_NUM1$ $HOUSE_NUM2$ STR_SUPPL1 STR_SUPPL2 STR_SUPPL3 $CITY$ $POST_CODE$ PO_BOX PO_BOX_LOC $LOCATION$ $REGION$ $COUNTRY$ $SHORT_COUNTRY$ $FULL_COUNTRY$

StreetFormat ist eine fest im Programm implementierte Art, die Strassenzeile aufzubauen und zusätzliche Straßenzeilen zur Hauptstraßenzeile zu platzieren. Implementiert sind folgende Formate:

US	United States	STR_SUPPL1 STR_SUPPL2 HOUSE_NUM1 STREET, HOUSE_NUM2 STR_SUPPL3 LOCATION
EU	Europa	STR_SUPPL1 STR_SUPPL2 STREET HOUSE_NUM1, HOUSE_NUM2 STR_SUPPL3 LOCATION

FR	Frankreich	STR_SUPPL1 STR_SUPPL2 HOUSE_NUM1, STREET, HOUSE_NUM2 STR_SUPPL3 LOCATION
DE	Deutschland	STR_SUPPL1 STR_SUPPL2 STREET HOUSE_NUM1 / HOUSE_NUM2 STR_SUPPL3 LOCATION
NL	Niederlande	STR_SUPPL1 STR_SUPPL2 STREET HOUSE_NUM1 HOUSE_NUM2 STR_SUPPL3 LOCATION
JP	Japan	STREET STR_SUPPL1 STR_SUPPL2 STR_SUPPL3 LOCATION

optional:

RDIForms {
  …
  SAPFORMULARNAME FENSTERNAME[,FENSTERNAME[…]]
  …
}

Globale Felder, die nicht aus dem Mainfenster herkommen, müssen für manche Applikationen in eine definierte Reihenfolge gebracht werden. Aus diesem Grund kann optional im Basisobjekt RDIForms eine Fensterreihenfolge definiert werden. Die globalen Felder werden der Reihenfolge der Definitionen nach sortiert, wobei gilt, dass Felder sortierter Fenster vor Feldern nicht sortierter Fenster stehen. Ist das Mainfeld nicht mit in der Definition aufgeführt, so werden globale Felder aus dem Mainfenster immer hinten angestellt.

Beispiel:

```
RDIFORMS {
   …
   ZAWIGEBUEBE01 FOOTER, INFO1,INFO,ADDRESS,REPEAT
   …
```

```
}
```

Erklärung optional RDI

optional:

RDI {

KillEmptyFields	Value
KillEmptyGlobals	Value
KillFieldsContainsOnlySpaces	Value
KillGlobalsContainsOnlySpaces	Value
KillEmptyPositions	Value
SuppressFixTextStrings	Value
EOL	Char
InsertAddressSourceFields	Value
InsertZIPFields	Value
UniqueAddressSourceFields	Value
MainWindow	Value
IncludeWindowName	Value
IncludeWindowNameInGlobalFields	Value
TrimAllFields	Value
LineControl	Value
FloatingCityLine	Value
MaxSpaceLines	Value
ConvertPositionToField	Value
KillGunderscore	Value
ReduceRDI2SRDI	Value
IgnoreCP	Value
StandardCP	Value
StandardFromCountry	Value
StandardToCountry	Value
IgnoreLocation	Value
IgnoreDistrict	Value

}

KillEmptyFields Value

Schalter mit 0 oder 1. Standardwert 1
bestimmt, ob leere Felder zu löschen sind.

KillEmptyGlobals Value

Schalter mit 0 oder 1. Standardwert 1
bestimmt, ob leere Globals zu löschen sind.

KillFieldsContainsOnlySpaces Value

Schalter mit 0 oder 1. Standardwert 1
bestimmt, ob Felder gelöscht werden, die nur Spaces enthalten.

KillGlobalsContainsOnlySpaces
Value

Schalter mit 0 oder 1. Standardwert 1
bestimmt, ob Globals gelöscht werden, die nur Spaces enthalten.

KillEmptyPositions Value

Schalter mit 0 oder 1. Standardwert 1
bestimmt, ob leere Positionen gelöscht werden.

SuppressFixTextStrings Value

Schalter mit 0 oder 1. Standardwert 0
bestimmt, ob Felder ohne Feldnamen unterdrückt werden.

EOL Char

Buchstabe. Standardwert
Zeilenende-Kennzeichen für einen willkürlichen Zeilenumbruch.

InsertAddressSourceFields Value

Mit InsertAddressSourceFields wird festgelegt, ob die Quellfelder einer SAP RDI-Adresse mit in den Datenstrom aufgenommen werden sollen. Standardwert ist 0 (0 für aus und 1 für an).

```
Beispiel:

RDI {
   ...
   INSERTADDRESSSOURCEFIELDS 1
   ...
}
```

InsertZIPFields Value

InsertZIPFields ist ein Schaltermit den Werten 0 und 1, wobei 1 der Standardwert ist.
Ist der Schalter eingeschaltet, so werden bei einer SAP-Adresse auch die Postleitzahl (PLZ) und das Länderkennzeichen (LKZ) mit generiert.

UniqueAddressSourceFields Value

UniqueAddressSourceFields ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so werden für AddressSourceFields eindeutige Namen erzeugt.

MainWindow Value

MainWindow betitelt den Namen des SAPScript-Fensters, in dem dynamische Elemente über Seitengrenzen hinweg aufgelistet werden. Der Standardwert ist MAIN. Alle anderen Fenster werden als seitenabhängig betrachtet.

IncludeWindowName Value

IncludeWindowName ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird vor dem Namen einer jeden Position der Name des Main-Fensters separiert durch ein Underline gestellt.

IncludeWindowNameInGlobalFields Value

IncludeWindowNameInGlobalFields ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird dem Namen globaler Felder der Fenster-Name vorangestellt, wenn es sich nicht um das MainWindow handelt. Da Adressfelder bereits den Namen des Fensters enthalten können, wird bei eingeschaltetem Schalter der Fenster-Name voran und nicht wie sonst hinten angestellt.

TrimAllFields Value

Aufforderung an das RDI-Interface, auf alle Felder einen TRIMS Befehl anzuwenden. Dabei werden alle führenden und abschließenden Leerzeichen entfernt. TrimAllFields ist ein Schalter mit 0 für deaktiviert und 1 für aktiviert. 0 ist der Standardwert.

Beispiel:

```
RDI {
   ...
   TRIMALLFIELDS 0
   ...
}
```

LineControl Value

LineControl ist ein Schalter mit den Werten 1 und 0. Standardwert ist 0. Ist LineControl eingeschaltet, so interpretiert das RDI-Interface RDI-Control-Zeilen, die den Beginn oder das Ende einer Zeilen-Sektion kennzeichnen als ^POSITION und ^ENDPOSITION Befehle.

Syntax

LineControl Value

```
Beispiel:

RDI {
 ...
 LineControl 1
 ...
}

CRDI-CONTROL %%LINES-BEGIN ZISUA_TK_BARS_HEADER TEXT ST DE
CRDI-CONTROL %%LINES-END ZISUA_TK_BARS_HEADER TEXT ST DE

wird umgesetzt nach     ^POSITION ZISUA_TK_BARS_HEADER
und                     ^ENDPOSITION
```

FloatingCityLine Value

FloatingCityLine ist ein Schalter mit den Werten 1 und 0. Standardwert ist 1. FloatingCityLine wirkt auf den SAP Adress-Operator. Ist FloatingCityLine eingeschaltet, so wird die Zeile mit der Stadt so ausge-geben, dass zwischen der Stadt und der Straße (oder dem Postfach) genau soviele Zeilen platziert werden, wie unter MaxSpaceLines definiert ist. Ist FloatingCityLine ausgeschaltet, so wird die Zeile mit der Stadt so ausgegeben, dass diese unabhängig von allen anderen Angaben möglichst immer an derselben Stelle steht. Die Deutsche Post erwartet heute eine FloatingCityLine.

Syntax

FloatingCityLine Value

```
Beispiel:

  RDI {
   ...
   FloatingCityLine  1
   ...
  }
```

MaxSpaceLines Value

MaxSpaceLines gibt die maximale Anzahl von Zeilen zwischen dem oberen Teil der Adresse (Name, Straße usw.) und dem unteren Adressteil (Postleitzahl, Ort, Land usw.) an. Standardwert ist 0. Der Wert -1 bedeutet, MaxSpaceLines wurde deaktiviert. MaxSpaceLines hat nur eine Auswirkung, wenn FloatingCityLine auf 1 gesetzt, sprich aktiviert wurde.

Syntax

MaxSpaceLines Value

```
Beispiel:

  RDI {
   ...
   FloatingCityLine     1
   MaxSpaceLines        1
   ...
  }
```

ConvertPositionToField Value

ConvertPositionToField ist ein Schalter mit den Werten 0 und 1 und dem Standardwert 0. Der eingeschaltete Schalter bewirkt, dass ^POSITION Kommandos im RDI-Datenstrom in globale Felder gewandelt werden und nicht als Positions interpretiert werden. Die erzeugten Felder tragen folgenden Namen: POS__XX wobei XX der Name der gerufenen Position ist.

KillGUnderscore Value

Ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so werden alle Positionsnamen, die mit "G_" beginnen, von diesem Präfix befreit.

ReduceRDI2SRDI Value

Kommaseparierte Liste. Es gibt Fälle, in denen es sinnvoll ist, aus einem kompletten RDI-Datenstrom nur die Daten zu verwenden, die von SAP erzeugt worden wären, wenn die Ausgaben nicht über RDI, sondern SRDI erfolgt wären. Die Liste ReduceRDI2SRDI definiert, welche Formulare in der Art zu reduzieren sind. Der Wert * bedeutet dabei „alle“.

IgnoreCP Value

IgnoreCP ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Steht IgnoreCP auf 1, wird der CodePage-Befehl des RDI-Formates ignoriert. Das kann sinnvoll sein, wenn die Angaben im RDI nicht mit der tatsächlich angelieferten CodePage harmonieren. Der Schalter arbeitet eng zusammen mit dem Schlüsselwort StandardCP, mit dessen Hilfe eine CodePage willkürlich gesetzt werden kann.

StandardCP Value

StandardCP setzt willkürlich eine OMS-ReportWriter CodePage für das RDI-Interface, allerdings kann die CodePage durch den CodePage-Befehl des RDI-Formates jederzeit verändert werden. Diese Eigenschaft lässt sich jedoch mit dem Schalter IgnoreCP abschalten.

StandardFromCountry Value

StandardFromCountry ist ein optionaler Schalter. Es ist kein Standardwert definiert.
Ist in einer SAP-Adresse der Wert FROM_COUNT nicht gesetzt, so wird der Wert von StandardFromCountry verwendet. Dabei ist der Inhalt von FROM_COUNT ein zweistelliger Ländercode nach SAP Standard.

```
Beispiel:

RDI {
 ...
 StandardFromCountry DE
 ...
}
```

StandardToCountry Value

StandardToCountry ist ein optionaler Schalter. Es ist kein Standardwert definiert.

Wird in einer SAP-Adresse der Wert COUNTRY nicht gesetzt, so wird der Wert von StandardToCountry verwendet. Dabei ist der Inhalt von COUNTRY ein zweistelliger Ländercode nach SAP-Standard.

```
Beispiel:

RDI {
 ...
 StandardToCountry NL
 ...
}
```

IgnoreLocation Value

IgnoreLocation ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird die Ortsteil-Angabe der SAP-Adresse bei der Ausgabe ignoriert.

IgnoreDistrict Value

IgnoreDistrict ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird der Ortsteil (District) der SAP-Adresse, der in der Struktur City2 übergeben wird, bei der Adressaufbereitung ignoriert.

 Erklärung optional RDIAddress_Paragraphs

optional:

RDIAddress_Paragraphs {
  ...
  Paragraph Name {
    FloatingCityLine Value
    MaxSpaceLines Value
    Lines Value
  }
  ...
}

RDIAddress_Paragraphs ist eine Liste von benannten Paragraph Objekten, die je einen Paragraph beschreiben. Ein Paragraph ist eine Formatierungsoption für Adressen und definiert die drei Schlüsselwörter FloatingCityLine, MaxSpaceLines und Lines.

FloatingCityLine Value

FloatingCityLine ist ein Schalter mit den Werten 0 und 1 und funktioniert genauso wie der gleichnamige Schalter in Basis-Objekt RDI.

MaxSpaceLines Value

MaxSpaceLines definiert die maximale Anzahl an Leerzeilen zwischen dem oberen Teil des Adressblocks und der CityLine, wenn FloatingCityLine auf 1 steht.

Lines Value

Wenn Lines definiert wurde, überschreibt es die in der SAP-Adresse definierte Anzahl von Zeilen, die die Größe der Adresse bestimmt.

XML-Interface

Mit Hilfe des XML-Interfaces ist der OMS-ReportWriter in der Lage, XML-Daten zu lesen. Da XML-Dateien mehrfach verschachtelte Baumstrukturen enthalten, wurde die interne Struktur der gelesenen Daten auf eine beliebig verschachtelbare Baumstruktur erweitert. Die interne Baumstruktur gleicht XML, ist jedoch entsprechend den Aufgaben des OMS-ReportWriters abgeändert.

Ein XML-Tag kann sowohl Text als auch weitere Unter-Tags beinhalten. In der internen Datenstruktur des OMS-ReportWriters wird ein XML-Tag zu einem Feld, wenn es Text ohne einen weiteren Untertag besitzt oder ein XFP/XFA-XHTML-Textelement definiert. In allen anderen Fällen wird das XML-Tag zur Position, welches Felder oder auch weitere SubPositionen beinhalten kann. Attribute einer Position werden zu Feldern dieser Position.

In XML sind CDATA Abschnitte zurlässig, in denen keine Entities benötigt werden. Der ReportWriter interpretiert auch diese Abschnitte als Text.

Felder und SubPositionen werden im OMS-ReportWriter getrennt gespeichert. Die Reihenfolge von XML-Tags mit Text und Unter-Tags geht beim Speichern verloren, kann aber in der TCI im Basisobjekt Positions neu definiert werden.

In der Feldliste müssen im Gegensatz zu XML alle Feldnamen eindeutig sein. Durch Anhängen von Zahlen an doppelte Feldnamen realisiert ein Mechanismus im OMS-ReportWriter diese Forderung. Enden Feldnamen mit einer Ziffer, so kann es zu unerwünschten Umbenennungen kommen, wenn der gleiche Feldname ohne abschließende Ziffer mehrfach in der Tabelle vorkommt.

Alle Feldnamen werden im Gegensatz zu XML im OMS-ReportWriter in Großbuchstaben konvertiert, da Adobe Present/Central und der OMS-ReportWriter "Case-insensitiv" arbeiten.

Der oberste Tag-Name des Dokuments spielt bei der Erkennung des Datenstroms eine Rolle. Er steht als globale Variable STXXML_DOCNAME zur Verfügung.

Es besteht die Möglichkeit, XML-Files als SourceCopyFile zu schreiben. Dabei werden die Dokumente so in den SourceCopyFile geschrieben, dass mehrere Dokumente in einem XML-File untergebracht werden können.

Aus diesem Grund erzeugt der OMS-ReportWriter als erstes Tag ein Tag mit dem Namen <RW_LIST_TAG>, so dass alle anderen Dokumente eine Etage nach unten verschoben werden.
Bei erneutem Einlesen dieser Dateien muss dieses Tag in der xml.ini nicht als LIST_TAG gesetzt werden, da der OMS-ReportWriter seine eigenen Dateien am ersten Tag erkennt. Unabhängig von den Einstellungen in der xml.ini wird dann für diese Dateien das erste Tag weggelesen.

Alle Zeilen und Objekte werden über den LineReader als Preprozessor gelesen und unterstützen Includes und das Lesen aus Sections.

Syntax

xml.ini

```
XML {
  ...
  FirstTagIsOnlyListTag      Value
  XFPContentType             Value
  SuppressXFPNameSpaces      Value
  IgnoreNameSpacePrefix      Value
  ...
}
```

FirstTagIsOnlyListTag

FirstTagIsOnlyListTag ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist.
Eine XML-Datei enthält nach der XML-Spezifikation nur exakt einen Wurzel-Tag. Es muss erkannt werden, ob es sich bei dem zu lesenden XML-File um ein Einzeldokument oder eine Liste von Dokumenten handelt. Bei einem Einzeldokument ist das erste Tag des XML-Files auch das Wurzel-Tag des Dokuments. Bei einer Liste von Dokumenten ist das erste Tag die Aufzählung. Alle darunter befindlichen Tags stellen Wurzel-Tags unterschiedlicher Dokumente dar.

XFPContentType

XFPContentType ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so sucht das XML-Interface nach Tags mit dem Attribut "xfa:contentType" bzw. "xfp:contentType". Wenn der ContentType vom Typ "text/html" ist, so werden die darunterliegenden Tags als XHTML-Textbeschreibung interpretiert.

SuppressXFPNameSpaces

SuppressXFPNameSpaces ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so werden die folgenden Namespace-Informationen beim Lesen der XML-Datei übersprungen:

xfp:dataNode
xfa:dataNode
xmlns:xfp
xmlns:xfa

IgnoreNameSpacePrefix Value

IgnoreNameSpacePrefix ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so werden alle Prefixe von XMLNameSpaces ignoriert und sind nicht mehr Bestandteil von Namen.

Adobe Present-Interface (JetForm-Interface)

Das Adobe Present-Interface liest Daten in die interne Struktur des OMS-ReportWriters, die im JetForm-Field-Nominated - Format vorliegen. Dieser Datentyp wird durch den Befehl ^JOB am Anfang des Datenstroms erkannt. Der OMS-ReportWriter interpretiert nur einige JetForm-Statements.

Dazu gehören:

^JOB Jobname ... –zPRINTER -assCodePage	JOB definiert die Variablen JF_JOBNAME ; JF_JOBPRINTER ; JF_JOBTOKEN.
^FIELD FeldName	Alle folgenden Dateninhalte werden in ein Feld mit dem angegebenen Feld-Namen geschrieben.
^GLOBAL FeldName	Alle folgenden Dateninhalte werden in ein globales Feld mit dem angegebenen Feld-Namen im globalen Sektor des Dokuments geschrieben.
^SYMBOLSET CodePage	Nummer für ein JetForm-Symbolset oder Name einer OMS-ReportWriter-CodePage.

Zusätzlich zu diesen Jetform-Befehlen wurden weitere Befehle eingeführt:

^POSITION PositionName	Startet eine neue Position mit dem angegebenen Namen. Alle folgenden Felder werden dieser Position zugeordnet.
^ENDPOSITION	Beendet eine mit ^POSITION geöffnete Position.
^GROUP GroupName	Startet eine neue Group mit dem angegebenen Namen. Alle folgenden Positionen werden dieser Group zugeordnet.
^ENDGROUP	Beendet eine mit ^GROUP geöffnete Group.
^SUBPOSITION PositionsName	Startet eine SubPosition in der aktuellen Position oder im globalen Sektor.
^ENDSUBPOSITION	Beendet eine mit ^SUBPOSITION geöffnete Position.

^SUBPOSITION und ^ENDSUBPOSITION dienen der Erstellung eines strukturierten Baumes, ähnlich einem XML-Baum. Im Gegensatz zu ^POSITION verschachtelt jedes ^SUBPOSITION den Baum um eine Ebene. Während ^POSITION eine zuvor geöffnete Position automatisch abschließt, muss jede ^SUBPOSITION mit einem ^ENDSUBPOSITION abgeschlossen werden.

Die Angaben des Symbolsets erfolgt im Adobe Present Format typischerweise als Zahl. Der OMS-ReportWriter wandelt diese Zahl dann nach der folgenden Tabelle in eine CodePage-Angabe:

JetForm Symbolset	***OMS-ReportWriter CodePage***
0	CP1051
2	CP850
5	CP1252
7	DECSUPPL
10	SHIFTJIS
30	BIG5
36	GBP
37	BIG5
60	CP1250
61	CP1251
62	CP1252
63	CP1253
64	CP1254
65	CP1255
67	CP1257
68	CP1258
86	CP866
108	UTF8

Im Unterschied zu Adobes jfmerge kann im OMS-ReportWriter anstelle einer Zahl auch gleich der Name der CodePage angegeben werden.

REPORTW.INI

Alle Zeilen und Objekte werden über den LineReader als Preprozessor gelesen und unterstützen Includes und das Lesen aus Sections. Die Datei reportw.ini wird vom OMS-ReportWriter erst nach der Basis-TCI geladen. Die reportw.ini muss im aktuellen INIPATH der Basis-TCI zu finden sein.

Seriennummer

Der OMS-ReportWriter ist nur über eine gültige Seriennummer zu betreiben. In der Seriennummer sind Angaben über die Ablaufzeit der Seriennummer, die Betriebssystem-Plattform und die Berechtigung zur Verwendung einzelner Module verschlüsselt. SerNo ist ein Basis-Objekt mit gleichnamigen Schlüsselwort SerNo.

Beispiel:

```
SerNo {
  SerNo XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
}
```

JobID

Wenn die JobID (_Qxxxx oder Feld JobID) zur Erkennung des Ausgabefiles verwendet wird und gleichzeitig mehrere OMS-ReportWriter-Umgebungen (OMS-Spooler bzw. Central-Instanzen) vorhanden sind, dann ist es sinnvoll, pro OMS-ReportWriter-Umgebung einen Nummernkreis für die JobID zu definieren. Zur Definition des Nummernkreises gibt es das Objekt JobID mit den Schlüsselwörtern FROM und TO.

Beispiel:

```
JobID {
  From 10000
  To   19999
}
```

PDF

Unter diesem Objekt werden alle Einstellungen für die PDF-Ausgabe zusammengefasst. Die Unterobjekte des PDF-Objektes beschäftigen sich mit den Teilbereichen und Objekten der PDF-Ausgabe. Das Objekt ist optional.

```
Beispiel:

PDF {
  ...
}
```

Common

Das Common-Objekt definiert die allgemeine Einstellung zur Erzeugung eines PDFs. Das betrifft vor allem den Modus und die Passwörter, mit denen ein PDF angelegt wird. Das Objekt ist optional.


Erklärung

Common {
  TraceFile Value
  License Value
}

TraceFile Value

TraceFile für die Aufrufe der PDFLib. Ist ein TraceFile definiert, so wird automatisch bei jeder PDF-Erzeugung ein TraceFile geschrieben. Ist kein File angegeben, so wird auch kein TraceFile erzeugt. Standardwert ist kein TraceFile.

License Value

Sollten Sie mit einer anderen PDFLib-Lizenz als der des OMS-ReportWriter arbeiten wollen, so können Sie diese hier angeben.

Profiles

Das Profiles-Objekt definiert Eröffnungs-Profile für PDF-Dateien. Diese Profile steuern die wesentlichen Grundeigenschaften des zu eröffnenden PDFs. Die Profile stehen in engem Zusammenhang mit dem Schlüsselwort PDFProfile im WorkItem-Objekt. Ein WorkItem ist durch die Angabe von PDFProfiles in der Lage, ein spezielles Eröffnungsprofile anzuziehen, wenn es die PDF-Datei, in die es generiert wird, neu anlegt. Archiv-Dokumente ziehen grundsätzlich das Profile ARCHIVE. Existiert das geforderte Profile nicht, so wird das Profile DEFAULT verwendet. Um sicher zu gehen, dass das Profile Default existiert, legt der OMS-ReportWriter dieses selbständig an, wenn es nicht in der reportw.ini definiert wurde. Das Objekt ist optional.


Erklärung

Profiles {
  Name {

Compression	Value
Author	Value
Title	Value
Subject	Value
KeyWords	Value
MasterPassword	Value
UserPassword	Value
SystemPassword	Value
PDFVersion	Value
Linearize	Value
Optimize	Value
InMemory	Value
OutputType	Value
SuppressOutput	Value
Ticketing	Value
MetaData	Value
EmbedFonts	Value
EmbedStandardFonts	Value
StandardFonts	Value
AddStandardFonts	Value
LoadStaticPartAsMacro	Value
Permissions	Value
ViewerPreferences	Value

```
    ExternalEncoding        Value
    RotateTo                Value
    MaterialReport          Value
  }

  NameN {
    ...
  }
}
```

Compression Value

Compression ist ein Wert zwischen 0 und 9. Wobei der Wert 0 keine Komprimierung der PDF-Datei und der Wert 9 maximale Komprimierung der PDF-Datei bedeutet. Standardwert ist 9

Author Value

FieldOrValue
Der im PDF-Dokument angegebene Autor.

Title Value

FieldOrValue
Der im PDF-Dokument angegebene Titel.

Subject Value

FieldOrValue
Das im PDF-Dokument angegebene Thema.

KeyWords Value

FieldOrValue
Die im PDF-Dokument angegebenen Schlüsselworte.

MasterPassword Value

FieldOrValue und NVC
Das MasterPassword ist das Passwort des Erzeugers und Besitzers des PDF-Files. Es dient vor allem zum Passwortschutz der vergebenen Rechte.

UserPassword Value

FieldOrValue und NVC
Das UserPassword ist das Passwort, mit dem Empfänger des PDF-Files diesen öffnen und je nach Rechtevergabe bearbeiten können. Ist das UserPassword nicht gesetzt, so ist zum Öffnen des PDF-Files kein Passwort erforderlich.

SystemPassword Value

Das SystemPassword setzt ein MasterPassword, welches aber in keiner Datei des OMS-ReportWriters nachlesbar ist. Dieses Passwort ist auch PDFxOut bekannt. Schalter mit den Werten 0 und 1. Standardwert ist 0.

PDFVersion Value

Sprachversion des PDFs (1.3 bis 1.7). Standardwert ist der höchste Wert, den die PDFLib erwartet. Im Moment ist das 1.6. Für die Ausgaben von PDF/A nach dem ISO-Standard 19005-1 Level b ist folgender Wert als PDF-Version anzugeben: PDFA1B2005.

Linearize Value

Linearize ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter angeschaltet, so wird das PDF für die schnelle WEB-Anzeige linearisiert.

Optimize Value

Optimize ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird das PDF nach dem Erzeugen optimiert, indem redundante Objekte beseitigt werden.

InMemory Value

InMemory ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so erfolgen die Linearisierung und die Optimierung nicht in einem temporären File, sondern im Hauptspeicher. Dafür wird doppelt so viel Speicher benötigt, wie die PDF-Datei groß ist. Dieses Verfahren führt zu einer erheblichen Beschleunigung auf Kosten des Speicherverbrauchs.

OutputType Value

OutputType bestimmt das Dateiformat, in dem der Druckjob erzeugt wird. Zulässig sind folgende Formate: PDF, XML und MIME, wobei PDF der Standard ist. Bei der Ausgabe als MIME und XML werden alle VOL-Fields mit ins Format ausgegeben.

SuppressOutput Value

SuppressOutput ist ein Schalter mit den Werten 0 und 1. Standardwert ist 0. SuppressOutput dient zur Untersuchung von PDF-Daten. Ist der Schalter aktiv, so wird die Druck-PDF nicht erzeugt.

Ticketing Value

Ticketing ist ein Schalter mit folgenden Werten:

NONE	Ticketingvariablen werden nicht mitgegeben
ANNOTATION	Ticketingvariablen werden als Annotationen an der Seite weitergegeben
XMP	Ticketingvariablen werden als XMP-Stream am Page-Objekt mitgegeben (Standardwert)
BOTH	Variablen werden sowohl als XMP als auch als Annotation mitgegeben

PDF als Seitenbeschreibungssprache kennt keine seitenspezifische Steuerung wie Papierfacheinzug, Papierfachablage, Duplexsteuerung oder ähnliches. Um PDF um diese Fähigkeiten zu erweitern, ist ein Ticketing erforderlich, das diese Art von Variablen definiert. Standardmäßig wird das Ticketing als XMP-Stream generiert, der folgendem Namespace entspricht:

xmlns:pfPage='http://ns.profiforms.com/pfPage/content/1.0/'

MetaData Value

MetaData ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist MetaData eingeschaltet, so werden XMP-MetaDaten in das erzeugte PDF eingebettet. MetaDaten sind Daten, die ein Dokument näher beschreiben. Der OMS-ReportWriter generiert vier Arten von Einträgen:

Allgemeine Angaben
Adobe definiert eine ganze Anzahl von Standard-Einträgen im XMP. Diese betreffen vor allem Informationen zum Titel, Autor, Erstellungsprodukt usw.
Alle Arten erstellter PDFs bekommen die allgemeinen Angaben ins XMP eingebettet.

Job-Informationen
Der OMS-ReportWriter definiert für Druck-PDFs einen eigen NameSpace und bringt in diesem alle Informationen unter, die auch in der VOL-Datei zu finden sind.

Einzeldokument-Informationen
Für Einzeldokumente, die mit dem OutputType XML oder MIME erzeugt werden, ist ein eigener NameSpace definiert, der Angaben zum Einzeldokument enthält.

Archiv-Informationen
Für Archivdokumente ist ein eigener NameSpace verwendet, in dem alle ArchivIndex-Variablen des Dokuments enthalten sind.

EmbedFonts Value

Schalter mit den Werten 0 und 1. Ist EmbedFonts auf 1 gesetzt, so werden alle Fonts eingebettet, die nicht unter StandardFonts aufgeführt sind. Standardwert ist 1.

EmbedStandardFonts Value

Schalter mit den Werten 0 und 1. Ist EmbedStandardFonts auf 1 gesetzt, so werden die unter StandardFonts definierten Schriftfamilien mit ins PDF eingebettet, andernfalls nicht.
Standardwert ist 0.

StandardFonts Value

Kommaseparierte Liste von Font-Familien-Namen, die in den PDF-Anzeige- und Druckprogrammen als vorinstalliert angenommen werden. Standard ist :

Times,Times-Roman,Helvetica,Courier,Symbol,ZapfDingbats

Achtung! Die Liste darf keine Leerzeichen enthalten.

AddStandardFonts Value

Kommaseparierte Liste von Font-Familien-Namen. AddStandardFonts fügt zur Liste der StandardFonts noch weitere Fonts hinzu.

Achtung! Die Liste darf keine Leerzeichen enthalten.

LoadStaticPartAsMacro Value

LoadStaticPartAsMacro ist ein Schalter mit den Werten 1 und 0. Der Standardwert ist für alle die Dokumente 0, die als Einzeldokumente in einen File generiert werden. Das sind alle Archiv-Dokumente und alle Dokumente des OutputTyps XML und MIME. Alle anderen Dokumente haben den Standardwert 1. Ist der Schalter eingeschaltet, so werden sich wiederholende statische Elemente als xObject bzw. Makro definiert. Das spart in größeren Druckfiles enorm Speicherplatz. In kleineren Druckfiles wirkt hingegen der zusätzliche Rahmen eines Makros Speicherplatz vergrößernd.

Permissions Value

FieldOrValue
Permissions sind Rechte, die in Verbindung mit einem Master- oder Systempasswort vergeben werden können. Permissions kann entweder ein Subobjekt oder ein Schlüsselwort sein.

Ist Permissions ein Subobjekt, dann sind folgende Schlüsselwörter erlaubt:

noprint	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, so ist der Druck des Dokuments verboten.
nomodify	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, ist die Veränderung des Dokuments verboten.
nocopy	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, ist es verboten, aus dem Dokument herauszukopieren.
noannots	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, dürfen Bemerkungen weder erstellt noch verändert werden.
noforms	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, ist es verboten, Formularfelder auszufüllen.
noaccessible	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, ist es verboten, Texte und Grafiken zu extrahieren.
noassemble	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, ist das Hinzufügen von Elementen zum PDF verboten.
nohiresprint	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, ist der Druck des Dokuments in hoher Auflösung verboten.
plainmetadata	Schalter mit den Werten 0 und 1. Standardwert ist 0. Ist der Schalter eingeschaltet, so bleiben Metadaten unverschlüsselt.

Ist Permissions ein Schlüsselwort, dann ist es FieldOrValue-fähig auf die Felder von PDFFields.
Die Schlüsselwörter sind folgende:

noprint und print	Gibt an, ob der Druck des Dokuments verboten ist.
nomodify und modify	Gibt an, ob die Veränderung des Dokuments verboten ist.
nocopy und copy	Gibt an, ob es verboten ist, aus dem Dokument herauszukopieren.
noannots und annots	Gibt an, ob es verboten ist, Bemerkungen zu erstellen oder zu verändern.
noforms und forms	Gibt an, ob es verboten ist, Formularfelder auszufüllen.
noaccessible und accessible	Gibt an, ob es verboten ist, Texte und Grafiken zu extrahieren.
noassemble und assemble	Gibt an, ob das Hinzufügen von Elementen zum PDF verboten ist.
nohiresprint und	Gibt an, ob der Druck des Dokuments in hoher Auflösung verboten ist.

hiresprint plainmetadata und noplainmetadata	Gibt an, ob Metadaten unverschlüsselt bleiben.

Beispiel:

```
PDF {
 Profiles {
  DEFAULT {
    ...
    Permissions noprint nomodify
    Systempassword 1
    ...
  }
 }
}
```

ViewerPreferences Value

FieldOrValue
ViewerPreferences steuert die Einstellungen, die das PDF dem Acrobat mitteilt, um sich zu konfigurieren. ViewerPreferences kann entweder ein Subobjekt oder ein Schlüsselwort sein.

Ist ViewerPreferences ein Subobjekt, dann sind folgende Schlüsselwörter erlaubt:

centerwindow	Schalter mit den Werten mit den Werten 0 und 1, Standardwert ist 0. Gibt an, ob das Dokumentfenster am Bildschirm zentriert wird.
direction	Schalter mit den Werten L2R und R2L, wobei L2R der Standardwert ist. Definiert die Lesereihenfolge eines Dokuments, welche sich auf das Blättern in der Doppelseitenansicht auswirkt.
displaydoctitle	Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Gibt an, ob der Titel des PDFs aus der Info-Struktur in der Acrobat Fensterzeile angezeigt werden soll.
fitwindow	Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Gibt an, ob das Fenster von Acrobat an die Größe der ersten Seite angepasst wird.
hidemenubar	Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Gibt an, ob die Menüleiste von Acrobat sichtbar ist.
hidetoolbar	Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Gibt an, ob die Werkzeugleiste von Acrobat sichtbar ist.
hidewindowui	Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Gibt an, ob die Fenstersteuerelemente von Acrobat sichtbar sind.

Beispiel:

```
Profiles {
  Default {
   OutputType PDF
   ViewerPreferences {
      fitwindow 0
      hidetoolbar 1
      hidemenubar 1
      hidewindowui 1
   }
  }
}
```

Ist ViewerPreferences ein Schlüsselwort, dann ist es FieldOrValue-fähig auf die Felder von PDFFields.

Die Schlüsselwörter sind folgende:

centerwindow und nocenterwindow	Gibt an, ob das Dokumentfenster am Bildschirm zentriert wird.
directionl2r und directionr2l	Definiert die Lesereihenfolge eines Dokuments, welche sich auf das Blättern in der Doppelseitenansicht auswirkt: l2r = von links nach rechts r2l = von rechts nach links.
displaydoctitle und nodisplaydoctitle	Gibt an, ob der Titel des PDFs aus der Info-Struktur in der Acrobat Fensterzeile angezeigt werden soll.
fitwindow und nofitwindow	Gibt an, ob das Fenster von Acrobat an die Größe der ersten Seite angepasst wird.
hidemenubar und nohidemenubar	Gibt an, ob die Menüleiste von Acrobat sichtbar ist.
hidetoolbar und nohidetoolbar	Gibt an, ob die Werkzeugleiste von Acrobat sichtbar ist.
hidewindowui und nohidewindowui	Gibt an, ob die Fenstersteuerelemente von Acrobat sichtbar sind.

```
Beispiel 1:

Profiles {
  Default {
   OutputType PDF
   ViewerPreferences nofitwindow hidetoolbar hidemenubar
hidewindowui
  }
}

Beispiel 2:

Profiles {
  Default {
   OutputType PDF
   ViewerPreferences @ViewerSettings
  }
}
```

ExternalEncoding Value

FieldOrValue

ExternalEncoding definiert den Zeichensatz im PDF, wenn der Font nicht eingebettet ist. Wählbar sind folgende Zeichensätze:

unicode (Standardwert)
winansi

RotateTo Value

FieldOrValue
RotateTo bestimmen ob die Seite in der Ausgabe gedreht werden soll.
Folgende Werte sind möglich:

NONE	keine Drehung (Standardwert)
PORTRAIT	querformatige Seiten werden gegen die Uhr gedreht, hochformatige Seiten werden nicht gedreht
LANDSCAPE	hochformatige Seiten werden gegen die Uhr gedreht, querformatige Seiten werden nicht gedreht
LEFT	Drehungen gegen die Uhr
RIGHT	Drehungen mit der Uhr
HEADSTAND	Seiten werden auf den Kopf gestellt

RotateTo hat ein Pendant im WorkItem, das Vorrang hat. RotateTo wird nur ausgewertet, wenn RotateToFirst bzw. RotateToNext auf NONE stehen.

MaterialReport Value

FieldOrValue
Ist der Schalter eingeschaltet, so wird in dem Fall, dass der Debug-Modus nicht angeschaltet ist, im Inhaltsverzeichnis des erzeugten PDFs ein Report über die verwendeten Materialien ausgegeben. Im Moment stehen dort nur die Beilagen die angezogen werden. Ist der Debug-Modus eingeschaltet, so werden alle Informationen über Beilagen ohnehin mit ausgegeben. Der Schalter soll es ermöglichen, sich ein Bild davon machen zu können, wie die Bestandteile einer Sendung ausgesehen haben, die grafisch nicht im PDF enthalten sind.

Fonts

Das Fonts-Objekt definiert alle Font-spezifischen Einstellungen für die PDF-Generierung.

 Erklärung

Fonts {

StandardFonts	Value
AddStandardFonts	Value
DefaultFont	Value
EmbedStandardFonts	Value
IgnoreNonexistingResources	Value
EmbedFonts	Value
Subsetting	Value
SubsetLimit	Value
SubsetMinSize	Value
XDPPlacingMethod	Value
ExternalEncoding	Value
Replacements {	
…	
}	

}

StandardFonts Value

Kommaseparierte Liste von Font-Familien-Namen, die in den PDF-Anzeige- und Druckprogrammen als vorinstalliert angenommen werden. Standard ist :

Times,Times-Roman,Helvetica,Courier,Symbol,ZapfDingbats

Achtung! Die Liste darf keine Leerzeichen enthalten.

AddStandardFonts Value

Kommaseparierte Liste von Font-Familien-Namen. AddStandardFonts fügt zur Liste der StandardFonts noch weitere Fonts hinzu.

Achtung! Die Liste darf keine Leerzeichen enthalten.

DefaultFont Value

Wird ein Font nicht gefunden, so kann über IgnoreNonExistingResources der Standard-Font gezogen werden. Mit DefaultFont kann dieser festgelegt werden.
Standardwert ist Courier.

EmbedStandardFonts Value

Schalter mit den Werten 0 und 1. Ist EmbedStandardFonts auf 1 gesetzt, so werden die unter StandardFonts definierten Schriftfamilien mit ins PDF eingebettet, andernfalls nicht.
Standardwert ist 0.

IgnoreNonexistingResources Value

Schalter mit den Werten 0 und 1. Ist IgnoreNonexistingResources auf 1 gesetzt, so werden nicht gefundene Fonts durch Courier ersetzt, andernfalls erfolgt ein Abbruch. Standardwert ist 0.

EmbedFonts Value

Schalter mit den Werten 0 und 1. Ist EmbedFonts auf 1 gesetzt, so werden alle Fonts eingebettet, die nicht unter StandardFonts aufgeführt sind. Standardwert ist 1.

Subsetting Value

Schalter mit den Werten 0 und 1. Ist Subsetting auf 1 gesetzt, so wird von eingebetteten Fonts nur ein SubSet der verwendeten Buchstaben eingebettet. Gerade bei großen Fonts senkt das Subsetting die Größe der PDF-Datei erheblich. Standardwert ist 1.

SubsetLimit Value

Prozentwert von 0 bis 100. Werden in einem Dokument prozentual zur Gesamtzahl der Glyphen im Font mehr Glyphen verwendet, als durch den Prozentsatz vorgegeben ist, wird die Untergruppenbildung für den Font deaktiviert und dieser komplett eingebettet. Dies spart Rechenzeit, die Ausgabe wird jedoch größer. Standardwert ist 100.

SubsetMinSize Value

Kilobyte-Wert. Mit SubsetMinSize lässt sich die Untergruppenbildung für kleine Fontdateien vollständig deaktivieren. Ist die ursprüngliche Fontdatei kleiner als der Wert von SubsetMinSize in KB, wird die Untergruppenbildung für diesen Font deaktiviert. Der Standardwert beträgt 10 (10KB).

XDPPlacingMethod Value

Die Verfahren zum Setzen eines Textes innerhalb eines Textfeldes oder statischen Textes sind in den verschiedenen Programmen unterschiedlich gelöst. Da der OMS-ReportWriter Designs aus unterschiedlichen Quellen verarbeitet, ist es notwendig, auf das Setzverfahren Einfluss nehmen zu können. Dabei unterscheidet der OMS-Report-Writer zwischen 4 unterschiedlichen Verfahren:
Jetform-XFT, Adobe6-XDP, Adobe7-XDP, ReportWriter 5.2 und AXTE (Adobe XDP Text Engine).

XDPPlacingMethod ist ein Schalter mit den Werten ReportW52, Adobe6, Adobe7 und AXTE (Adobe XDP Text Engine), wobei AXTE der Standardwert ist.

REPORTW52: Verwendet nur Ascender und Descender eines Fonts. Das bedeutet, dieses Verfahren ist auch möglich, wenn kein LineGap ermittelbar ist. Die Zeilenabstände stimmen nur annäherungsweise mit denen von Designer 6 und 7 überein. Typografisch wird allerdings „vernünftig“ gesetzt, was zu einem ausgewogenen Schriftbild führt. Das Verfahren geht mittelmäßig sparsam mit dem zur Verfügung stehenden Platz um.

ADOBE6: Verwendet nur Ascender, Descender und LineGap eines Fonts. LineGap gibt es erst mit Fontreg 1.2 oder PDFLib7. Typografisch ist das Ergebnis fast perfekt. Einziges Problem ist, dass das LineGap oberhalb des Ascenders angezogen wird. Nachteil des Verfahrens ist, dass es nicht plattformneutral ist und viele Fonts fälschlicherweise kein LineGap definieren. Das Verfahren ist sehr sparsam mit Platz.

ADOBE7: Verwendet nur Ascender und Descender eines Fonts. Das bedeutet, dieses Verfahren ist auch möglich, wenn kein LineGap ermittelbar ist. Um das fehlende LineGap zu ersetzen und plattformneutral zu sein, wird eine BreakEven-Rechnung über das Leading angestellt. Das Verfahren ist sehr platzaufwändig.

AXTE: Das Setzverfahren Adobe XDP Text Engine (AXTE) verwendet nur Ascender und Descender eines Fonts. Das bedeutet, dieses Verfahren ist auch möglich, wenn kein LineGap ermittelbar ist. Um das fehlende LineGap zu ersetzen und plattformneutral zu sein, wird mit einem prozentualen LineGap gerechnet. Das Verfahren ist sehr platzaufwändig.

Replacements

Replacements ist ein Unterobjekt von Fonts und enthält eine Tabelle von Fontersetzungen.
Die Replacements Tabelle enthält dabei eine Liste von Austauschpärchen für Font-Namen.
Der erste Wert des Pärchens ist der Von-Name. Der zweite Wert ist der Zu-Name.

```
Beispiel:

Fonts {
 ...
 Replacements {
  "Myriad Pro","Arial"
  "Courier Std","Courier"
 }
 ...
}
```

ExternalEncoding Value

FieldOrValue
ExternalEncoding definiert den Zeichensatz im PDF, wenn der Font nicht eingebettet ist. Wählbar sind folgende Zeichensätze:

unicode (Standardwert)
winansi

Pattern

Das Pattern-Objekt definiert alle Pattern- bzw. Füllmuster-spezifischen Einstellungen für die PDF-Generierung.


Erklärung

Pattern {
  PatternSize Value
  PatternStrokeThickness Value
  StippleAsMixedColor Value
  StippleWeight Value
  StipplePointSize Value
}

PatternSize Value

MillimeterWert. Bestimmt die Breite und Höhe des Pattern-Quadrates. Standard ist 2,845 mm.

PatternStrokeThickness Value

MillimeterWert. Bestimmt die Strichbreite von StrokePattern. Standard ist 0,1763889 mm.

StippleAsMixedColor Value

Schalter mit den Werten 0 und 1. StipplePattern zur Erzeugung von Graustufen gelten allgemein als veraltet. Ein besserer Weg ist die Umsetzung eines StipplePattern in eine Grauschattierung und bei farbiger Gestaltung in einen MixedColor-Wert von Hintergrund- und Vordergrund-Farbe. Die Anwendung von echten StipplePattern in einem PDF kann zu Anzeige- oder Druckproblemen führen.
Standardwert ist 1.

StippleWeight Value

Prozentwert. Wenn StipplePattern als MixedColors generiert werden, so entsteht oft der Wunsch, die StippleRate mit einem Gewichtsfaktor zu versehen, mit dem man eine Aufhellung erreichen kann. Ein StippleWeight von 0 bedeutet eine resultierende StippleRate von 0, keine Stipples und ist ganz hell. Standardwert ist 100.

StipplePointSize Value

Millimeterwert. Bestimmt die Größe eines StipplePoint, wenn StipplePattern nicht nach MixedColors umgewandelt werden. Standardwert ist 0,08466 mm.

Strokes

Das Strokes-Objekt definiert alle Strokes- bzw. Strich-spezifischen Einstellungen für die PDF-Generierung.

 Erklärung

Strokes {
  StrokeThickness Value
}

StrokeThickness Value

MillimeterWert. Bestimmt die Standardbreite eines Strichs. Standard ist 0,35278.

Images

Das Images-Objekt definiert alle Image- bzw. Bild-spezifischen Einstellungen für die PDF-Generierung.

 Erklärung

Images {
  IgnoreNonexistingResources Value

```
}
```

IgnoreNonexistingResources Value

Schalter mit den Werten 0 und 1. Ist IgnoreNonexistingResources auf 1 gesetzt, so werden nicht gefundene Images ignoriert, andernfalls wird das Programm abgebrochen. Standard ist 0.

Texts

Das Texts-Objekt definiert alle Text-spezifischen Einstellungen für die PDF-Generierung.


Erklärung

Texts {
AllowInlineCommands Value
AllowFieldSubstitutions Value
TabInterval Value
LinkUnderline Value
LinkBorder Value
LinkColor Value

}

AllowInlineCommands Value

Schalter mit den Werten 0 und 1. Ist AllowInlineCommands auf 1 gesetzt, so werden in statischen Texten JetForm-InlineCommands unterstützt. Standard ist 1.

AllowFieldSubstitutions Value

Schalter mit den Werten 0 und 1. Ist AllowFieldSubstitutions auf 1 gesetzt, so werden in statischen Texten JetForm-Variablensubstitutionen unterstützt. Standard ist 1.

TabInterval Value

TabInterval setzt das Standardgröße eines Tabulators. Standardwert ist 1/2 Zoll.
Die Angabe des Wertes erfolgt wie üblich als Zahl mit einer Längeneinheit

LinkUnderline Value

LinkUnderline ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird der Link-Text unterstrichen.

LinkBorder Value

LinkBorder ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird der Link-Text durch ein Rechteck umrahmt.

LinkColor Value

LinkColor hat einen HTML-Farbwert als Parameter. Ist eine gültige Farbe gesetzt, so wird der Link-Text in der gewählten Farbe dargestellt.

```
Beispiel:

LinkColor green
LinkColor #00FF00
```

Fields

Das Fields-Objekt definiert alle Fields- bzw. Feld-spezifischen Einstellungen für die PDF-Generierung.


Erklärung

Fields {
  AllowInlineCommands Value
  AllowFieldSubstitutions Value
}

AllowInlineCommands Value

Schalter mit den Werten 0 und 1. Ist AllowInlineCommands auf 1 gesetzt, so werden in dynamischen Feldern JetForm-InlineCommands unterstützt. Standard ist 1.

AllowFieldSubstitutions Value

Schalter mit den Werten 0 und 1. Ist AllowFieldSubstitutions auf 1 gesetzt, so werden in dynamischen Feldern JetForm-Variablensubstitutionen unterstützt. Standard ist 1.

SubForms

Das SubForms-Objekt definiert alle SubForm-spezifischen Einstellungen für die PDF-Generierung.


Erklärung

SubForms {
  XDPEstimateXPositionOnCall Value
}

XDPEstimateXPositionOnCall Value

Schalter mit den Werten 0 und 1. Ist XDPEstimateXPositionOnCall auf 1 gesetzt, so wird beim Aufruf eines XDP-SubForms die X-Position des SubForms zum Zeitpunkt des Entwurfes beachtet und das SubForm um diesen Betrag nach rechts geschoben. Standard ist 0.

Substitutions

Das Substitutions-Objekt definiert die Verhaltensweise von Variablen-Substitutionen in Feldern und Texten.

Substitutions {

StepIn	Value
StepOut	Value
ReplaceAll	Value
AllowMissingStepOut	Value

}

StepIn Value

StepIn enthält eine Liste von Zeichen, die den Start einer Variablen-Substitution anzeigen.

StepOut Value

StepOut enthält eine Liste von Zeichen, die das Ende einer Variablen-Substitution anzeigen.

ReplaceAll Value

ReplaceAll ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist.
Ist der Schalter eingeschaltet, so werden auch Variablen-Substitutionen ausgeführt, für die keine zugehörige Variable gefunden werden kann. Ist der Schalter ausgeschaltet, so bleibt der komplette Text der Variablen-Substitution in der Ausgabe erhalten, wenn die darin enthaltene Variable nicht exisitiert.

AllowMissingStepOut Value

AllowMissingStepOut ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist.
Ist der Schalter eingeschaltet, so werden als StepOut auch Zeilenumbrüche und Textende akzeptiert. Wenn ReplaceAll eingeschaltet ist, dann sollte AllowMissingStepOut aus Sicherheitsgründen ausgeschaltet sein.

```
Beispiel:

Substitutions {
  StepIn @$
  StepOut .$
  ReplaceAll 1
  AllowMissingStepOut 0
}
```

PDFImports

PDFImports steuert die Import-Eigenschaften importierter PDF-Seiten.


Erklärung

PDFImports {
  AutoRotate Value
  AutoShrink Value
  Box **Value**
  NoOfOpenFiles **Value**
}

AutoRotate Value

AutoRotate ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. PDF-Importseiten werden automatisch rotiert, wenn das Seitenformat (Hoch- und Querformat) nicht übereinstimmt mit der Seite, auf die die importierte Seite platziert wird.

AutoShrink Value

AutoShrink ist ein Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist die importierte Seite größer als die Seite, auf die die importierte Seite platziert wird, dann wird bei eingeschaltetem Schalter die importierte PDF-Seite verkleinert., so dass der komplette Inhalt erhalten bleibt.

Box Value

Eine Box ist der Rand einer PDF-Seite, der die Seitengröße definiert. PDFs besitzen mehrere solcher Boxen um bestimmte Teile der Seite sichtbar oder unsichtbar zu machen. Diese Boxen können auch gleichzeitig vorhanden sein, so dass diese unterschiedliche Teile einer Seite sichtbar machen. Box bestimmt, welche der folgenden Box-Angaben zur Ermittlung der Größe einer importierten Seite verwendet werd.
Box ist ein Schalter mit folgenden Werten:

none	es wird kein bestimmter Rand gesetzt und damit die CropBox oder alternativ die Mediabox verwendet (Standard)
media	MediaBox (immer vorhanden)
crop	CropBox, falls vorhanden, sonst MediaBox
bleed	BleedBox, falls vorhanden, sonst CropBox
trim	TrimBox, falls vorhanden, sonst CropBox
art	ArtBox, falls vorhanden, sonst CropBox

NoOfOpenFiles Value

NoOfOpenFiles bestimmt die Zahl der offen gehaltenen PDF-Dateien. Standardwert ist 27.
Bei Anwendungen mit vielen gleichzeitig angelegten PDF-Output-Dateien ist es wichtig, die Anzahl der offenen PDF-Input-Dateien herabsetzen zu können. Bei Anwendungen mit vielen unterschiedlichen PDF-Input-Dateien, jedoch nur einer PDF-Output-Datei, sollte die Zahl dagegen eher erhöht werden.

Barcodes

Das Barcode-Objekt definiert Barcode-Eigenschaften und ist optional.

Erklärung

Barcodes {
  StartCast Value
  StopCast Value
  Standards {
    ...
  }
  Casts {
    ...
  }
}

StartCast Value

Der OMS-ReportWriter untersucht Objektnamen nach dem Vorkommen der StartCast-Sequenz. Ist diese vorhanden, so wird der Objektname in einen neuen Objektnamen und den Barcode-Cast-Typ aufgesplittet. Ein Beispiel finden Sie im Kapitel Barcodes. Standardwert ist „_BCCAST_".

StopCast Value

Ist in einem Objektnamen die StartCast-Sequenz vorhanden, so kann mit StopCast das Ende des Barcode-Cast-Typs angezeigt werden. Standardwert ist „".

Standards und Casts

Die Objekte Standards und Casts definieren Barcode-spezifische Eigenschaften.

Syntax

```
Barcodes {
  ...
  Casts {
    ...
    Barcode {
      siehe Barcode
    }
    ...
  }
  Standards {
    ...
    Barcode {
      siehe Barcode
    }
    ...
  }
  ...
}
```

Syntax

Barcode {

Name	Value
Type	Value
CheckDigit	Value
TextPlace	Value
Format	Value
Ratio	Value
BlackWidths	Value
WhiteWidths	Value
DataLength	Value
StartChar	Value
EndChar	Value
ErrorCorrectionLevel	Value
Rows	Value
Columns	Value
FixStrokeWidth	Value
AddOnFormat	Value
StringFormat	Value
Link	Value
DatamatrixSize	Value
MaxiCodeMode	Value
MaxiCodeUsePreamble	Value
MaxiCodePreamble	Value
MaxiCodeSCMServiceClass	Value
MaxiCodeSCMCountryCode	Value
MaxiCodeSCMPostalCode	Value
QRCodeVersion	Value
QRCodeMask	Value
AI	Value

}

Name

Name des Barcodes.

Type

OMS-ReportWriter Barcode-Typ entsprechend der Liste implementierter Barcode-Typen im Kapitel Barcodes.

CheckDigit

CheckDigit ist die Bezeichnung der Prüfsumme für den Barcode. Die meisten Barcodes haben eine zum Barcode-Typ zugehörige Prüfsumme. Diese kann aber überschrieben werden. Der Standardwert ist DEFAULT.
Die wichtigsten CheckDigit Werte sind:

DEFAULT	Der Barcode bestimmt selbst, ob NONE oder STANDARD gewählt wird
NONE	Keine Prüfziffer
STANDARD	Prüfziffer wird berechnet nach dem zugehörigen Standardverfahren

Überdies können auch andere Prüfziffernverfahren gewählt werden. Es ist aber nicht sicher, ob der gewählte Barcode-Typ die Prüfziffer darstellen kann. Weitere CheckDigit Werte sind:

MOD10 MOD43 2MOD47 DPLEIT DPIDENT 1CODE11 2CODE11 POSTNET MSI1 MSI2 PLESSEY EAN8 EAN13 UPCA UPCE EAN128 CODE128 RM4SCC PZN MOD11W7 EAN14

TextPlace

TextPlace bestimmt, ob und wo der Dateninhalt als Text dargestellt wird. Nicht jeder Barcode-Typ ist in der Lage, Text generell oder an einer bestimmten Stelle darzustellen. Standardwert ist DEFAULT. Folgende Werte sind einstellbar:

STANDARD	Der Barcode bestimmt selbst, ob und wo Text dargestellt wird
NONE	Kein Text

Weitere Werte sind:

FULL BELOW ABOVE BELOW_EMBEDDED ABOVE_EMBEDDED
BELOW_PARTIALLY_EMBEDDED ABOVE_PARTIALLY_EMBEDDED
LEFT RIGHT

Format

Format ist nur für die 2D-Barcodes PDF417, PDF417Trunc und Datamatrix von Bedeutung. Standardwert ist DEFAULT. Einstellbar sind folgende Werte:

DEFAULT UCCEAN INDUSTRY MACRO05 MACRO06

DatamatrixSize

Der Barcode-Typ Datamatrix bestimmt seine Größe anhand der Daten, die er aufnehmen muss. Die Größe kann aber auch vorgewählt werden. Standardwert ist UNDEFINED.
Einstellbar sind folgende Werte:

UNDEFINED 10x10 12x12 14x14 16x16 18x18 20x20 22x22 24x24 26x26 32x32 36x36 40x40
44x44 48x48 52x52 64x64 72x72 80x80 88x88 96x96 104x104 120x120 132x132 144x144
8x18 8x32 12x26 12x36 16x36 16x48

Ratio

Das Ratio bestimmt das Verhältnis von dicken zu dünnen Linien eines Barcodes. Im Fall von PDF417 und PDF417Trunc bestimmt es das Verhältnis der Höhe zur Breite eines Strichs. Jeder Barcode-Typ hat eine Standardratio. Das Ratio ist damit vom Typ des Barcodes abhängig. Bitte beachten Sie, dass wenn Sie mit BlackWidths und/oder WhiteWidths arbeiten und mehrere Breiten angeben, das Ratio implizit mit definiert wurde und das hier definierte Ratio nicht zur Anwendung kommen kann. Im Fall von MaxiCode bestimmt Ratio die prozentuale Größe eines Oktagons. Die Zahl 0 bedeutet dabei 0 % und die Zahl 1 bedeutet 100 % der maximalen Größe. Standardwert ist 0,75.

BlackWidths und WhiteWidths

Der OMS-ReportWriter berechnet die Strichbreiten selbständig aus dem zur Verfügung stehenden Platz und dem Ratio. Für manche Anwendungen ist das aber zu ungenau. Mit BlackWidths und WhiteWidths können die Strichbreiten für schwarze und weiße Striche exakt eingestellt werden. Beide Schlüsselwörter definieren eine Liste von Breitenangaben mit Maßeinheit. Die einzelnen Breiten sind durch Leerzeichen separiert und gehen von links nach rechts wachsend zu größeren Breiten.

Syntax

BlackWidths StrichbreiteSchmal StrichbreiteDicker StrichbreiteDick StrichbreiteGanzDick
WhiteWidths LeerbreiteSchmal LeerbreiteDicker LeerbreiteDick LeerbreiteGanzDick

```
Beispiel:

BlackWidths 1.0mm 2.0mm 3.0mm 4.0mm
WhiteWidths 0.9mm 1.8mm 2.6mm 3.4mm
```

Durch die Angabe mehrerer Strichbreiten wird das Ratio implizit mit definiert. Das hat zur Folge, dass die Einstellungen unter Ratio nicht zur Anwendung kommen. Es ist auch möglich, weniger Strichbreiten anzugeben, als der Barcode benötigt. Der OMS-ReportWriter errechnet dann die Strichbreiten aus dem Ratio der ersten beiden Striche oder Leerbreiten bzw. aus dem Ratio, das über das Schlüsselwort definiert wurde. Sind BlackWidths und WhiteWidths identisch, so reicht die Angabe eines der Schlüsselwörter.

DataLength

DataLength definiert die maximale Anzahl Zeichen im Barcode, der Rest wird abgeschnitten. Bei vielen Barcodes ist die DataLength fest vorgeschrieben. Bitte verzichten Sie bei diesen Barcodes auf die Angabe dieses Parameters. Standardwert 0 (bedeutet unbeschränkt).

StartChar und EndChar

Barcodes wie Codabar benötigen einen Start- und Ende-Buchstaben. Bitte geben Sie hier nur Werte an, die zum Barcode-Typ passen.

ErrorCorrectionLevel

Die Barcode-Typen QR-Code, PDF417 und PDF417Trunc besitzen die Fähigkeit, Informationen redundant im Barcode zu speichern. Für PDF417 und PDF417Trunc gehen die Werte 0 bis 8 zu steigender Redundanz.

Rows und Columns

Die Barcode-Typen PDF417 und PDF417Trunc berechnen die Anzahl der Zeilen und Spalten selbständig. Allerdings kann die Größe auch vorgegeben werden. Folgende Werte sind einstellbar:

Rows	3 .. 90
Columns	1 .. 38

FixStrokeWidth

FixStrokeWidth ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Der Schalter wird nur von einigen Barcode-Typen interpretiert. FixStrokeWidth sorgt dafür, dass bei 1D-Barcodes mit variabler Länge die im Design als DataLength angegebene Zahl an Stellen für die Berechnung der Einzelstrichbreiten herangezogen wird. Unterschiedliche Dateninhalte führen nicht zu unterschiedlichen Strichbreiten.

AddOnFormat

Barcodetypen, wie z. B. der Presse-Barcode (EAN13 P2) oder der ISBN-Barcode (EAN13 P5) haben neben dem Hauptbarcode noch einen zusätzlichen AddOn-Barcode. Mit dem Schalter AddOnFormat wird die Höhe des Barcodes bestimmt und festgelegt, wo der Text des AddOn-Barcodes zu platzieren ist. Folgende Werte sind möglich:

Value	***Bedeutung (Höhe; Textposition)***
FULL	Volle Barcodehöhe; Text unten
PART	Halbe obere Barcodehöhe; Text unten
FULL_TOP	Volle Barcodehöhe; Text oben
PART_TOP	Obere Barcodehöhe-Texthöhe; Text oben

StringFormat

StringFormat ist ein Formatierungsstring für die Inputdaten eines Barcodes. Ist StringFormat gesetzt, so kann der Barcode Code128 variabel zwischen den Subset A, B und C umgeschaltet werden. StringFormat ist optional.

#	Steht für nächstes Datenzeichen
&	Steht für alle übrigen Datenzeichen
^	Steht für nächste Prüfziffer
A	Schaltet zu Subset A um (nur in Code 128, EAN 128, UCC 128)
B	Schaltet zu Subset B um (nur in Code 128, EAN 128, UCC 128)
C	Schaltet zu Subset C um (nur in Code 128, EAN 128, UCC 128)

Link

Link ist Schalter mit den Werten 0 und 1, wobei 0 der Standardwert ist. Ist Link eingeschaltet, so wir über den gezeichneten Barcode im Ausgabe PDF noch eine Link-Action mit dem Inhalt des Barcode-Textes gelegt, die durch anklicken im PDF-Viewer zu der im Link angegebenen Seite springt. Der Schalter Link ist vor allem für Barcode-Typen sinnvoll, die tatsächlich einen Link auf eine Web-Adresse beinhalten können. Predestiniert hierfür sind z.B. die Barcodetypen QR-Code, Datamatrix oder PDF417 aber auch Code128 und seine Derivate.

MaxiCodeMode

Modus eines MaxiCode-Barcodes. Ein Modus legt die Kodierungsart der Daten und das Fehlerkorrekturverfahren für den Barcode fest. MaxiCode kennt 6 Modi, wovon vier implementiert sind. Standardwert ist Modus 4.

Modus	***Beschreibung***
2	SCM Numeric – Structured Carrier Message (Postal Code ausschließlich numerische Zeichen, bis zu 9 Ziffern)
3	SCM Alphanumeric – Structured Carrier Message (Postal Code kann auch alphanumerische Zeichen enthalten, bis zu 6 Zeichen)
4	numerische und alphanumerische Zeichenfolgen (mit normaler Fehlerkorrektur)
5	Full EEC – maximale Fehlerkorrektur (sicherer, aber weniger Nutzdaten möglich)

MaxiCodeUsePreamble

MaxiCodeUsePreamble ist ein Schalter mit den Werten 0 und 1 bzw. false und true, wobei 0 der Standardwert ist. Ist der Schalter eingeschaltet, so wird die Präambel-Sequenz "[)>" automatisch generiert.

MaxiCodePreamble

MaxiCodePreamble ist ein String, der eine erweiterte Präambel enthält, die dem Datenstrom vorangestellt wird.

MaxiCodeSCMServiceClass

MaxiCodeSCMServiceClass ist ein String zur Vorbelegung der "Service Class" einer Structured Carrier Message. Die erlaubten Werte gehen von "000" bis "999".

MaxiCodeSCMCountryCode

MaxiCodeSCMCountryCode ist ein String zur Vorbelegung des "Country Code" einer Structured Carrier Message. Die erlaubten Werte gehen von "000" bis "999".

MaxiCodeSCMPostalCode

MaxiCodeSCMPostalCode ist ein String zur Vorbelegung von "Postal Code" einer Structured Carrier Message. Im Modus 2 sind 9 Ziffern erlaubt. Im Modus 3 sind 6 Buchstaben erlaubt.

QRCodeVersion

QRCodeVersion beschreibt die Größe des QR-Codes und kennt folgende Werte:

leer = Version wird automatisch berechnet
21x21, 25x25, 29x29, 33x33, 37x37, 41x41, 45x45, 49x49,
53x53, 57x57, 61x61, 65x65, 69x69, 73x73, 77x77, 81x81,
85x85, 89x89, 93x93, 97x97, 101x101, 105x105, 109x109,
113x113, 117x117, 121x121, 125x125, 129x129, 133x133,
137x137, 141x141, 145x145, 149x149, 153x153, 157x157,
161x161, 165x165, 169x169, 173x173, 177x177

QRCodeMask

QRCodeMask ist die Maske, die auf einen QR-Code angewendet wird,
um eine möglichst gute Leesbarkeit zu erreichen und kennt folgende Werte:
leer = automatische Berechnung
0-7 Maske 0 bis 7 (sinnvoll bei zeitkritischen Anwendungen)

AI

AI ist der Application Identifier oder kurz AI. Dieser spielt nur beim Format Industry eine Rolle. Mögliche Werte sind zwei Ziffern oder ein Buchstabe (groß oder klein)

Locales

Das Basisobjekt Locales beinhaltet Einstellungen zu Gebietsschemaparametern. Innerhalb des Objekts Locales werden benannte Locale-Objekte aufgelistet. Ein Locale-Objekt beinhaltet Gebietsschema-parameter für geografische Einheiten. Der Name des Gebietsschemas entspricht dem Locale-Namen gemäß RFC 4646. Die Locales de_DE und en_US sind vordefiniert, können aber überschrieben werden. Locales in XDP-Formularen werden in den gleichen Strukturen gespeichert und überschreiben vorhandene Locales.

Syntax

```
Locales {
  ...
  Default Value
  Locale Name {
    ...
  }
  ...
}
```

Default Value

Default definiert das StandardLocale, mit dem das Programm arbeiten soll. Das StandardLocale ist unabhängig von der verwendeten Plattform. Standardwert ist en_US. Das StandardLocale wird auch dann verwendet, wenn in XDP-Formularen das spezielle Locale ambient gesucht wird.

Locale Name

Die genaue Beschreibung von Locale ist nicht Bestandteil dieser Dokumentation, da im XDP Locale-Definitionen auf Formularebene mitgeliefert werden und eine Definition des Locale über die reportw.ini nicht benötigt wird.

Beispiel:

```
Locales {
 Default de_DE
 Locale de_DE {
  ...
 }
 ...
}
```

DESIGNS

Designs sind Formularbestandteile, die nicht automatisch wie Seiten oder Subforms aufgerufen werden, sondern eher wie Overlays bzw. Underlays auf einen Referenzpunkt auf einer Seite gezogen werden.

Position

Ein Design wird so auf einen Referenzpunkt gezogen, dass die linke obere Ecke des Designs auf dem Referenzpunkt liegt. Es gibt drei unterschiedliche Arten der Angabe eines Referenzpunktes:

1. Feste Positionsbezeichnung
Es gibt drei feste Positionsbezeichnungen NULL, ROOT und ORIGIN. Alle drei meinen die 0-Position auf der linken oberen Ecke der Seite (x=0,0cm y=0,0cm). Unterschieden werden die drei Positionsbezeichnungen einzig danach, zu welchem Zeitpunkt das Aufbringen eines Designs durchgeführt wird. Diese Unterscheidung ist sinnvoll, wenn man bedenkt, dass eine Seite gedreht und durch einen Bundsteg verschoben werden kann und sich damit die 0-Position mitdrehen oder mitverschieben kann. Während NULL die 0-Position nach Drehung und Verschiebung der Seite ist, meint ORIGIN die 0-Position der festen realen ungedrehten und nicht verschobenen Seite. ROOT wird nach einer Rotation und vor der Verschiebung durch einen Bundsteg ausgeführt. Damit ist ROOT geeignet Designs auf eine Seite zu bringen, die zwar mit der Seite gedreht werden, aber durch einen Bundsteg nicht verschoben werden.

2. Ein Feld auf der Seite oder auf dem Subform
Wird ein Feld als Referenzpunkt angegeben, so muss dieses Feld auf der Seite oder dem Subform existieren. Das Feld wird gesucht und die linke obere Ecke des Feldes als Referenzpunkt gesetzt. Der Referenzpunkt wird wie bei NULL mit gedreht und mit verschoben.

3. Absolute Position
Der Referenzpunkt kann auch als absolute Position definiert werden. Dazu muss kein Feld auf der Seite oder dem Subform existieren. Referenzpunkt wird wie bei NULL mit gedreht und mit verschoben.

Designs können unter die Seite als Underlay oder über die Seite als Overlay gezogen werden.
Erfolgt keine spezifische Angabe darüber, ob ein Designs als Underlay oder als Overlay gezogen werden soll, so werden diese als Underlay gezogen.

In der Kombination des Zeitpunktes und der Ebene ergibt sich folgende Reihenfolge der Ausführung:

- alle ORIGIN, nicht OnTop
+ danach evtl. Drehung der Seite
- alle ROOT, nicht OnTop
+ danach evtl. Verschiebung durch den Bundsteg
- alle nicht ORIGIN und nicht ROOT, nicht OnTop
+ die Seite selbst wird erzeugt
- alle nicht ORIGIN und nicht ROOT, OnTop
+ danach evtl. Rückverschiebung durch den Bundsteg
- alle ROOT, OnTop
+ danach evtl. Rückdrehung der Seite
- alle ORIGIN, OnTop

Syntax der Position:

zu 1.) NULL, ROOT oder ORIGIN
zu 2.) Wird die Position über ein Sprungfeld definiert, so entspricht die Position dem Namen eines Sprungfeldes auf der aktuellen Seite.

zu 3.) Wird die Position über x-/y-Koordinaten definiert, so ist folgende Syntax zu verwenden:
[xPos:yPos]
Koordinatenursprung ist links oben. xPos und yPos sind Gleitkommazahlen mit optionaler Einheit. Zulässige Einheiten sind cm, mm, in und pt, wobei in der Standardwert ist.

Beispiel:

```
ROOT
MYFIELD
[2.5cm:3inch]
```

Design und Subform

Als Design lassen sich Seiten oder Subforms eines Formulars oder aber auch Seiten eines PDFs oder Images verwenden. Ist das Design eine Page oder ein Subform, so werden die Felder der Page oder des Subforms durch die lokale Datenstruktur mit ausgefüllt. Ein Designaufruf aus einem WorkItems heraus hat als lokale Datenstruktur ist immer der globale Bereich. Ein Designaufruf aus einem Subform heraus hat als lokale Datenstruktur die Daten der Position.

Syntax des SubFormNamens:

a) Befindet sich das Subformular im aktuellen Designfile, so ist der SubFormName gleich dem Namen des Subformulars.

b) Befindet sich das Subformular in einem anderen Designfile, so wird der Name des Designfiles dem Namen des Subformulars vorangestellt. Separator ist das Zeichen |.

Standardmäßig werden alle Designs auf der Seite platziert, bevor die dynamischen Daten des Formulars erzeugt werden. Damit liegen die Designs gestaltungsmäßig unter den dynamischen Daten und werden von diesen unter Umständen überschrieben. Sollen die Designs nicht unterhalb der dynamischen Daten, sondern oberhalb dieser liegen, so ist im Namen des Subformulars das Zeichen + hinten anzustellen.

```
Beispiel:

MYDESIGN|SubForm10;
SubForm5+
CI.PDF|1
```

Optionen

Weiterhin gibt es Optionen zur Verarbeitung. Hier kann einem Design mitgegeben werden, dass es nur auf der Vorderseite eines Blattes oder nur auf der Rückseite eines Blattes verwendet werden soll.

Syntax: FRONTPAGE oder F|BACKPAGE oder B

Designaufruf

Ein vollständiger Designaufruf besteht aus Position, Subform und optional den Optionen. Alle drei Bestandteile werden durch ein Komma separiert.

Syntax: POSITION,SUBFORM[,OPTION]

Sollen mehrere Designs gerufen werden, so werden alle Designs als eine durch Semikolon separierte Liste aneinander gehängt.

```
Beispiel:

NULL,MYDESIGN|SubForm10+,B;[2.5cm:3inch],CI.PDF|1
```

FELDER ADRESSIEREN

Alle Variablen eines Dokuments befinden sich unabhängig vom Input-Interface in einer Baumstruktur, die sich in drei Teilbereiche unterteilt. Der oberste Teil der Baumes enthält Variablen, die als globale Variablen immer erreichbar sind. Globale Variablen dienen auch dazu, Felder auf Positionsebene auszufüllen, wenn in dieser Position kein gleichnamiges Feld vorhanden ist.
Unter den Globalen Variablen befinden sich die Positionen, die POSITIONS, mit ihren Variablen, aber auch weitere Subpositionen. Weiterhin kennt der OMS-ReportWriter noch Gruppen, die GROUPS, wobei jede Gruppe einen Namen trägt und Positionen enthält.

Jede Abfrage auf eine Variable wird aus einem Fokus heraus gestartet. Um Felder im Baum richtig anzusprechen, ist es wichtig, den aktuellen Fokus zu beachten. Alle Funktionen auf DocRec-, DocDef-, und WorkListVariant-Ebene haben den lokalen Fokus auf den Global-Bereich des Dokuments. Funktionen auf WorkItem-Ebene haben den lokalen Fokus auf auf den Bereich, der unter DataBind angegeben wurde. Fehlt ein DataBind so ist es auch hier der Globale Bereich. Funktionen auf Positions- und SubForm-Ebene haben einen lokalen Fokus auf die Position, die gerade abgearbeitet wird.

Wird eine Variable nur über ihren Namen angesprochen, so wird diese ausschließlich mit dem lokalen Fokus gesucht. Soll ein anderer Bereich angesprochen werden, so kann dies durch ein Prefix geschehen, das eine Referenz auf einen anderen Bereich darstellt. Folgende Prefixe sind erlaubt:

$record	Referenz auf den lokalen Bereich
$global	Referenz auf den globalen Bereich
$widatabind	Referenz auf das DataBind des WorkItems
$job	Referenz auf eine Datenstruktur, die nicht im Dokument liegt und den ganzen Job über die selbe ist

Beispiel:

```
$global.STXRDI_TDFORM
```

In manchen Fällen ist es nicht bekannt, an welcher Stelle im Baum die gesuchte Variable existiert. Durch ein vorangestelltes „*." wird der OMS-ReportWriter angewiesen, die Variable vom Global-Bereich startend im Baum abwärts zu suchen. Die Suche ist erfolgreich abgeschlossen, wenn die erste Variable im Baum gefunden wurde.

Beispiel:

```
*.DOARCHIVE
```

Die Variablen lassen sich auch unter genauer Ortsangabe adressieren. Dabei enthält der Variablenname Pfad- und Vorkommensangaben.
Die Pfade geben an, in welcher Position die Variable zu finden ist:

```
Beispiel:

+---Pos1
   |
   +---Pos2
         +---Field1

Pos2.Field1 oder Pos1 sind gültig
```

Kommen Positionen oder Felder wiederholt im Baum vor, so ist es oft sinnvoll, das genaue Vorkommen mit anzugeben. Dies geschieht durch eckige Klammern und die Angabe des Vorkommens, wobei 0 das erste Vorkommen ist. Die Angabe ist optional, der Standardwert ist [0].

```
Beispiel:

+---Pos1
|
+---Pos1
|
+---Pos2
      +---Field1
      +---Field1

Pos2.Field1[0] oder Pos1[1] sind gültig
```

FIELDORVALUE-ANGABEN

Zu den wichtigsten internen Mechanismen gehören FieldOrValue-Berechnungen. Diese Mechanismen stehen mit fast allen Basisobjekten in Verbindung und sind deshalb von besonderer Bedeutung.

FieldOrValue-Angaben bringen große Variabilität in die TCI, da die Steuerung wichtiger Parameter entweder fest vorgegeben ist, aus einem Feld des Datenstroms gelesen wird oder aus anderen Feldern berechnet werden kann. FieldOrValue versetzt Sie in die Lage, wichtige Werte zur Ablaufsteuerung dynamisch aus dem Datenstrom ermitteln zu können.
Parameter, die FieldOrValue-fähig sind, können

- fix festgelegt werden

```
Bsp.: Printer Printer
```

- aus einem Kopffeld des Dokuments gelesen werden

```
Bsp.: Printer @STXRDI_TDPRINTER
```

- aus einem beliebigen Feld des Dokuments gelesen werden

```
Bsp.: Printer @*.PRINTER
```

- aus einem Kopffeld über eine Austauschtabelle gelesen werden

```
Bsp.: Printer @GETSUBSTITUTE(“PRINTERTAB“,STXRDI_TDPRINTER)
```

- aus anderen Feldern berechnet werden

```
Bsp.: Printer @CALC(CONCATS(“PRINTER_“,ARBEITSPLATZNUMMER)).
```

Substitute-Tabellen sind Konvertierungstabellen. Über eine solche Konvertierungstabelle kann ein Feld im Datenstrom gelesen werden. Ein Ziel dieser Konvertierung ist, mandantenspezifische Werte in die TCI einzupflegen, ohne das DocDef oder das WorkItem für jeden Mandanten neu anlegen zu müssen. Die

andere Variante ist das Konvertieren der Werte eines Feldes in ein Format, welches der OMS-ReportWriter versteht.

Calcs sind einfache Berechnungen mit Feldern des Dokuments. Zur Wahl stehen heute arithmetische Befehle, String-Manipulationsbefehle und Befehle wie Substitute. Die Anzahl der Befehle ist heute noch sehr gering. Erfahrungsgemäß wächst an solch offenen Stellen ein Programm am schnellsten.

INTERNATIONALISIERUNG

Zeichensätze

Der OMS-ReportWriter arbeitet intern stets mit Unicode als CodePage – genauer gesagt mit UCS-2, der 16-Bit Variante. Die vom OMS-ReportWriter unterstützten Plattformen arbeiten für Unicode-Zeichen mit unterschiedlichen Zeichensätzen. Einige verwenden UCS-2 und andere UCS-4, die 32-Bit Variante. Unabhängig von diesen Unterschieden ist die CodePage für den OMS-ReportWriter auf allen Plattformen UCS-2 und damit beschränkt auf ca. 65000 unterschiedliche Zeichen.

Alle internen Operationen basieren auf Unicode. Die Schnittstellen zur Außenwelt sind für den OMS-ReportWriter allerdings nicht immer Unicode-basiert und so ist es erforderlich, Texte und Strings anderer CodePages in oder aus UCS-2 zu konvertieren.

Eingabe

Konvertierung für die Eingabe

Die Eingabe kann Unicode-basiert oder nicht Unicode-basiert sein. Eingaben, die Unicode-basiert sind, erfolgen zumeist in den UCS-Transfer-Formaten (UTF). Wobei der Unterschied zwischen UTF16 und UCS2 im Allgemeinen vernachlässigbar ist. Der OMS-ReportWriter unterstützt folgende UCS-Transfer-Formate:

UTF8	8-Bit Transfer dem OMS-ReportWriter muss der CodePage mitgeteilt werden
UTF8+BOM	8-Bit Transfer + vorangestellte Byte-Order-Mark automatische Erkennung durch den OMS-ReportWriter
UTF16 Little Endian	16-Bit Transfer, niederwertiges Byte zuerst dem OMS-ReportWriter muss der CodePage mitgeteilt werden
UTF16 Big Endian	16-Bit Transfer, höherwertiges Byte zuerst dem OMS-ReportWriter muss der CodePage mitgeteilt werden
UTF16+BOM	16-Bit Transfer + vorangestellte Byte-Order-Mark der OMS-ReportWriter erkennt automatisch an der Byte-Order-Mark, ob es sich um UTF16 Little Endian oder UTF16 Big Endian handelt

Eingaben, die nicht Unicode-basiert sind, kommen immer aus einer spezifischen 8-Bit oder 16-Bit CodePage und müssen über einen CodePage-Konverter nach UCS-2 gewandelt werden.

Hierfür besitzt der OMS-ReportWriter zwei Mechanismen:

- Konvertieren über Betriebssystem-Funktionen (Zeichensätze local und oem)
- Konvertieren über eigene Konvertierungstabellen

Bei der Konvertierung über Betriebssystem-Funktionen wird typischerweise die CodePage local verwendet. Einzig Microsoft-Windows-basierte Betriebssysteme haben für manche Länder zwei CodePages: eine für den grafischen Teil und eine für den Teil der Eingabeaufforderung. In diesem Fall ist local die CodePage des grafischen Teils und oem die CodePage der Eingabeaufforderung. Auf allen anderen Plattformen sind die CodePages local und oem gleich.

Im OMS-ReportWriter ist ein CodePage-Konverter enthalten, der in der Lage ist, aus unterschiedlichen nicht Unicode-basierten 8-Bit und 16-Bit CodePages (Multibyte-CodePages) in die UCS-2 CodePage zu konvertieren. Es gibt sehr viele unterschiedliche nicht Unicode-basierte CodePages, so dass es nicht möglich ist, alle bekannten CodePages im CodePage-Konverter abzubilden. Viel schlimmer noch ist die Tatsache, dass ein und dieselbe CodePage bei unterschiedlichen Standardisierungs-Gremien und Herstellern auch unterschiedliche Namen trägt. Die Suche nach der richtigen CodePage kann kompliziert sein. Beim Setzen einer CodePage werden alle Underline- und Minus-Zeichen aus dem CodePage-Namen entfernt und dann eine zugehörige CodePage gesucht. Die folgende Tabelle stellt die realisierten CodePages des OMS-ReportWriters dar:

CodePage	***Name und alternative Namen***	***Beschreibung***
CP437	CP437 IBM437	MS DOS 437 Latin US
CP737	CP737 IBM737	MS DOS 737 Griechisch (437G)
CP775	CP775 IBM775	MS DOS 775 Baltisch
CP850	CP850 IBM850 PC850	MS DOS 850 Latin 1
CP852	CP852 IBM852	MS DOS 852 Latin 2
CP855	CP855 IBM855	MS DOS 855 Kyrillisch
CP857	CP857 IBM857	MS DOS 857 Türkisch
CP860	CP860 IBM860	MS DOS 860 Portugiesisch
CP861	CP861 IBM861	MS DOS 861 Isländisch
CP862	CP862 IBM862 PC862	MS DOS 862 Hebräisch
CP863	CP863 IBM863	MS DOS 863 Kanadisches Französisch
CP864	CP864 IBM864	MS DOS 864 Arabisch
CP865	CP865 IBM865	MS DOS 865 Skandinavisch
CP866	CP866 IBM866	MS DOS 866 Kyrillisch
CP869	CP869 IBM869	MS DOS 869 Griechisch Modern
CP874	CP874 WINDOWS874	MS Windows/OEM 874 Thailändisch
CP950	CP950 BIG5 BIGFIVE TRADITIONALCHINESE	Microsoft CodePage 950 Traditional Chinese oder Big 5 für Taiwan, Hong Kong und Malaysia
CP936	CP936 GBK SIMPLIFIEDCHINESEGBK	Microsoft CodePage 936

	SIMPLIFIEDCHINESE	Simplified Chinese + Euro-Zeichen GBK (Guo Biao Kuozhan)
CP949	CP949 KOREAN	Microsoft CodePage 949 Koreanisch
CP932	CP932 SHIFTJIS JAPANESE	Microsoft CodePage 932 Japanisch
CP1051	CP1051 IBM1051 ROMAN8	HP Roman-8
CP1250	CP1250 WINDOWS1250	MS Windows 1250 Osteuropa
CP1251	CP1251 WINDOWS1251	MS Windows 1251 Kyrillisch
CP1252	CP1252 WINDOWS1252	MS Windows 1252 Westeuropa
CP1253	CP1253 WINDOWS1253	MS Windows 1253 Griechisch
CP1254	CP1254 WINDOWS1254	MS Windows 1254 Türkisch
CP1255	CP1255 WINDOWS1255	MS Windows 1255 Hebräisch
CP1256	CP1256 WINDOWS1256	MS Windows 1256 Arabisch
CP1257	CP1257 WINDOWS1257	MS Windows 1257 Baltisch
CP1258	CP1258 WINDOWS1258	MS Windows/DOS 1258 Vietnamesisch
ISO-646	ISO646 CEISO646 USASCII ASCII IBM367 CP367 ISOIR6 ISO646US	7-Bit ASCII
ISO-8859-1	ISO88591 CEISO88591 ISOLATIN1 LATIN1	Latin 1 Westeuropa
ISO-8859-2	ISO88592 CEISO88592 ISOLATIN2 LATIN2	Latin 2 Osteuropa
ISO-8859-3	ISO88593 CEISO88593 ISOLATIN3 LATIN3	Latin 3 Südeuropa
ISO-8859-4	ISO88594 CEISO88594 ISOLATIN4 LATIN4	Latin 4 Nordeuropa
ISO-8859-5	ISO88595 CEISO88595 ISOLATIN5 LATIN5	Latin 5 Kyrillisch
ISO-8859-6	ISO88596 CEISO88596 ISOLATIN6 LATIN6	Latin 6 Arabisch
ISO-8859-7	ISO88597 CEISO88597 ISOLATIN7 LATIN7	Latin 7 Griechisch
ISO-8859-8	ISO88598 CEISO88598 ISOLATIN8 LATIN8	Latin 8 Hebräisch
ISO-8859-9	ISO88599 CEISO88599 ISOLATIN9 LATIN9	Latin 9 Türkisch
ISO-8859-10	ISO885910 CEISO885910 ISOLATIN10 LATIN10	Latin 10 Skandinavisch
ISO-8859-11	ISO885911 CEISO885911 ISOLATIN11 LATIN11	Latin 11 Thailändisch
DECSUPPL	DECSUPPL DECSUPPLIMENTAL	Ähnlich Latin1
KOI8R	KOI8R	Unix/Relcom Kyrillisch
LOCAL	LOCAL HOST L	Lokale CodePage (auf Windows-Systemen CodePage der grafischen Oberfläche)
OEM	OEM	Nur auf Windows-Systemen DOS-CodePage
UTF8	UTF8	Universal Codeset Transfer Format 8 Bit
UTF16	ISO10646UCS2 CEISO10646UCS2 UCS2 UTF16 UTF16	Universal Codeset Transfer Format 16 Bit mit automatischer Erkennung

		der Byte Order Mark Bei Nichterkennung wird Little Endian angenommen (UCS2 und UTF16 wird gleichgesetzt)
UTF16B	ISO10646UCS2B CEISO10646UCS2B UCS2B UTF16B UTF16B	Universal Codeset Transfer Format 16 Bit Big Endian (UCS2 und UTF16 wird gleichgesetzt)
UTF16L	ISO10646UCS2L CEISO10646UCS2L UCS2L UTF16L UTF16L	Universal Codeset Transfer Format 16 Bit Little Endian (UCS2 und UTF16 wird gleichgesetzt)

Was ist, wenn eine CodePage im OMS-ReportWriter nicht vorhanden ist?

- versuchen Sie bei der Ausgabe eine andere unterstützte CodePage auszuwählen
- benutzen Sie einen Konverter, der Ihre CodePage in ein Unicode-basiertes Format umwandelt
- fordern Sie von uns eine Erweiterung des CodePage-Konverters an

Für nicht Unicode-basierte CodePages muss dem OMS-ReportWriter die CodePage mitgeteilt werden. Er kann sie nicht automatisch erkennen.

Character- und Entity-Referenzen

Erfolgt die Eingabe über nicht Unicode-basierte CodePages, so sind natürlicherweise nicht alle Unicode-Zeichen adressierbar. Damit diese Zeichen trotzdem gewählt werden können, unterstützt der OMS-ReportWriter die so genannten Character-Referenzen, ein Verfahren, das in Auszeichnungssprachen wie SGML und HTML gebräuchlich ist. Genauer ausgedrückt werden alle numerischen Character- und Entity-Referenzen, die in HTML 4.0 definiert sind, unterstützt. Bei numerischen Character-Referenzen kann der Unicode-Wert des Zeichens in Dezimal- oder Hexadezimalschreibweise angegeben sein.

Das Ein- bzw. Ausschalten dieser Funktionalität geschieht über den Schalter ResolveEntities in den CommonSettings der TCI und bezieht sich sowohl auf die Input-Interfaces als auch auf die TCI- und INI-Dateien. Die Auflösung von numerischen Character- und Entity-Referenzen erfolgt nach der Konvertierung in die UCS-2 CodePage und ist damit unabhängig von der CodePage des Quellformats.

Beispiel:

```
&#229;   Buchstabe a mit kleinem Kreis darüber (dezimal)
&#xE5;   Buchstabe a mit kleinem Kreis darüber (hex, kleines x)
&#Xe5;   Buchstabe a mit kleinem Kreis darüber (hex, großes X)
&#x20AC; Eurozeichen (hexadezimal)
&#8364;  Eurozeichen (dezimal)
&euro;   Eurozeichen (Entity-Name)
```

Speziell für Barcode-Steuersequenzen gibt es eine Anzahl Entities, die sich aus keinem Standard herleiten. Die Namen dieser Entities und deren Funktion finden Sie im Kapitel Barcodes.

Wahl der CodePage

Im einem Durchlauf des OMS-ReportWriters wird eine ganze Anzahl an Dateien gelesen, die alle aus unterschiedlichen CodePages nach UCS-2 gewandelt werden. Einige der Datei-Formate besitzen eigene Schalter und Möglichkeiten, um die CodePage zu wählen. Für einige Datei-Formate kann die CodePage als Parameter der Kommandozeilen übergeben werden.
Die folgende Auflistung erläutert Möglichkeiten zur Einstellung der CodePage für die einzelnen Datei-Formate:

Datei-Typ	***Standard-CodePage***	***Möglichkeiten zur Änderung***
XML-Input	keine	In der Datei definiert Unterstützte CodePages: UTF-8, UCS-2, ASCII, ISO-8859-1, ISO-8859-2, ISO-8859-3, ISO-8859-4, ISO-8859-5, ISO-8859-6, ISO-8859-7, ISO-8859-8
RDI-Input	ISO-8859-1	In der Datei definiert Unterstützte CodePages siehe RDI-Interface
JFDN-Input	local oder über –ass Kommdozeilenpara.	In der Datei definierbar Unterstützte CodePages: alle + JetForm-Nummern siehe Adobe-Present Interface Angabe einer Byte-Order-Mark am Anfang der Datei
Calc-Befehle: readvarsfromfile convertvarsinfile	local	Angabe einer Byte-Order-Mark am Anfang der Datei
TCI-Datei	local	Angabe einer Byte-Order-Mark am Anfang der Datei
INI-Dateien	local	Angabe einer Byte-Order-Mark am Anfang der Datei

Ausgabe

Der OMS-ReportWriter besitzt keinerlei eigenen CodePage-Konverter für die Ausgabe von UCS-2 in eine nicht Unicode-basierte CodePage. Warum auch? Die Internationalisierung schreitet voran und das bedingt Unicode als CodePage. Trotz alledem gibt es Stellen, an denen die Ausgabe nicht Unicode-basiert sein kann. In diesen Fällen bedient sich der OMS-ReportWriter der Betriebssystem-Funktionen zur Konvertierung der CodePages. Die Ausgaben auf dem Bildschirm (stdout und stderror) erfolgen immer in der CodePage oem, die in Log-Dateien immer in der CodePage local. Alles andere wird in UTF-8 ausgegeben - VOL-Files, Sort-Files usw.

Auch Texte und Bookmarks der erzeugten PDFs werden in Unicode ausgegeben. Dass die Texte im PDF auch vollständig enthalten sind, bedingt, dass auch der gewählte Font alle vorkommenden Zeichen darstellen kann. Das ist ein komplexeres Problem als die Ansteuerung eines Zeichensatzes. Einige Font-Formate unterstützen kein Unicode. Andere Font-Formate unterstützen zwar Unicode, aber die meisten

Fonts sind nicht voll besetzt, so dass ganze Zeichenbereiche fehlen. Vollbesetzte Unicode-Fonts sind noch die Ausnahme. Haben Sie Probleme mit dem Finden des richtigen Fonts, dann wenden Sie sich bitte an uns. Neben der Möglichkeit der Font-Manipulation und -Konvertierung bieten wir auch vollbesetzte Global-Font zum Kauf an.

Gebietsschemata

Internationalisierung und Lokalisierung sind Bestandteil des XFA-Konzepts. Nahezu jedes XFA-Objekt besitzt die Fähigkeit, ein Locale, ein Gebietsschema, zu tragen. Ist in einem XFA-Objekt kein Locale definiert, so wird dieses vom übergeordneten XFA-Objekt vererbt. Das Locale ist eine Kombination aus einer geografischen (ISO 3166) und einer sprachlichen (ISO 639) Abkürzung.

Locales werden vor allem benutzt, um eine gebiets- und sprachspezifische Formatierung von numerischen Feldern und von Datums-Uhrzeitfeldern vornehmen zu können. Dazu wird bei der Ausgabe die Definition des verwendeten Locales gesucht und über ein Patternmuster in das spezifische Format konvertiert.

In XDP-Formularen wird zu jedem verwendeten Locale eine vollständige Locale-Definition im Formular mit untergebracht.

Die Patternmuster, die auf die Locales zurückgreifen, sind ebenfalls Bestandteil der XFA-Spezifikation und im Handbuch des Adobe LiveCycle-Designers beschrieben.

Reporting

Wichtigstes Kontroll-Element für die Arbeit des OMS-ReportWriters ist das Log-Protokoll. Damit dieses allgemein verständlich ist, kann bei der Reporting-Language zwischen Deutsch und Englisch gewählt werden. Der OMS-ReportWriter ermittelt die Sprache des Betriebssystems und setzt die Reporting-Language automatisch auf Deutsch, wenn dies auch die Sprache des Betriebssystems ist. In allen anderen Fällen wird Englisch als Reporting-Language gesetzt. Mit Hilfe des Kommandozeilen-Parameters -lan kann die Sprache aber auch willkürlich umgeschaltet werden.

BARCODES

Zur Berechnung von Barcode-Strichbreiten verwendet der OMS-ReportWriter die TBarcode-Lib der Firma TEC-IT Datenverarbeitung GmbH (www.tec-it.com), nicht aber für die grafische Umsetzung ins PDF.
Die Erstellung eines Barcode-Feldes oder eines statischen Barcode-Textes ist Aufgabe des XDP-Designers. Der OMS-ReportWriter liest die Informationen aus dem XDP-File und führt die Berechnung und die grafische Darstellung im PDF automatisch aus. Eine spezielle Konfiguration im OMS-ReportWriter ist nicht notwendig.

Trotzdem gibt es in der reportw.ini unter Barcodes eine ganze kleine Welt an Einstellungen zur Barcode-Konfiguration. Warum?

1. Der OMS-ReportWriter kennt mehr oder andere Barcode-Typen als Adobe XDP
2. Pro Barcode-Typ sind mehr Einstellungen möglich als im Adobe XDP

Die Barcode-Angaben in der reportw.ini sind optional und nur notwendig, wenn das automatische Ergebnis nicht zufrieden stellt.

Wenn der vom OMS-ReportWriter angezogene Barcode-Typ nicht stimmt oder verändert werden soll, so ist das Barcode Type Casting der richtige Weg. Es gibt zwei Ausprägungen des Barcode Type Castings:

Casting beliebiger Felder aus XDP:

Wenn nicht ein Barcode-Typ gecasted werden soll, sondern nur ein einziges Feld, dann sollte über den Feldnamen ein Cast ausgeführt werden. Der Gedankenansatz dazu ist recht einfach. Der OMS-ReportWriter sucht in Objektnamen nach einer Barcode Cast Sequenz. Findet er diese, dann wird der Objektname in zwei Teile geteilt: dem neuen Objektnamen und der Barcode Typ Bezeichnung.

```
Beispiel:

Ihr Feld trägt den Namen POST1_BCCAST_DATAMATRIX und die Start
Cast Sequenz ist wie folgt definiert:


Barcodes {
 ...
 StartCast _BCCAST_
 ...
}

Dann wird der Feldname wie folgt aufgelöst:

Neuer Feldname: POST1
Barcode Typ:    DATAMATRIX

Auch eine StopCast Sequenz ist definierbar, dann geht der neue
Feldname nach diese Sequenz weiter.
```

Das Objekt wird beim Einlesen des XDPs umbenannt und der Typ auf den gelesenen Barcode-Type gesetzt.

Auch die Grundeigenschaften von Barcodes lassen sich verändern. Das betrifft die Strichbreiten, das Ratio und viele andere Barcode-spezifische Parameter. Der OMS-ReportWriter unterscheidet zwar dabei zwischen Cast- und Standard-Barcodes, die Einstellungsmöglichkeiten sind aber identisch. Der OMS-ReportWriter trennt Cast- und Standard-Barcodes nach ihrer Verwendung.

Standard-Barcodes sind generelle Barcode-Typen, wie „2 of 5 Interleaved“, „Code 128“, „PDF417“ oder „Datamatrix“. Der OMS-ReportWriter hat für alle diese Basis-Typen Standard-Einstellungen definiert. Ändern Sie bei Bedarf unter Barcodes/Standards diese Standard-Einstellung.

Cast-Barcodes sind spezielle Ausprägungen der Standard-Barcodes für bestimmte Zwecke. Zum Beispiel wollen Sie den Standard-Barcode Datamatrix nicht ändern, da dieser in vielen Formularen vorkommt. In einem Formular benötigen Sie den Datamatrix-Barcode nicht in der quadratischen Standard-Ausprägung, sondern in der rechteckigen. Dann legen Sie einen neuen Barcode unter Barcodes/Casts an. Dieser könnte den Namen „DATAMATRIX16X48“ tragen. Angesprochen wird der Barcode dann über einen der beiden Cast-Mechanismen, wie sie oben beschrieben sind.

Die Auflistung der möglichen Parameter für Casts und Standards finden Sie unter reportw.ini Barcodes.

Folgende Barcodes sind im OMS-ReportWriter verfügbar:

Barcode Name	***OMS-ReportWriter Barcode-Type***
Code 11	Code11
Code 2 of 5 Standard	2of5
Code 2 of 5 Interleaved	2of5IL
Code 2 of 5 IATA	2of5IATA
Code 2 of 5 Matrix	2of5M
Code 2 of 5 Data Logic	2of5DL
Code 2 of 5 Industrial	2of5IND
Code 3 of 9	3of9
Code 3 of 9 ASCII	3of9A
Code 93	9of3
Code 93 ASCII	9of3A
Code 128	Code128
Code 128 (CharSet A)	Code128a
Code 128 (CharSet B)	Code128b
Code 128 (CharSet C)	Code128c
Deutsche Post Leitcode	DPLeit
Deutsche Post Identcode	DPIdent
Plessey Code	Plessey
MSI Code	MSI
LOGMARS	Logmars
SSCC18 (Serial Shipping Container Code)	SSCC18
Codabar	Codabar
Datamatrix	Datamatrix
PDF417	PDF417
PDF417 Truncated	PDF417Trunc
EAN 128	EAN128
UCC 128	UCC128
EAN 8	EAN8
EAN 8 P2	EAN8P2
EAN 8 P5	EAN8P5
EAN 13	EAN13
EAN 13 P2	EAN13P2
EAN 13 P5	EAN13P5
UPC 12	UPC12
UPCA	UPCA
UPCA P2	UPCAP2
UPCA P5	UPCAP5
UPCE	UPCE
UPCE P2	UPCEP2

UPCE P5	UPCEP5
MaxiCode	MaxiCode
QR-Code	QRCode

Speziell für die Barcode-Steuerung definiert der OMS-ReportWriter spezielle Entity-Referenzen:

Entity-Referenz	***Wert***	***Bedeutung***
&newline;	x0A	Zeilenumbruch
&cr;	x0D	Wagenrücklauf
&linefeed;	x0A	Zeilenvorschub
&lf;	x0A	Zeilenvorschub
&formfeed;	x0C	Seitenvorschub
&ff;	x0C	Seitenvorschub
&bell;	x07	Glocke (Alarm)
&backspace;	x08	Zeichen zurück
&bs;	x08	Zeichen zurück
&tab;	x09	Horizontaler Tabulator
&ht;	x09	Horizontaler Tabulator
&vt;	x0B	Vertikaler Tabulator
&FNC1;	xD2	FNC1 für die Barcode-Typen: Code 128, EAN128, UCC128, 2D Codes
&FNC2;	xD3	FNC2 für die Barcode-Typen: Code 128, EAN128, UCC128
&FNC3;	xD4	FNC3 für die Barcode-Typen: Code 128, EAN128, UCC128
&FNC4;	xD5	FNC4 für die Barcode-Typen: Code 128, EAN128, UCC128
&DC1;	x11	DC1 für die Barcode-Typen: Code93, Code93Ext
&DC2;	x12	DC2 für die Barcode-Typen: Code93, Code93Ext
&DC3;	x13	DC3 für die Barcode-Typen: Code93, Code93Ext
&DC4;	x14	DC3 für die Barcode-Typen: Code93, Code93Ext
&Rs;	x1E	Rs für den Barcode-Typ MaxiCode (Mode 2,3 SCM)
&Gs;	x1D	Gs für den Barcode-Typ MaxiCode (Mode 2,3 SCM)
&Eot;	x04	Eot für den Barcode-Typ MaxiCode (Mode 2,3 SCM)
&CSWITCH;	xE000	ColorSwitch für Pharmacode
&FNC;	xE001	FNC1 oder Gs; UCC/EAN Feldseparator
&ECI;	xE002	ECI (Extended Character Interpretation) für die Barcode-Typen MaxiCode, Data Matrix und QR Code. Wird verwendet zum Umschalten zwischen unterschiedlichen CodePages (multiple character sets).
&ECIB;	xE003	ECI Begin, eröffnet eine Verschachtelungsebene
&ECIE;	xE004	ECI End, schließt eine Verschachtelungsebene, QR Code
&GLI;	xE005	GLI (Global Language Identifier), ähnlich wie ECI nur im PDF417
&Esc;	x1B	Escape-Zeichen

FONTS

Fast alle Ressourcen lädt der OMS-ReportWriter aus Formular- oder Design-Files. Schriftarten, im Folgenden Fonts genannt, allerdings nicht. Das liegt vor allem daran, dass Fonts eher allgemeiner Natur sind und nicht Formular-spezifisch. Es kann aber auch daran liegen, dass Fonts sehr komplexe Gebilde sind, die nur bedingt in einen Formular-Designer integrierbar sind.

Da im Allgemeinen kein Formular ohne Fonts auskommt, ist es wichtig, die im OMS-ReportWriter implementierte Font-Technologie in ihrem Funktionsumfang, der Art der Installation und in den Möglichkeiten der Konfiguration zu erläutern.

Funktionsumfang

Da der OMS-ReportWriter seine Ausgabe als PDF generiert, ist die Font-Technologie an der des Adobe PDFs ausgerichtet. Das betrifft vor allem die unterstützten Font-Formate. Folgende Font-Formate werden unterstützt:

PostScript-Fonts

Beschreibung	*Datei-Endungen*
Adobe Font Metrics (Metric-Datei) Plattform-unabhängig	.afm
Printer Font ASCII (Outline-Datei) Plattform-unabhängig	.pfa
Printer Font Metrics (Metric-Datei) Windows	.pfm
Printer Font Binary (Outline-Datei) Windows	.pfb
OpenType mit PostScript-Zeichenbeschreibungen	.otf

Für die Einbettung eines PostScript-Fonts reicht eine Metric-Datei allein nicht aus. Es wird auch immer eine Outline-Datei benötigt, die auch die Font-Glyphen beinhaltet.

TrueType- und OpenType-Fonts

Beschreibung	*Datei-Endungen*
TrueType-Font	.ttf
OpenType-Font mit TrueType-Zeichenbeschreibung	.ttf .otf
TrueType Collections	.ttc

Unabhängig von dieser Liste ist es im Zusammenhang mit der Ausgabe über OMS-PDFxOut günstig, TrueType oder TrueType-basiertes OpenType als Font-Format zu verwenden. Das darin enthaltene TrueType verstehen viele Drucker direkt, so dass OMS-PDFxOut sich meist die Fontkonvertierung ersparen kann.

Installation

Grundsätzlich müssen die im Formular definierten Fonts erst einmal als Datei auf dem Rechner existieren, auf dem der OMS-ReportWriter läuft. Bitte beachten Sie hierbei die Rechte der Hersteller der Fonts. Unter Umständen ist das Recht zur Verwendung eines Fonts an ein bestimmtes Betriebssystem gebunden. Der OMS-Spooler legt bei seiner Installation ein Unterverzeichnis Fonts an, in dem Font-Dateien abgelegt werden können.

Existiert der Font, so muss der OMS-ReportWriter diesen auch noch finden. Dafür bietet der OMS-ReportWriter zwei Mechanismen. Auf Microsoft Windows Systemen reicht es aus, wenn der Font dem Betriebssystem bekannt gemacht wurde. Der OMS-ReportWriter findet die entsprechenden Einträge in der Registry. Auf Nicht-Windows-Systemen leistet das Programm OMS-Fontreg die Arbeit der Fontregistrierung. OMS-Fontreg arbeitet aber auch unter Microsoft Windows Systemen. Das Grundprinzip, nach dem OMS-Fontreg arbeitet, ist dasselbe wie bei Microsoft Windows. Erst muss ein Font registriert bzw. bekannt gemacht werden, dann findet der OMS-ReportWriter den Font automatisch.

Zur Registrierung eines Fonts wird der Font in das Fonts-Verzeichnis des OMS-Spoolers kopiert und dann das Programm OMS-Fontreg aufgerufen. Der Pfad des Fonts-Verzeichnisses wird als Kommandozeilen-Parameter mitgegeben. Die entstehende fonts.ini wird ins etc-Verzeichnis des OMS-Spoolers kopiert.

Konfiguration

Nach der erfolgreichen Installation sollte der OMS-ReportWriter alle Fonts automatisch finden. In einigen Fällen soll aber nicht der Original-Font angezogen werden, sondern ein anderer Font, der dem Original-Font ähnlich sieht. Wir sprechen hier von einem Alias-Namen, der einem wirklich existierenden Font zugewiesen wird. Die Eintragung von Alias-Namen geschieht in der reportw.ini unter PDF/Fonts/Replacements.

Unicode

Der interne Zeichensatz des OMS-ReportWriters ist UCS-2, eine Unicode-Darstellung. Damit sind rund 65000 Zeichen adressierbar. Um ein Zeichen darstellen zu können, wird immer ein Font benötigt, in dem die darzustellenden Zeichen vorhanden sind. Das klingt einfach, ist in der Praxis aber recht schwierig. Einige Font-Formate unterstützen kein Unicode. Andere Font-Formate unterstützen zwar Unicode, aber die meisten Fonts sind nicht voll besetzt, so dass ganze Zeichenbereiche fehlen. Vollbesetzte Unicode-Fonts sind noch die Ausnahme. Achten Sie darauf, dass Sie Fonts verwenden, die in den von Ihnen verwendeten Zeichenbereichen vollständig sind.

Haben Sie Probleme mit dem Finden des richtigen Fonts, dann wenden Sie sich bitte an uns. Neben der Möglichkeit der Font-Manipulation und -Konvertierung bieten wir auch vollbesetzte Global-Font zum Kauf an.

POSTALISCHE AUFARBEITUNG

Die postalische Aufarbeitung von Sendungen ist ein Schwerpunkt dieses Programms. Dabei geht es darum, eine Sendung einem Postdienstleister und dessen Produkten zuzuordnen und Druckstapel zu bilden, die den Einliefer- und Produktionsanforderungen des jeweiligen Postdienstleisters entsprechen. Es werden dabei zwei Module der postalischen Aufarbeitung unterschieden:

1. frei konfigurierbares Postdienstleister-Modul
2. DV-Freimachung für die DPAG (Deutsche Post AG)

Das frei konfigurierbare Postdienstleister-Modul gestattet es Ihnen in einer einfachen Definition Postdienstleister, deren Produkte und deren Flächendeckung zu konfigurieren. Damit ist es geeignet die Hauptprodukte eines nationalen und international tätigen Postdienstleisters oder eines lokalen Postdienstleisters abzudecken. Spezielle Produktions- und IT-Verfahren eines Postdienstleisters lassen sich allerdings nicht abbilden.

Das Modul DV-Freimachung für die DPAG ist dagegen ein speziell auf die Bedürfnisse der Deutschen Post AG zugeschnittenes Modul, das neben der Produktzuordnung auch die Portooptimierung, und viele spezielle Produktions- und IT-Verfahren unterstützt. Dieses Modul ist von der DPAG zertifiziert

Beide Module lassen sich auch miteinander verbinden, in dem eine Sendung erst einem bevorzugten frei konfigurierten Postdienstleister zugeordnet wird. Wird bei diesem Druckdienstleister kein Produkt oder keine Flächendeckung für die Sendung gefunden, so kann die DPAG alternativ als Postdienstleister angegeben werden.

Einstellungen in der TCI für die Registrierung der Sendung

Um die postalische Aufarbeitung zu aktivieren, muss in der TCI im Basis-Objekt CommonSettings der Schalter PostalService auf 1 gesetzt werden.

```
Beispiel:

CommonSettings {
  ...
  PostalService 1
  ...
}
```

Um eine Sendung für die postalische Aufarbeitung zu konfigurieren, sind im führenden WorkItem des 1. Kanals im Objekt EnvelopeSortSystem folgende Angaben zu machen:

PostalServiceProvider	Name des Postdienstleisters (DPAG oder anderer)
Mandant	Name des Mandanten in der dpdv.ini
Class	gewünschte Sendungsart
Action	Versandaktion für Sendungsart InfoBrief und InfoPost
ZIPCode	Postleitzahl des Sendungsempfängers
CountryCode	Länderkennzeichen des Sendungsempfängers
ITProcessingDate	DD.MM.YYYY (optional)
PostingDate	DD.MM.YYYY (optional)
Dimension	Dimension (optional - Pflicht bei InfoPost mit Behälterfertigung)
WindowDesign	Positions, SubForm Design-Objekt mit den Elementen des Fenstertextes

Alle Schlüsselwörter sind als FieldOrValue ausgelegt.

```
Beispiel:

EnvelopeSortSystem {
   ...
   PostalServiceProvider      DPAG
   Mandant                    @Mandant
   Class                      Brief
   ZIPCode                    @PLZ
   CountryCode                @LKZ
   Dimension                  DL
   WindowDesign               FENSTERPOS,DPDVFenster.xdp|DMGROSS
   ...
}
```

Freie Postdienstleister Konfiguration

Das Modul zur postalischen Aufarbeitung mit freier Konfiguration der Postdienstleister ist ein allgemeines Modul, das auf keinen speziellen Druckdienstleister zugeschnitten ist und eine einfache Konfiguration von Postdienstleistern, deren Produkten und deren Flächendeckung realisiert.

Vor der Nutzung des Moduls müssen in der Konfigurationsdatei psp.ini die Postdienstleister und deren Produkte eingetragen werden. Produkte lassen sich auf PLZ, Postleitzahlenbereiche und Länder beschränken, so dass eine Flächenverteilung für den Postdienstleister definierbar ist.

Es können mehrer Postdienstleister angelegt werden, die in einer Abfolge hintereinander geschaltet werden. Der erste Postdienstleister, der ein passendes Produkt hat, bekommt die Sendung zugeteilt. Wird kein passendes Produkt gefunden, so wird der alternative Postdienstleister gesucht und beauftragt, ein passendes Produkt für die Sendung zu finden. Pro Postdienstleister und Produkt wird ein Druckstapel angelegt, der alle Sendungen enthält, die diesem Produkt zugeteilt wurden. Optional kann der Druckstapel nach PLZ sortiert werden.

Der Postdienstleister „DPAG“ ist fest vorgegeben und kann in der psp.ini nicht konfiguriert werden. DPAG steht für Deutsch Post AG und ruft immer die DPDV-Freimachung, die ein anderes Modul darstellt und die in der dpdv.ini konfiguriert wird.

Angaben in der psp.ini

Alle Zeilen und Objekte werden über den LineReader als Preprozessor gelesen und unterstützen Includes und das Lesen aus Sections.

PostalServiceProvider {

PostalServiceProvider ist ein Basis-Objekt und definiert den Postdienstleister mit seinen Produkten:

PostalServiceProvider Name {
  Products {{
    ...
    Name {
      ...
    }
    ...
  }
  AlternativePSP Value
}

AlternativePSP

AlternativePSP ist der Names eines weiteren Postdienstleisters, der für eine Sendung ein Produkt suchen soll, wenn der aktuelle Postdienstleister kein passendes Produkt anbieteten kann.

Ein Produkt definiert sowohl die Eigenschaften für die Erkennung eines Produktes aus den Sendungsdaten, als auch die Art der Verarbeitung. Produkte haben folgenden syntaktischen Aufbau:

Name {
  Class Value
  Weight Value
  National Value
  DoSortZIPCodes Value
  AllowedZIPCodes Value
  AllowedCountrys Value
  Postage Value
}

Class

Sendungsart des Produktes. Typische Sendungsarten sind BRIEF oder INFOBRIEF. Erfolgt die Produktsuche über mehrere Postdienstleister hinweg, so muss die Sendungsart bei allen Postdienstleistern gleich sein. Diese Angabe ist Pflicht.

Weight

Maximalgewicht der Sendung in Gramm. Diese Angabe ist Pflicht.

National

National ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so handelt es sich um ein Inlandprodukt.

DoSortZIPCodes

DoSortZIPCodes ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so werden die Sendungen der PLZ nach aufsteigend sortiert.

AllowedZIPCodes

AllowedZIPCodes definiert, für welche Postleitzahlen ein Produkt Gültigkeit hat.
AllowedZIPCodes ist eine kommaseparierte Liste von Postleitzahlen oder Fragmenten von Postleitzahlen. Bei Fragmenten werden die Postleitzahlen von links mit den Fragmenten verglichen.

AllowedCountrys

AllowedCountrys definiert, für welche Länder ein Produkt Gültigkeit hat.
AllowedCountrys ist eine kommaseparierte Liste mit Länderbezeichnungen. Die Länderkennzeichen müssen denen entsprechen, die im EnvelpoeSortSystem der Sendung angegeben wurden.

Postage

Postage ist das Porto oder das Entgelt, welches für die Sendung zu entrichten ist. Die Angabe erfolgt ohne Währungskennzeichen im Format zzz9.zz.

Die Reihenfolge der Auflistung der Produkte spielt keine Rolle. Sie werden automatisch nach dem Porto aufsteigend sortiert.

```
Beispiel:

PostalServiceProvider ZeitungsvertriebVonHier {
 Products {
  Standard {
   Class Brief
   Weight 20
   National 1
   DoSortZIPCodes 1
   AllowedZIPCodes 603,604,605
   Postage 0,36
  }
  Kompakt {
   Class Brief
   Weight 50
   National 1
   DoSortZIPCodes 1
   AllowedZIPCodes 603,604,605
   Postage 0,67
  }
 }
 AlternativePSP DPAG
}
```

Reporting {

Reporting ist ein Basis-Objekt und definiert das Fehlerverhalten und das Verhalten für das Reporting:

```
 Reporting Name {
   CancelOnError       Value
 }
```

CancelOnError

CancelOnError ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so erfolgt ein Abbruch des Programmlaufes, wenn ein Fehler bei der Bestimmung des Produktes auftrat oder kein Produkt gefunden werden konnte.

DV-Freimachung Deutsche Post

Dieses Modul ist im Handbuch DVFreimachung ausführlich beschrieben.

ARBEITEN MIT TABELLEN

Im OMS-ReportWriter wurden einige grundsätzliche Änderungen bei der Generierung von SubForms eingeführt. So wurde die SubForm-Generierung durch ein Tabellenkonzept erweitert.

Das Tabellenkonzept geht davon aus, dass die generierten SubForms nicht einfach nur in einer Liste gespeichert und dann ausgegeben werden, sondern die Liste ist jetzt ein Baum von ineinander geschachtelten Tabellen. Warum Tabellen? Es gibt mindestens zwei Gründe:

1. Mit Tabellen kann ein Protect-Konzept realisiert werden, das Tabellen vor dem Umbruch von einer Seite auf die andere schützt.

2. Mit Tabellen können automatisch Header- und Trailer-SubForms generiert werden. Auch die Seitenumbruch-SubForms lassen sich mit Tabellen einfacher und flexibler lösen.

Warum ein Baum von Tabellen?

Erst durch ineinander geschachtelte Tabellen bekommt das Tabellenkonzept die Flexibilität, die in realen Projekten benötigt wird.

An welcher Stelle entsteht dieser Tabellenbaum?

Die SubForm-Tabellen werden durch die SubForms selbst generiert. Die SubForm-Tabellen haben von Ihrem Aufbau nichts mit der Hierarchie der Positionen, dem Positioning oder Grouping zu tun. Sie werden erzeugt und gesteuert durch tatsächlich auf der Seite ausgegebene SubForms. Benötigen Sie zusätzlich SubForms, die nur zur Steuerung verwendet werden, dann setzen Sie den Schalter TableControl dieses SubForms auf 1. SubForms bei TableControl werden zwar für den Druck erzeugt, nach dem Aufbau des Tabellenbaumes aber wieder gelöscht. TableControl ist ein Schalter mit den Werten 0 und 1. 0 ist der Standardwert. TableControl ist FieldOrValue.

Beispiel:

```
SubForm {
  ...
  TableControl 1
  ...
}
```

In direkten Zusammenhang mit den TableControl SubForms steht der Schalter RemoveEmptyTables des Basisobjektes CommonSettings. Da nach dem Entfernen von SubForms, die nur zum Aufbau des Tabellenbaumes dienten, leere Tabellen übrig bleiben können, ist die Entscheidung zu treffen, wie mit diesen leeren Tabellen umzugehen ist. Die Verfahrensweise wird durch den Schalter RemoveEmptyTables mit den Werten 0 und 1 geregelt. Standardwert ist 1.

Beispiel:

```
CommonSettings {
  ...
  RemoveEmptyTables 0
  ...
}
```

Erstellen des Tabellenbaumes

WrapTables

WrapTables sind Tabellen, die entstehen, wenn ein SubForm angewiesen wurde, sich selbst und alle Unter-SubForms in eine Tabelle zu schreiben.

Beispiel:

```
  SubForm {
    FIELDS Test:80
    Name TestPos
    WrapTable PosWrap
  }

Das SubForm TestPos erzeugt ein oder mehrere SubFormulare, je
nachdem, wie groß der Text im Feld Test ist. Alle diese SubForms
werden in eine Tabelle PosWrap geschrieben, die an
anderen Stellen der TCI noch ausführlich definiert werden kann
(aber nicht muss).
```

TableCommands

TableCommands sind Befehle in der SubForms-Definition, die ein SubForm anweisen, sich in eine bestimmte bestehende Tabelle mit einzuordnen, eine neue Tabelle zu eröffnen usw. Hier gibt es eine ganze Anzahl von Befehlen und Logiken, die weiter unten beschrieben werden.

Beispiel:

```
  SubForm {
    Name Start
    Table ProtectPos:ON
  }
  SubForm {
    Name Inhalt
  }
  SubForm {
    Name Stop
    Table ProtectPos:OFF
  }
```

```
Das SubForm Start öffnet die Tabelle ProtectPos und Stop beendet
die Tabelle ProtectPos. Das SubForm Inhalt steht zwischen beiden
und wird auch mit in die Tabelle ProtectPos übernommen.
```

Zur Realisierung des Tabellenkonzeptes wurde am Objekt SubForm ein neues Schlüsselwort Table eingeführt. Table ist als FieldOrValue Wert ansteuerbar und enthält einen Tabellennamen und ein Kommando, die durch einen Doppelpunkt voneinander separiert sind:

Syntax

Table TABLENAME:COMMAND

Der Tabellenname kann frei definiert werden. Optimal ist es, wenn es zum Tabellennamen noch ein Basisobjekt Table mit demselben Namen gibt, dann werden die Eigenschaften der Tabelle durch dieses Basisobjekt gesteuert. Existiert das Basisobjekt Table nicht, so werden die Eigenschaften der Tabelle durch Standardwerte gesteuert.

Folgende Kommandos(Command): können verwendet werden:

ON:

Start einer Tabelle mit dem angegebenen Tabellennamen. Sich wiederholende ON Kommandos mit denselben Tabellennamen werden ignoriert. Kommt ein ON Kommando innerhalb einer offenen Tabelle, so wird im Baum ein neuer Tabellenzweig eröffnet, sprich die Verschachtelungstiefe erhöht sich um eine Ebene. Sie bewegen sich dann in einer Untertabelle der Tabelle.

OFF:

Ende der angegeben Tabelle nach diesem SubForm. Das nachfolgende SubForm ist nicht mehr Bestandteil der Tabelle. Ist kein Tabellenname angegeben, so wird die aktuelle Tabelle geschlossen.

OFFBEFORE:

Ende der angegeben Tabelle vor diesem SubForm. Dieses SubForm ist nicht mehr Bestandteil der Tabelle. Ist kein Tabellenname angegeben, so wird die aktuelle Tabelle geschlossen.

ONOFF:

Dieses SubForm legt eine neue Tabelle an, in die nur dieses SubForm geschrieben wird. Das aktuelle SubForm öffnet und schließt die Tabelle. Dieses Kommando wird vor allem dann benutzt, wenn die Einstellung dieser Tabelle aus dem Basisobjekt Table definiert wird.

RESTART:

Ende der aktuellen Tabelle vor diesem SubForm und Start einer neuen Tabelle mit dem angegebenen Namen. Da RESTART die aktuelle Tabelle abschließt, kann dieses Kommando keine Untertabellen erzeugen.

RESTARTOFF:

Ende der aktuellen Tabelle vor diesem SubForm, Start einer neuen Tabelle mit dem angegebenen Namen und Stop dieser Tabelle nach dem aktuellen SubForm. Da RESTARTOFF die aktuelle Tabelle abschließt, kann dieses Kommando keine Untertabellen erzeugen.

RESTARTAFTERCHANGE:

Ende der aktuellen Tabelle vor diesem SubForm und Start einer neuen Tabelle mit dem angegebenen Namen. Allerdings wird der Restart nur ausgeführt, wenn zwischen dem letzten Tabelleneröffnungs-Kommando und dem RESTARTAFTERCHANGE Kommando mindestens ein SubForm mit einem anderen Tabellennamen vorkommt. Da RESTARTAFTERCHANGE die aktuelle Tabelle abschließt, kann dieses Kommando keine Untertabellen erzeugen.

RESET:

Ende der aktuellen Tabelle nach dieser Position und Rückkehr aus allen Verschachtelungsebenen. Die nächste Position beginnt wieder auf dem obersten Level. Die Angabe eines Tabellennamens ist nicht notwendig und wird ignoriert.

RESETBEFORE:

Ende der aktuellen Tabelle vor diesem SubForm und Rückkehr aus allen Verschachtelungsebenen. Die aktuelle SubForm beginnt wieder auf dem obersten Level. Die Angabe eines Tabellennamens ist nicht notwendig und wird ignoriert.

Definition des Tabellenverhaltens

Um das Verhalten von Tabellen zu steuern, wurde das Basisobjekt Table entwickelt.

Syntax

```
Table TableName {
  Protect Value
  HeaderSubForm {
    ...
  }
  TrailerSubForm {
    ...
  }
  PageBottomSubForm {
    ...
  }
  PageTopSubForm {
    ...
  }
  PageBottomJoinedPosition Value
  PageTopJoinedPosition Value
}
```

Protect Value

Protect ist ein Schalter mit den Werten 0 und 1. Standardwert wird über den Schalter AutoProtectTables unter CommonSettings eingestellt. Der Standardwert von AutoProtectTables ist 0. Ist Protect eingeschaltet, so werden alle in der Tabelle enthaltenen Positionen und Subtabellen zusammengehalten und so vor einem Seitenumbruch innerhalb der Tabelle geschützt. Ist die Tabelle zu groß, um auf einer Seite des Formulars gedruckt zu werden, so wird die Eigenschaft Protect automatisch zurückgesetzt.

PageBottomJoinedPosition Value

PageBottomJoinedPosition ist eine Zahlenangabe von 0 bis n. Definiert die Anzahl von SubForms und Subtabellen, die bei einem Seitenumbruch innerhalb einer Tabelle wenigstens am unteren Ende der Seite

stehen müssen. Der Standardwert ist 0. HeaderSubForm, TrailerSubForm, PageBottomSubForm spielen bei der Berechnung keine Rolle. Diese Funktion ist auch als „Schusterjungenregel“ bekannt.

PageTopJoinedPosition Value

PageTopJoinedPosition ist eine Zahlenangabe von 0 bis n. Definiert die Anzahl von SubForms und Subtabellen, die bei einem Seitenumbruch innerhalb einer Tabelle wenigstens am oberen Anfang der Seite stehen müssen. Der Standardwert ist 0. HeaderSubForm, TrailerSubForm, PageTopSubForm spielen bei der Berechnung keine Rolle. Diese Funktion ist auch als „Schusterjungenregel“ bekannt.

HeaderSubForm , **TrailerSubForm**, **PageBottomSubForm**, **PageTopSubForm**

HeaderSubForm, TrailerSubForm, PageBottomSubForm, PageTopSubForm:

Alle SubForm-Befehle sind normale SubForms, wie diese unter dem Basisobjekt Position als SubForm beschrieben sind.

HeaderSubForm: Tabellenkopf, der vor den SubForms der Tabelle gedruckt wird.
TrailerSubForm: Tabellenfuß, der nach den SubForms der Tabelle gedruckt wird.
PageBottomSubForm: Tabellenfuß beim Umbruch innerhalb der Tabelle.
PageTopSubForm: Tabellenkopf beim Umbruch innerhalb der Tabelle.

Definition des Tabellenverhaltens aus Standardwerten

Nur die Eigenschaft Protect von Tabellen lässt sich als Standardwert definieren. So existiert im Basisobjekt CommonSettings das Schlüsselwort AutoProtectTables. AutoProtectTables ist ein Schalter mit den Werten 0 und 1. Der Standardwert ist 0. AutoProtectTables definiert den Standardwert für das Schlüsselwort Protect des Objektes SubForm.

LESEN VON ADOBE XDP-DATEIEN

Ein wesentlicher Punkt beim Design einer TCI ist es, Doppeldefinitionen zu vermeiden. Dies gelingt dadurch, dass der OMS-ReportWriter Adobe XDP-Dateien lesen und so auf unsinnige Doppeldefinitionen verzichten kann.

XDP-Dateien sind XML-basierte Designbeschreibungen, die aus dem Adobe Designer oder anderen kompatiblen Designern stammen. Das Verzeichnis für die XDP-Dateien kann als Übergabeparameter -afp mitgegeben werden

Der OMS-ReportWriter liest aus den Designfiles alle Höhen- und Breiten-Angaben, die Feldbezeichnungen und Teile der Business-Logik heraus. Einzig die Höhenangabe von SubForms kann im OMS-ReportWriter überschrieben werden.

Zur Vereinfachung des Positionings dient der Schalter AutoPositioning im Basis-Objekt DocDef. AutoPositioning ist ein FieldOrValue Schalter mit den Werten 0 und 1, wobei 0 der Standard ist.

Wenn AutoPositioning eingeschaltet ist, dann ist jede andere Positioning-Angabe ohne Wirkung. Ist AutoPositioning eingeschaltet, so erfolgt die Zuordnung der Felder der globalen Tabelle zu Positionen über die aus der XDP-Datei gelesenen Informationen. AutoPositioning arbeitet nur, wenn auf DocDef-Ebenen ein FormFileName angegeben wurde.

Aus der angegebenen XDP-Datei werden alle SubForms gelesen und die Felder der globalen Tabelle mit den Feldern der SubForms verglichen. Passt ein Feld zu einem SubForm, so wird eine neue Position im OMS-ReportWriter eröffnet und das Feld dort hinein bewegt. Sind die folgenden Felder auch im SubForm enthalten, so werden diese ebenfalls dorthin bewegt. Feldwiederholungen führen auch zur Wiederholung der Position. Passt ein Feld nicht in das zuletzt gefundene SubForm, so beginnt die Suche nach einem passenden SubForm erneut.

LADEN VON TCI-FILES

Vor der Verarbeitung der Daten wird der Basis-TCI-File vom OMS-ReportWriter geladen. Alle Einstellungen globaler Natur müssen in diesem TCI-File vorhanden sein. Zur besseren Modularisierung ist es in einigen Objekten der TCI allerdings erlaubt, nachzuladende TCI-Files anzugeben. Diese werden nach Bedarf aufgerufen und dürfen aber nicht mehr alle Basisobjekte der TCI verwenden. Wenn die nachzuladenden Formular-TCIs ohne Pfad angegeben werden, dann versucht der OMS-ReportWriter diese im gleichen Verzeichnis zu finden wie die Basis-TCI.

Um ein Projekt mit verschiedenen Formularen übersichtlich zu halten, kann ein Gesamtprojekt in eine Basis-TCI und mehrere Formular-TCIs aufgeteilt werden. Die Basis-TCI wird beim Starten des OMS-ReportWriters geladen.

Die Formular-TCIs werden vom OMS-ReportWriter nachgeladen, wenn ein DocDef oder ein WorkItem erkannt wird, das auf eine zusätzliche Formular-TCI verweist. Die Formular-TCIs werden in denselben Namensraum geladen, so als wenn alle Formular-TCIs und Basis-TCIs in einer TCI zusammengefasst wären. Der Verweis auf eine Formular-TCI kann im DocRec- und im WorkListVariant-Objekt angegeben werden. Eine nachgeladene Formular-TCI darf nicht mehr alle Basis-Objekte definieren. Folgende Objekt-Typen werden beachtet:

- DocDef
- Position
- Table
- WorkListVariant
- WorkItem
- Substitute

Alle anderen Objekt-Typen werden in den nachgeladenen Formular-TCIs ignoriert und müssen demzufolge in der Basis-TCI angegeben werden, die als erstes geladen wird.

Beispiel:

```
DOCREC {
  COVER_NACHSCHUB DOCDEF=HEADER (COVER_NACHSCHUB.TCI)
  COVER           DOCDEF=HEADER (COVER.TCI)
}

Je nach Bedarf werden die COVER_NACHSCHUB.TCI, die COVER.TCI oder
beide nachgeladen.
```

Da die TCI über das Modul LineReader gelesen wird, gibt es neben dem Nachladen von TCIs auch die Möglichkeit, einzelne Abschnitte einer TCI aus einem anderen File zu lesen. Mehr darüber finden Sie im Abschnitt LineReader.

INLINE-SEQUENZEN

Der OMS-ReportWriter hat die Aufgabe, unformatierte Nettodaten mit Formularen zu mischen und formatiert auszugeben. In einigen Fällen ist es aber sinnvoll, wenn die Applikation, die die Daten erstellt, dem OMS-ReportWriter Formatierungsanweisungen mitgeben kann. Diese Formatierungsanweisungen werden in Form von Inline-Sequenzen definiert und sind in ihrem Stil an die Inline-Sequenzen der Firma Jetform angelehnt. Inline-Sequenzen befinden sich innerhalb normaler Textfelder und werden bei ihrer Ausführung interpretiert. Über Inline-Sequenzen lassen sich die wesentlichen Textformatierungseigenschaften steuern. Inline-Sequenzen beginnen mit einem \ und enden entweder mit einem . oder mit einem Leerzeichen. Folgende Inline-Sequenzen werden durch den OMS-ReportWriter unterstützt:

n	NewLine-Zeichen
b	Bold-Steuerung (Ein- oder Ausschalten von Fett)
i	Italic-Steuerung (Ein- oder Ausschalten von Kursiv)
ul	UnderLine-Steuerung (Ein- oder Ausschalten von Unterstreichung)
normal	Schaltet alle Steuerungen wie Bold und Italic auf ausgeschaltet
plain	Zurücksetzen auf die im Design festgesetzten Font-Eigenschaften
dn	BaseLine-Shift versetzt die Zeichenlinie nach unten
up	BaseLine-Shift versetzt die Zeichenlinie nach oben
fs	FontSize verändert die Schriftgröße
fn	FontName: Wechseln des Fonts über seinen Namen
f	FontNumber: Wechseln des Fonts über einen Alias-Namen
ol	OverLine-Steuerung (Ein- oder Ausschalten von Überstrich)
sl	StrikeOutLine–Steuerung (Ein- oder Ausschalten der Durchstreichung)
cn	Color: Setzen der Farbe
u+	Unicode Codepoint wird eingefügt zur Unterstützung von Unicode-Zeichen
t	Unterstützt Tabulatoren im Text
link	Link auf eine URL

n NewLine-Zeichen

Einfügen eines Zeilenumbruchs

b Bold-Steuerung (Ein- oder Ausschalten von Fett)

Bold ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standard ist. Mit diesem Schalter kann das Textattribut Fett ein- oder ausgeschaltet werden.

 Syntax

b[0|1]

i Italic-Steuerung (Ein- oder Ausschalten von Kursiv)

Italic ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standard ist. Mit diesem Schalter kann das Textattribut Kursiv ein- oder ausgeschaltet werden.

 Syntax

i[0|1]

ul UnderLine-Steuerung (Ein- oder Ausschalten von Unterstreichung)

UnderLine ist ein Schalter mit den Werten w, db, 0 oder 1, wobei 1 der Standard ist. Hier können die Eigenschaften für die Unterstreichung gesteuert werden.

 Syntax

ul[w|db|0|1]

w	Wortunterstreichung , die Zwischenräume zwischen den Worten werden nicht unterstrichen
db	DoubleLine: es werden zwei Unterstreichungslinien gezeichnet
0	Ausschalten der Unterstreichungen
1	Einfache Unterstreichung

ol OverLine-Steuerung (Ein- oder Ausschalten von Überstrich)

OverLine ist ein Schalter mit den Werten w, db, 0 oder 1, wobei 1 der Standard ist. Hier können die Eigenschaften für die Überstreichung gesteuert werden.

Syntax

ol[w|db|0|1]

w	Wortüberstreichung , die Zwischenräume zwischen den Worten werden nicht überstrichen
db	DoubleLine: es werden zwei Überstreichungslinien gezeichnet
0	Ausschalten der Überstreichungen
1	Einfache Überstreichung

sl StrikeOutLine–Steuerung (Ein- oder Ausschalten der Durchstreichung)

StrikeOutLine ist ein Schalter mit den Werten w, db, 0 oder 1, wobei 1 der Standard ist. Hier können die Eigenschaften für die Durchstreichung gesteuert werden.

Syntax

sl[w|db|0|1]

w	Wortdurchstreichung , die Zwischenräume zwischen den Worten werden nicht durchgestrichen
db	DoubleLine: es werden zwei Durchstreichungslinien gezeichnet
0	Ausschalten der Durchstreichungen
1	Einfache Durchstreichung

normal Schaltet alle Steuerungen wie Bold und Italic auf ausgeschaltet

Normal schaltet sämtliche Textattribute eines Fonts aus. Der danach folgende Text erfolgt als regulärer Text ohne Verschiebung der BaseLine.

plain Zurücksetzen auf die im Design festgesetzten Font-Eigenschaften

Plain ist ein Schalter, der alle gewählten Textattribute zurücksetzt auf den Font mit all seinen Eigenschaften, der im Design für dieses Objekt vorgegeben war.

dn BaseLine-Shift versetzt die Zeichenlinie nach unten
up BaseLine-Shift versetzt die Zeichenlinie nach oben

BaseLine-Shift down or up verschiebt die BaseLine, auf der die Buchstaben stehen, nach oben oder nach unten. Die BaseLine beschreibt den Fußpunkt der großgeschriebenen Buchstaben.

Syntax

dn[points]
up[points]

Points ist die Angabe der Verschiebung in ganzen Punkten, Kommawerte sind nicht zulässig.

fs FontSize verändert die Schriftgröße

FontSize setzt die Größe des gewählten Fonts.

Syntax

fs[decipoints]

Decipoints ist die Angabe der Schriftgröße in Decipoints das entspricht einem Wert von 1/720 Zoll. Kommawerte sind nicht zulässig. Die Fontgröße von 10.5 pt entspricht 105 decipoint und damit dem Kommando „\fs105.“.

fn FontName: Wechseln des Fonts über seinen Namen
f FontNumber: Wechseln des Fonts über einen Alias-Namen

FontName und FontNumber setzen gleichermaßen den Namen eines Fonts, wobei FontName eventuell vorkommende Hochkommas eliminiert. Soll der Font über einen Alias-Namen angesprochen werden, so ist dieser zuvor in der reportw.ini unter Fonts/Replacements zu definieren.


Syntax

```
fn“FontName“
fFontNumber
```

cn Color: Setzen der Farbe

Color setzt die Farbe des Textes. Dabei können 2 unterschiedliche Syntax-Varianten zum Einsatz kommen. In der Variante 1 wird die Farbe über einen RGB-Wert mit dem Wertevorrat von 0 bis 255 gesetzt. In der zweiten Syntax-Variante kann eine Farbe über einen vordefinierten Namen gewählt werden. Die Farbnamen sind aus dem Standard HTML 4.0 abgeleitet.


Syntax

```
cnrgb(rot,grün,blau)
cnName
```

Folgende Farbnamen sind bekannt: aqua, navy, black, olive, blue, purple, fuchsia, red, gray, silver, green, teal, lime, yellow, maroon, white

u+ Insert Unicode Code Point

Mit \u+ können Unicode-Zeichen in den Text eingefügt werden, die nicht auf der Tastatur vorhanden sind.

Syntax

u+[codepoint]

Codepoint steht als hexadezimale Zahl für ein Unicode-Zeichen.

```
Beispiel

Das Eurozeichen, \u+20AC, ist das Währungszeichen für die
europäische Währungsunion.

Das Ergebnis ist:
Das Eurozeichen, €, ist das Währungszeichen für die europäische
Währungsunion.
```

t Unterstützt Tabulatoren im Text

Anhand von \t gelangt man zum nächsten Tabulator. Durch Wiederholen des Befehls kommt man zu weiteren Tab-Stopps. Dieser Schalter dient als Alternative zum Setzen eines ASCII Dezimal 9-Tabulators in den Datenfluss, um zum nächsten Tab-Stopp zu gelangen.

Syntax

t

```
Beispiel:`

Produkt\t.Menge\t\t.Einheiten1001\t. 12\t\t.JE

Das Ergebnis ist:
Produkt     Menge       Einheiten
1001        12          JE
```

link Link auf eine URL

PDF-Texte können genauso wie HTML-Texte auch Links auf Webseiten beinhalten. Da PDF-Links nicht in allen PDF-Versionen unterstützt werden, werden Links nur in PDF aufgemacht, die nicht als PDF/A oder PDF/X eröffnet wurden. Das Inline-Kommando Link besitz einen optionalen Parameter, den URL-Link. Wenn ein URL-Link gesetzt wurde, dann gilt dieser für alle nachfolgenden Texte. Soll der aktuelle Link gelöscht werden, so wird der Befehl ohne Parameter angegeben.

Syntax

link["URL"]

Beispiel

```
Das ist ein
\link"http://www.docxworld.de/produkte/produktuebersicht/docxworld
daily/".Link\link. auf unsere daily-Produkte
```

LINEREADER

Der LineReader ist ein Preprozessor zum Lesen von Konfigurationsdateien. Alle über den LineReader gelesenen Dateien lesen die Dateien nicht direkt, sondern durchlaufen zeilenweise den Preprozessor. Dieses Verfahren hat den Vorteil, dass spezielle Befehle vom Preprozessor abgefangen und von diesem ausgewertet werden können, ohne dass die Module zur Auswertung der Konfigurationsdatei davon Kenntnis erlangen. Zeilen, die an der ersten Stelle mit dem Zeichen * beginnen, werden vom LineReader ausgewertet. Alle andere Zeilen werden direkt durchgereicht.

Wichtigste Aufgabe des LineReaders ist das Verzweigen in andere Dateien in Form von Includes.
Dies geschieht über den Include-Befehl. Der LineReader verzweigt dabei in eine andere Datei und kehrt nach dem Lesen dieser Datei wieder in die ursprüngliche Datei zurück, um dort die nächste Zeile zu lesen. Weiterhin ist es möglich, in der einzufügenden Datei nur einen Abschnitt, eine Section, zu lesen. Der Beginn und das Ende einer Section müssen wiederum über LineReader-Befehle kenntlich gemacht werden. In einer einzufügenden Datei können natürlich wieder Include-Befehle stehen. Die Verschachtelungstiefe ist dabei nicht begrenzt, wobei auf zyklische Endlosschleifen geprüft und im Fehlerfall abgebrochen wird.

Beim Aufruf des Include-Befehls ist es erlaubt, Variablen anzugeben, die LineReader-Variablen. LineReader-Variablen sind nicht deckungsgleich mit Variablen aus Dokumenten, sondern werden einzig über den -adv Parameter beim Aufruf des Programms gesetzt.

Es werden drei LineReader-Befehle unterstützt: *include, *sectionstart und *sectionstop.
Alle anderen mit * beginnenden Zeilen werden als Kommentare angesehen.


Syntax

***sectionstart** SectionName
***sectionstop** SectionName

Die Befehle *sectionstart und *sectionstop bilden eine Klammer um den zu lesenden Bereich. Wird kein *sectionstop Befehl gefunden, so liest der LineReader bis zum Ende der aktuellen Datei. SectionName ist ein fester Name des Bereichs und kann nicht über Variablen gelesen werden.


Syntax

***include** FileName [SectionName]

Der Befehl *include verzweigt in die angegeben Datei. Ist kein SectionName angegeben, so wird die gesamte Datei gelesen. Ist ein SectionName angegeben, so werden alle Zeilen vor dem Beginn des Abschnitts und nach dem Ende des Abschnittes ignoriert, nur der Abschnitt wird gelesen. Sowohl der FileName als auch der optionale SectionName können über eine LineReader-Variable gelesen werden. Der LineReader erkennt den Aufruf einer Variablen am Startzeichen @. Der Variablenname endet mit einem Leerzeichen.

Syntax

@VarName

Erfolgt die Angabe der zu lesenden Datei ohne Pfadangabe, so wird die Datei in den Pfaden gesucht, in denen vorherige ini-Dateien vom LineReader gefunden wurden. Dabei wird der zuletzt geöffnete Pfad zuerst untersucht.

Beispiel:

```
*include client.ini @client_Name
```

Beispiel 1:

1.	Datei c:\OMS\etc\reportw.ini	gesucht wird c:\OMS\etc\reportw.ini
2.	Datei Include application.ini in c:\OMS\etc\reportw.ini	gesucht wird c:\OMS\etc\application.ini

Beispiel 2:

1.	Datei c:\OMS\etc\reportw.ini	gesucht wird c:\reportw.ini
2.	Datei Include application.ini in c:\OMS\etc\reportw.ini	gesucht wird c:\OMS\etc\application.ini

Beispiel 3:

1.	Datei c:\OMS\etc\reportw.ini	gesucht wird c:\OMS\etc\reportw.ini
2.	Datei Include application.ini in c:\OMS\etc\reportw.ini	gesucht wird c:\application.ini
3.	Datei customer.ini in c:\application.ini	gesucht wird c:\customer.ini dann c:\OMS\etc\customer.ini

Der Zeichensatz mit dem der LineReader Dateien liest, ist die Standard-CodePage des Systems, auf das Programm läuft. Der Zeichensatz kann aber auch mit dem Kommandozeilen-Parameter –aic gesetzt werden. Für Dateien, die mit einer BOM (Byte Order Mark) am Anfang der Datei arbeiten, erkennt der LineReader automatisch das benötigte Encoding. Erkannt werden hier UTF-8, UTF-16 BE und UTF-16 LE.

GRAFIKEN

Image-Objekte und ImageField-Objekte im XFA erlauben die Einbindung von Rastergrafiken. Dabei werden die Grafikformate unterstützt, die sich ins PDF einbetten lassen. Das Konzept wurde hier noch um die Einbettung von PDF-Dateien erweitert. Die eingebetteten Daten werden in Größe, Auflösung und Farbtiefe nicht verändert. Ein "Down Sampling" erfolgt nicht. Es ist aus diesem Grund wichtig, vor dem Einbinden zu prüfen, ob das Bild für die Verarbeitung geeignet ist.

Einige der Grafikformate haben die Möglichkeit, mehr als ein Bild oder mehr als eine Seite pro Datei halten zu können. Hierfür ist es notwendig, nicht nur einen Dateinamen, sondern auch eine Seite definieren zu können. Die Angabe der Seite erfolgt ähnlich wie bei der Angabe des Subforms mit einem "|"-Zeichen nach dem Filenamen.

Es werden folgende Rasterformate unterstützt:

PNG	PNG-Bilder (ISO 15948)
JPEG	JPEG-Bilder (ISO 10918-1)
JPEG2000	JPEG2000-Bilder (ISO 15444-2 ab PDF Version 1.5)
GIF	GIF-Bilder (insbesondere GIF 87a und 89a)
TIFF	TIFF-Bilder
BMP	BMP-Bilder (Version 2 und 3)
CCITT	CCITT-Bilder (Gruppe 3 und Gruppe 4)
PDF	denselben oder einen höheren Standard

PDF/A

Das PDF/A-Format, das im Standard ISO 19005-1 definiert ist, stellt ein Subset von PDF dar.
Ziel ist es, Dokumente unter Berücksichtigung von Anforderungen der Langzeitarchivierung digitaler Dokumente zu erzeugen. PDF/A in der vorliegenden Version 1 basiert technisch auf PDF 1.4 und definiert neben Einschränkungen auch im PDF optionale Elemente, die zu Pflichtbestandteilen werden.

PDF/A Dokumente müssen in sich vollständig sein. Das bedeutet, dass alle im Dokument benutzten Ressourcen im Dokument selbst enthalten sein müssen. Das betrifft vor allem die Schriftarten und im besonderen auch die Standard-Schriftarten. Weiterhin ist die Angabe der Meta-Daten als XMP und genaue Farbdefinitionen Pflicht. Der ISO-Standard 19005-1 definiert zwei Conformance Level: den Level a und den Level b. Während Level b nur die visuelle Reproduzierbarkeit fordert, so erweitert Level a den Level b um Strukturinformationen, die als "Tagged PDF" bezeichnet werden. Aktuell ist nur der Level b umgesetzt, da "Tagged PDF" nicht unterstützt wird.

Die Erzeugung von PDF/A wird im PDF-Profile mit dem Schlüsselwort PDFVersion gesteuert. Anstelle der dort üblichen Version, wie z. B. "1.7", erfolgt hier die Angabe PDFA1B2005. Die strengeren Regulierungen im PDF/A gegenüber normalem PDF führen unter Umständen aber dazu, dass bereits lauffähige Konfigurationen für normales PDF nachgebessert werden müssen, um erneut lauffähig zu sein. Im Folgenden sollen die Restriktionen und Wege, diese zu umgehen, besprochen werden.

Zu den Restriktionen, die automatisch umgesetzt werden, gehört das strikte Verbot von Kennwörtern und von Rechten auf das PDF. Ist die PDFVersion auf PDF/A gesetzt, so werden alle Rechte und Passwörter automatisch zurückgesetzt. Ebenso automatisch wird die Font-Konfiguration, dem Standard PDF/A entsprechend, angepasst und unabhängig tatsächlicher Einstellungen auf "embedding" gesetzt. Das hat zur Folge, dass Schriftarten, die sich nicht einbetten lassen, zum Abbruch der Generierung führen. Dass sich Schriftarten nicht einbetten lassen, kann unterschiedliche Gründe haben. Das kann z. B. geschehen, wenn eine Schriftart das Einbetten nicht erlaubt. Sollte das der Fall sein, so ist beim Hersteller der Schriftart eine Version der Schriftart zu erwerben, die das Einbetten erlaubt oder es ist eine alternative Schriftart einzusetzen. Besonders problematisch sind in diesem Zusammenhang die PDF Standard-Schriften wie Courier und Helvetica. Da diese Schriftarten meist nicht eingebettet werden, sind für normales PDF nur die bekannten Informationen über die Zeichenbreiten von Bedeutung. Angaben über die genaue Form der Buchstaben, die Outlines, sind nicht notwendig.

Um Standard-Schriften mit ins PDF einzubetten, werden entweder lizensierte Adobe-Schriften benötigt oder es erfolgt ein Austausch der Schriftarten gegen auf dem System bereits vorhandene Schriften.
So kann z. B. die Schriftart "Courier" gegen "Courier New" und "Helvetica" gegen "Arial" getauscht werden. Die Definition der Austausch-Schriftarten kann in der reportw.ini/pdf/fonts/replacements geschehen und muss manuell dort eingetragen werden. PDF/A benötigt in jeder Datei zur vollstän-digen

Farbreproduktion einen Standard-Farbraum, den "Output Intent". Standardmäßig wird hier RGB gesetzt. Das bedeutet, dass in PDF/A nur RGB- oder GrayScale-Images eingebettet werden können.

Ebensolche Restriktionen gibt es in Bezug auf eingebettete PDFs. Diese müssen demselben Standard PDF/A-1b:2005 genügen und den Standard-Farbraum RGB gesetzt haben, um eingebettet werden zu können.

PDF/A Dokumente sind im Allgemeinen größer als normale PDF Dokumente. Das liegt zum einen daran, dass alle verwendeten Schriftarten eingebettet werden, zum anderen sind Metadaten im PDF/A Pflicht. Für ein einzelnes Dokument ist das Größenwachstum meist nicht von ausschlaggebender Bedeutung. Bei der massenhaften Erstellung von Archiv-Dokumenten summiert sich der Wachtums-betrag jedoch sehr schnell und führt zu einem nicht zu vernachlässigenden Anwachsen der Daten-menge, die sowohl bei der Netzlast zur Übertragung ins Archivsystem als auch bei der Berechnung der Speicher- und Mediengrößen im Archivsystem Beachtung finden muss.

Neben vielen Einschränkungen bei der Erzeugung von PDF/A und den gewachsenen Datengrößen bringt PDF/A aber auch viele Vorteile mit sich. So ist ein PDF/A-Dokument jederzeit vollständig visuell reproduzierbar und eignet sich damit auch für andere Anwendungen wie die Abgabe als signiertes Online-Dokument oder zur Weitergabe an den Druckdienstleister.

Durch die im PDF/A enthaltenen Metadaten ist es nicht mehr notwendig, zusätzliche beschreibende Informationen neben dem PDF/A Dokument zu erzeugen und zu transportieren. Je nach Verwendungszweck werden in den Metadaten des PDF/A spezifische Eintragungen gemacht, die das vorliegende Dokument näher beschreiben. Dies geschieht einerseits über die Standard-Name-Spaces von XMP und andererseits werden bis zu drei eigene Name-Spaces erzeugt.

http://ns.profiforms.com/pfDoc/content/1.0/	(Beschreibung eines einzelnen Dokuments)
http://ns.profiforms.com/pfDoc/job/1.0/	(Beschreibung eines Druckjobs)
http://ns.profiforms.com/pfDoc/archive/1.0/	(Beschreibung der ArcRefFields - der Archiv-Index-Variablen)

PAPERCONSUMPTION

Die Ermittlung von Verbräuchen spielt bei der Erzeugung von Dokumenten eine zentrale Rolle. Für die Material Berechnung eines Druckzentrums steht dabei der Papierverbrauch der verwendeten Materialen im Mittelpunkt. Um die anfallenden Verbräuche bereits nach der Dokumenterzeugung zu kennen, erfolgt eine komplette Verbrauchsabrechnung pro Dokument bereits bei der Erzeugung. Das Ergebnis dieser Abrechnung wird als PaperConsumption-Report oder kurz als PaperConsumption bezeichnet.

Der PaperConsumption-Report ist Bestandteil eines jeden erzeugten Dokuments und lässt sich als DocRef/SAPRef oder als XTF ausgeben. Damit der PaperConsumption-Report in all diese Kanäle ausgegeben werden kann, muss dieser extrem kurz und ohne Zeilenumbrüche sein. Aus diesem Grund ist er nicht in Prosa sondern in einer IT-artigen Kurzform notiert, die man kennen muss, um den Report fehlerfrei wieder einlesen (parsen) zu können.

Der PaperConsumption-Report dokumentiert den Papierverbrauch des gesamten Dokuments. Dabei werden sowohl gedruckte Seiten, Umschläge (Envelopes) und selektive Beilagen (Insertations) aufgeführt. Die ReportTypen entsprechen einem Material mit seiner Verarbeitungsart und setzen sich zusammen aus:

a) Class (Simplex, Duplex, Insertion und Envelope)
b) PaperType oder Tray (optional)
c) PaperDescription (optional)

und listen auf, wie viel Einträge es pro ReportType gibt.

Für bedrucktes Papier ohne Angabe des PaperTypes bleibt der PaperType leer. Für Beilagen wird der Name der Beilage auf den PaperType gemappt. Ist der Name der Beilage leer, so wir die Tray-Nummer der Beilage auf den PaperType aufgeführt. Um beide Varianten voneinander unterscheiden zu können, erfolgt die Ausgabe des PaperTypes in Anführungszeichen. Besitzt ein PaperType neben seinem Namen noch eine PaperDescription, so wird diese als nachfolgender Parameter kommasepariert ebenfalls in Anführungszeichen mit ausgegeben.

Ein ReportType kann folgende Syntax haben:

```
AnzahlClass
AnzahlClass(Tray)
AnzahlClass("PaperType"["PaperDescription"])
```

Anzahl ist die Anzahl des Vorkommens des ReportType. Class ist 'S' für Simplex, 'D' für Duplex, 'I' für selektive Beilage (Insertion)und 'E' für Briefumschlag (Envelope). Tray ist die Nummer der selektiven Beilage. PaperType ist der PaperType des Papiers und PaperDescription die Description des PaperTypes.

Aller ReportTypen werden in einen String ausgegeben und mit einem ':' voneinander getrennt.

```
Beispiel:

4 Duplex-Blätter vom PaperType BLANKO
4D("BLANKO")

1 Simplex-Blatt mit leeren PaperType
1S

2 Duplex-Blätter und 1 Simplex-Blatt vom PaperType BLANKO
2D("BLANKO"):1S("BLANKO")

2 Duplex-Blätter PaperType BLANKO und 1 Simplex-Blatt
vom PaperType ZAHLSCHEIN
2D("BLANKO"):1S("ZAHLSCHEIN")

3 Simplex-Blätter und eine Selektive Beilage in Fach2
3S:1I(2)

4 Simplex-Blätter und eine Selektive Beilage "Flyer 371"
4S:1I("Flyer 371")

5 Simplex-Blätter mit PaperType und PaperDescription
5S("MYLOGO","New Smart Design")
```

MATERIAL

Jedes Blatt, jeder Briefumschlag und jede Beilage besitzen eine Materialbeschreibung, auf die im Verlauf der Erzeugung und Verarbeitung Einfluss und Bezug genommen werden kann. Die Materialbeschreibung findet z.B. Verwendung im PaperConsumption-Report, der postalischen Freimachung, der Produktionssteuerung, der OMR-Marken-Erzeugung und ist Bestandteil eines jeden Dokuments.

Die Materialbeschreibung oder kurz das Material besitzt eine ganze Menge von Eigenschaften, die für die Art und die Verarbeitung unterschiedlich Werte und Interpretationen erlauben. Sprich, ja nachdem ob ein Material in Blatt, ein Briefumschlag oder eine Beilage ist, wird die Matreialbeschreibung unterschiedlich initialisiert und es sind unterschiedliche Werte zulässig.

Damit nicht jedes Dokument das Material neu definieren muss, gibt es ein sogenanntes Standardmaterial für jeden Verwendungszweck. Für ein normales Blatt ist der Standardwert ein A4-Papier mit einem Grammatur von 80g/m^2 und dem Volumen 1. Diese Blatt ist faltbar. Für eine Beilage ist der Standardwert gleich dem normalen Blatt nur mit dem Unterscheid, dass dieses bereits gefaltet ist und drei Falzlagen besitzt. Für einen Briefumschlag ist der Standardwert DIN Lang (DL) mit einer Abmessung von 110x220 mm.

Damit die Materialbeschreibung platzsparend definiert und weitergegeben werden kann, erfolgt Definition in einem Textstring, der alle Werte beinhaltet. Dieser Textstring ist nicht in Prosa sondern in einer IT-artigen Kurzform notiert, die man kennen muss, um den Materialbeschreibung fehlerfrei wieder einlesen (parsen) zu können.

Eine Matreialbeschreibung ist eine mit Leerzeichen separierte Liste von Eigenschaften. Jede Eigenschaft beginnt mit einem Buchstaben, der eine bestimme Eigenschaft kennzeichnet, gefolgt von einem ':' als Separator. Nach dem Separator folgt der Wert, der der Eigenschaft zugewiesen wird. Die Eigenschaften dürfen weder einen ':' noch einen Leerzeichen besitzen.

Die Materialbeschreibung besitzt folgende Eigenschaften:

I	Description	Beschreibung Achtung: darf keine Leerzeichen, Tabulatoren oder ':' enthalten (optional)
M	MaterialID	MaterialID Achtung: darf keine Leerzeichen, Tabulatoren oder ':' enthalten (Pflicht, wenn nicht über den Tray gearbeitet wird, sonst optional)
N	Number	Anzahl der Materialien bei Briefumschlägen und normalen Blättern exakt 1 sonst 1 und größer (Standardwert ist 1) (optional)

T	Tray	Beilagen-Stationsnummer nur für Beilagen (Pflicht, wenn nicht über die MaterialID gearbeitet wird, sonst optional)
F	Format	Format A4,DINLANG, usw. (optional)
D	Dimension	Dimension wenn es nicht aus dem Format hervorgeht Syntax: WIDTHxHEIGHTxTHICKNESS (Pflicht, wenn kein Format verwendet wird, sonst optional)
W	Weight	Gewicht Angabe inkl. Einheit (Standardwert ist 5 Gramm) (optional)
O	FoldType	Falztyp NOFOLDS nicht gefalzt und nicht zu falzen (Falzlagen=1) ISFOLDABLE nicht gefalzt und kann gefalzt werden (Falzlagen=1) ISFOLDED ist gefalzt und nicht weiter zu falzen (Falzlagen>1) (optional)
L	Layer	Anzahl der Falzlagen Die Falzlage ist die Anzahl Papier, die aufeinander liegen. (optional)
P	Optional	Optional true oder false; wobei false Standardwert ist nur für Beilagen (optional)

ZUSÄTZLICHE AUSGABEVARIABLEN

Zu den wesentlichen Eigenschaften des OMS-ReportWriters gehört es, in den Ausgabe-Datenstrom zusätzliche Variablen auf Dokument-Ebene zu generieren, die beschreibenden Charakter haben. Diese Variablen werden, wenn sie im Design vorhanden sind, gemeinsam mit den Druckdaten ins Dokument ausgegeben. Auch Header- und Trailer-Seiten werden mit StatusVariablen versorgt, die durch Anlegen eines gleichnamigen Feldes im Design mit ausgegeben werden können. Die letzte Kategorie sind die Variablen, die in eine VOL-Datei hinein generiert werden. VOL-Dateien dienen zur Übergabe von Spooljob-Informationen an den OMS-Spooler.

Variablen auf Dokument-Ebene:

OSPAGE_NO	Nummer der aktuellen Seite in der OutSelection
OSPAGE_NA	Anzahl aller Seiten in der OutSelection
OSSHEET_NO	Nummer des aktuellen Blattes in der OutSelection
OSSHEET_NA	Anzahl aller Blätter in der OutSelection
OSDDPAGE_NO	Nummer der aktuellen Seite einer Kopie des DocDef in der OutSelection
OSDDPAGE_NA	Anzahl der Seiten einer Kopie des DocDef in der OutSelection
OSDDSHEET_NO	Nummer des aktuellen Blattes einer Kopie des DocDef in der OutSelection
OSDDSHEET_NA	Anzahl der Blätter einer Kopie des DocDef in der OutSelection
PAGE_NO	Nummer der aktuellen Seite im WorkItem
PAGE_NA	Anzahl aller Seiten im WorkItem
FRONTPAGE	Vorderseite 1 Rückseite 0
PAGEDESCRIPTOR	PAGEDESCRIPTOR des WorkItems (siehe WorkItem-Objekt)
CONTINUATIONTEXT	CONTINUATIONTEXT des WorkItems (siehe WorkItem-Objekt)
COPYTEXT	COPYTEXT des WorkItems (siehe WorkItem-Objekt)
ENVELOPESHEETS	Anzahl der Blätter des Dokuments im Kuvert inkl. Kanal1, Kanal2 und selektiven Beilagen
ENVELOPEWEIGHT	Gewicht des Kuverts inkl. Umschlag, Kanal1, Kanal2 und selektiven Beilagen
ENVELOPECLASS	Portoklasse des Kuverts bei Portoklassenberechnung ohne DV-Freimachung

DOCUMENTIDMATCH	Match-Code für mehrkanaligen Druck, wenn mit DocRef gearbeitet wird
CHANNEL	Kanal, in dem sich diese Seite befindet
CHANNELMATCH8	Match-Code für mehrkanaligen Druck, wenn die Kuvertierstraße mit drei Bit Matchcode arbeitet
CHANNELMATCH16	Match-Code für mehrkanaligen Druck, wenn die Kuvertierstraße mit vier Bit Matchcode arbeitet
CHANNELMATCH32	Match-Code für mehrkanaligen Druck, wenn die Kuvertierstraße mit fünf Bit Matchcode arbeitet
CHANNELMATCH64	Match-Code für mehrkanaligen Druck, wenn die Kuvertierstraße mit sechs Bit Matchcode arbeitet
FILE_DOC_NO	Nummer des Dokuments in der Datei
FILESHEET_NO	Nummer des Blattes in der Datei
FILEPAGE_NO	Nummer der Seite in der Datei
SERIES_DOC_NO	Nummer des Dokuments in der Serie
SERIESPAGE_NO	Nummer der Seite in der Serie
SERIESSHEET_NO	Nummer des Blattes in der Serie
TABLE_HIERARCHY	Liste der Tabellennamen, in dem diese SubForm steht
DOCVAR_*	Variablen, die aus den XDP- oder PDF-File kommen

Variablen in Header-, Trailer- und Indicant-Dokumenten:

FILENAME	Name des Files
JOBID	Job ID
PRINTER	Name des Druckers
CHANNEL2PRINTER	Name des Druckers für den Kanal2 (nur im Kanal1)
CHANNEL2FILENAME	Name des Files für den Kanal2 (nur im Kanal1)
CHANNEL1PRINTER	Name des Druckers für den Kanal1 (nur im Kanal2)
CHANNEL1FILENAME	Name des Files für den Kanal1 (nur im Kanal2)
QUALIFIER	QUALIFIER für die Stapelbildung (siehe DocDef)
CHANNEL	Kanal dieses Files
NO_SHEETS	Anzahl Seiten im File
NO_PAGES	Anzahl Blätter im File
NO_DOCS	Anzahl Dokumente im File
FILESERIALNO	Nummer des Files in einer Serie
STARTPAGEINSERIES	Startblatt des Files in der Serie
STARTSHEETINSERIES	Startseite des Files in der Serie
STARTDOCINSERIES	Startdokument des Files in der Serie
NO_CHANNEL1FILESINSERIES	Anzahl der Files im Kanal1 in der Serie
NO_CHANNEL2FILESINSERIES	Anzahl der Files im Kanal2 in der Serie
NO_FILESINSERIES	Anzahl der Files in der Serie
NO_CHANNEL1PAGESINSERIES	Anzahl der Seiten im Kanal1 in der Serie
NO_CHANNEL2PAGESINSERIES	Anzahl der Seiten im Kanal2 in der Serie
NO_PAGESINSERIES	Anzahl der Seiten in der Serie
NO_CHANNEL1SHEETSINSERIES	Anzahl der Blätter im Kanal1 in der Serie
NO_CHANNEL2SHEETSINSERIES	Anzahl der Blätter im Kanal2 in der Serie
NO_SHEETSINSERIES	Anzahl der Blätter in der Serie
NO_DOCSINSERIES	Anzahl der Dokumente in der Serie
PROCESSQUALIFIER	PROCESSQUALIFIER spezieller Prozesse wie der DV-Freimachung
PAPERTYPEn_NAME	Name des Papiertyps N
PAPERTYPEn_TRAY	Schacht des Papiertyps N
PAPERTYPEn_NO	Anzahl Blätter des Papiertyps N
CHANNEL1_FILE_DOC_NO	Nummer des Dokuments im Kanal1
SIMPLEXJOB	Datei mit nur einseitig bedruckten Dokumenten
CONTAINSARCHIVEDOCUMENTS	In dieser Datei sind Archive-Dokumente enthalten, die dem OMS-Archiver übergeben werden müssen

Variablen in der VOL-Datei und Dokument XMP:

RW_Printer	Name des Druckers
RW_JobID	Job ID
NumberPages	Anzahl Seiten im File
RW_No_Pages	Anzahl Seiten im File
RW_No_Sheets	Anzahl Blätter im File
RW_No_Docs	Anzahl Dokumente im File
RW_Channel	Kanal des Files
RW_FileSerialCount	Nummer des Files in der Serie
RW_ProcessQualifier	ProcessQualifier (wird von Prozessen wie der DV-Freimachung belegt)
RW_Qualifier	Qualifier für Stapelbildung
RW_RunName	Beschreibung des Stapels (z.B. Post-Produkt)
RW_Channel2Printer	Name des Druckes für den Kanal2
RW_Channel2FileLink	Name des Files für den Kanal2
RW_FileType	Typ des Files: DECENTRAL, CENTRAL, INDICANT, ARCHIVE
RW_PaperType_NAME	Anzahl der Blätter im Papiertyp NAME
RW_PaperTray_TRAY	Name des Papiertyps im Tray TRAY
RW_PaperTypes	kommaseparierte Liste aller verwendeten Papertyps
RW_Doc_*	Variablen, die aus dem Dokument kommen
RW_OutputType	OutputType ist das File-Format und kann PDF, XML oder MINE sein
RW_SimplexJob	Datei mit nur einseitig bedruckten Dokumenten
No_Channel1FilesInSeries	Anzahl der Dateien in der Serie, die für Kanal 1 generiert wurden
No_Channel2FilesInSeries	Anzahl der Dateien in der Serie, die für Kanal 2 generiert wurden
RW_VersandplanIDs_*	Kommaseparierte Liste aller enthaltenen Versandplan-IDs
RW_EAIDs	Kommaseparierte Liste aller enthaltenen EntgeltabrechnungsIDs
RW_KAID	KonsolidiererabrechnungsID
RW_ContainsArchiveDocuments_*	In dieser Datei sind Archive-Dokumente enthalten, die dem OMS-Archiver übergeben werden müssen

DocRef-Variablen:

CHANNEL1_JOBID	JobID der ersten Kanal1-Datei
CHANNEL2_JOBID	JobID der ersten Kanal2-Datei
CHANNEL1_FILE_DOC_NO	Nummer des Dokuments im Kanal1
CHANNEL2_FILE_DOC_NO	Nummer des Dokuments im Kanal2
CHANNEL1_PAGES	Seiten im Kanal1
CHANNEL2_PAGES	Seiten im Kanal2
CHANNEL1_SHEETS	Blätter im Kanal1
CHANNEL2_SHEETS	Blätter im Kanal2
PRINTED_PAGES	Seiten im Kanal1 + Kanal2
PRINTED_SHEETS	Blätter im Kanal1 + Kanal2
ADDITIONAL_SHEETS	Blätter aus der selektiven Beilage
ENVELOPEWEIGHT	Gewicht des Kuverts inkl. Umschlag, Kanal1, Kanal2 und selektiven Beilagen
DOCREF	Dokument Referenz aus dem DocDef
QUALIFIER	QUALIFIER für die Stapelbildung (siehe DocDef)
PROCESSQUALIFIER	PROCESSQUALIFIER spezieller Prozesse wie der DV-Freimachung
RUNNAME	Beschreibung des Stapels (z. B. Post-Produkt)
CHANNEL1_FILENAME	Name des Kanal1 Files
CHANNEL2_FILENAME	Name des Kanal2 Files
CHANNEL1_PRINTER	Drucker des Kanal1
CHANNEL2_PRINTER	Drucker des Kanal2
C1FROMPAGE	Von Seite im Kanal1 (vorbelegt durch C1PAGEOFFSET)
C1TOPAGE	Bis Seite im Kanal1 (vorbelegt durch C1PAGEOFFSET)
C2FROMPAGE	Von Seite im Kanal2 (vorbelegt durch C2PAGEOFFSET)
C2TOPAGE	Bis Seite im Kanal2 (vorbelegt durch C2PAGEOFFSET)
C1PAPERTYPEn_NAME	Dokument-bezogene Papertype-Angabe Name des n-ten Papiertyps im 1. Kanal
C1PAPERTYPEn_NO	Dokument-bezogene Papertype-Angabe Anzahl der Seiten des n-ten Papiertyps im 1. Kanal
C2PAPERTYPEn_NAME	Dokument-bezogene Papertype-Angabe Name des n-ten Papiertyps im 2. Kanal
C2PAPERTYPEn_NO	Dokument-bezogene Papertype-Angabe Anzahl der Seiten des n-ten Papiertyps im 2. Kanal
PAPER_CONSUMPTION	Der Inhalt der Variable dokumentiert den

Papierverbrauch des Gesamtdokuments.
Siehe Abschnitt PaperConsumption.

LIMITATIONEN

Verfahrensbedingte Limitationen

CPU-Nutzung

Da der OMS-ReportWriter einen Job von vorn nach hinten abarbeitet, kann kein ausgeprägtes Multi-Threading genutzt werden. Es kann sein, dass für IO-Prozesse mehrere Threads benutzt werden.
Im Allgemeinen muss aber davon ausgegangen werden, dass nur eine CPU genutzt wird.

Anzahl DV-Freimachungen pro Verfahren

Wird die DV-Freimachung im OMS-ReportWriter genutzt, so ergeben sich daraus verfahrensbedingte Einschränkungen. Der OMS-ReportWriter arbeitet mit der Datei entgelt.db, in der alle Freimachungs-informationen gespeichert werden. Da diese Datei immer nur von einem OMS-ReportWriter bedient werden kann, ist es wichtig, dass alle Freimachungsläufe, die auf eine entgelt.db gehen, sukzessiv hintereinander abgearbeitet werden. Als Alternative können mehrere entgelt.db-Dateien in unterschiedlichen Verzeichnissen bedient werden. Diese müssen dann aber auch unterschiedliche Verfahren im Sinne der DV-Freimachung bedienen.

Formatbedingte Limitationen

Einige Limitationen sind formatbedingt. So ist es im PDF-Format z. B. nicht zulässig, Dateien größer 7,6 GB anzulegen oder im PDF Inhaltsverzeichnisse zu erstellen, die größer als 1 GB sind. Nähere Informationen zum PDF-Format finden Sie in der aktuellen Dokumentation des PDF-Formats.
Die Übergabe von Variablen im PDF an den nachfolgenden Prozess kann über XMP und/oder Annotation-Variablen erfolgen. Im Gegensatz zu XMP gibt es für Annotation-Variablen eine Größenbeschränkung von 512 Zeichen Inhalt pro Variable. Andere verwendete Formate haben ebenso Limitationen, die beim Überschreiten zu Abbrüchen führen. Allerdings sehen wir uns an dieser Stelle außerstande, alle Limitationen zu recherchieren und dokumentieren.

Ressourcenbedingte Limitationen

Um eine hohe Verarbeitungsgeschwindigkeit zu erzielen, versucht der OMS-ReportWriter, möglichst viele Zwischendaten im Hauptspeicher (RAM) zu erzeugen und von dort zu verarbeiten. Dabei sind einige Werte wie die Anzahl der Seiten eines Dokuments oder die Anzahl der Dokumente pro DV-Freimachung besonders kritisch. Diese Art der Datenhaltung hat zur Folge, dass der Prozess abbricht, sobald kein weiterer Hauptspeicher verfügbar ist.

Da diese Art von Limitation nicht bei allen unterstützten Plattformen gleich ist, können an dieser Stelle keine pauschalen Werte angegeben werden. Bitte informieren Sie sich bei unserem Support über plattformbezogene Erfahrungswerte.

Plattformbedingte Limitationen

Je nach verwendeter Betriebssystemversion und je nach Compiler ergeben sich plattformbedingte Limitationen. So gibt es z. B. einige Plattformen, die keine Dateien größer 2GB verarbeiten oder erzeugen können. Weiterhin gibt es Einschränkungen in der Anzahl der Dateien, die geöffnet werden können. Diese sind teilweise durch Einstellungen im Betriebssystem veränderbar. Bitte befragen Sie unseren Support zu den bekannten plattformbedingten Limitationen Ihrer Plattform.

Teilumsetzungen

An einigen Stellen wurden nur Teile eines Standards oder einer Aufgabe umgesetzt. Dies geschah mit dem Ziel, die wichtigen Bestandteile des Standards oder der Aufgabe früher fertigstellen zu können. Dabei wurde bewusst auf Funktionalität verzichtet. Teilumsetzungen existieren vor allem bei der Interpretation von XFA und dem Adressoperator.

XFA

Aus dem XFA-Standard wurden vor allem die Elemente umgesetzt, die grafische Objekte repräsentieren, welche nicht interaktiv sind. Besonderes Augenmerk wurde auf Images, Texte, Felder, Numerische Felder, Datums-/Uhrzeit-Felder und Barcodes gelegt. Diese wurden zum Teil stark erweitert. Nicht implementiert wurden das Data-Binding und das Scripting. Ebenso fehlen alle interaktiven grafischen Objekte. Sizeable Texte und Subforms werden unter folgenden Einschränkungen unterstützt:

1. Variablen, die erst nach der Höhenberechnung ermittelt werden, können nicht zur Höhenberechnung herangezogen werden. Sollen diese Variablen in einem Subform ausgegeben werden, das sizeable ist, so müssen die Variablen in nicht-sizeable Feldern stehen.

2. Zur Höhenberechnung ist die Breite eine Voraussetzung, die die Höhe beeinflussen kann. Um zum Zeitpunkt der Höhenberechnung überhaupt eine Breite zu haben, wird die Breite der ersten ContentArea der FirstPage verwendet.

Links

Links im PDF sind Actions, die in einem PDF-Viewer (z.B. dem Acrobat Reader) ausgeführt werden können. Eine Link-Action öffnet beim Clicken in das Objekt die darin enthaltenen URL im Standard-Web-Browser. Da Links im PDF zu einer anderen Zeit geschrieben werden als die Seitenelemente, funktionieren diese nur auf Seiten korrekt, die nicht gedreht wurden.

Adressoperator

Das Datenstromformat BC-RDI arbeitet mit einer Datenstruktur, die zu einer Adresse eine Anzahl Felder bereitstellt, welche vom Präsentationsprogramm dann zu einer gültigen Adresse zusammengebaut werden müssen. Für ein einzelnes Locale ist dies eine überschaubare Aufgabe. Soll die Adressbildung für alle

HTML/XHTML-Textinterpretation

Neben dem XHTML das der Adobe LiveCycle-Designer erzeugt, wurde jetzt auch das Markup-XML des Textlayout-Frameworks des Adobe Flash-Players implementiert. Zusätzlich wurde die bereits bestehende XHTML-Implementierung deutlich um Funktionalität erweitert. Auch die Tags ältere HTML-Versionen werden jetzt verarbeitet, soweit diese den XML-Regeln entsprechen und jedes Tag geschlossen wird. Nicht umgesetzt sind im Moment:

Prozentwerte in Längen- und Größenangaben
Leading und Leading-Methode
Grafiken im Text
Kerning und Letterspacing
Direction
Vertical Alignment
Ligaturen

Die Textinterpretation erlaubt es bidirektionale Texte zu schreiben, wie diese im Arabischen und Hebräischen gebraucht wird. Jeder Paragraph beginnt mit der Schreiberichtung LEFT-TO-RIGHT.

Beginnt der Text mit Zeichen, die eine andere Schreiberichtung vorschreiben, so wird Schreiberichtung automatisch umgekehrt. Weiterhin kann die Schreiberichtung durch Zeichen im Text willkührlich beeinfluß werden. Dazu gibt es folgende Entities:

lre	LEFT-TO-RIGHT EMBEDDING
rle	RIGHT-TO-LEFT EMBEDDING
lro	LEFT-TO-RIGHT OVERRIDE
rlo	RIGHT-TO-LEFT OVERRIDE
pdf	POP DIRECTIONAL FORMATTING
lrm	LEFT-TO-RIGHT MARK
rlm	RIGHT-TO-LEFT MARK

INKOMPATIBILITÄTEN

Tabulatorsupport

Ab dem OMS-ReportWriter 5.4 ist ein XFA-Tabulatorsupport enthalten. Hier entsteht eine Inkompatibilität zu vorangegangenen Versionen bezüglich der Textpositionierung bei Texten, die Tabulatoren enthalten. Da bis einschließlich OMS-ReportWriter 5.3 keine Tabulatoren unterstützt wurden, waren Texte mit Tabulatoren immer falsch dargestellt. Es kann sein, dass sich Anwender an die falsche Darstellung gewöhnt haben und nun die richtige Darstellung als störend empfinden. Da Datenströme und Designs auf das Vorhandensein von Tabulatoren zu überprüfen sind, kann es evtl. zu einem Umstellungsaufwand kommen. Aufgrund der Komplexität der Umstellungen kann leider kein Schalter angeboten werden, der zwischen der alten und der neuen Verfahrensweise umschaltet.

Das Standard-Tabulator-Intervall beträgt ½ Zoll. In früheren Versionen gab es unterschiedliche Standard-Tabulator-Intervalle. Der Wert der Version 5.6 war 12,7pt. Soll ein alter Wert für das Standard-Tabulator-Intervall eingestellt werden, so kann dies unter reportw.ini/PDF/textes/TabInterval geschehen.

Inline-Sequenz NewLine

Bis jetzt wurde \n in einen Zeilenumbruch gewandelt, ohne den Mechanismus für Inline-Sequenzen zu bemühen. Jedes nachfolgende Zeichen blieb erhalten. In der JetForm-Logik ist \n aber eine Inline-Sequenz, die mit Space, mit '.' mit '\' oder Zeilen- bzw. Textende terminiert wird. Folgt nach dem \n keiner dieser Terminatoren, so wird kein \n erkannt.

```
Beispiel:

Hello\nWorld here I am.

Das Zeichen \ leitet eine Inline-Sequenz mit dem Namen nWorld ein.
Die Inline-Sequenz nWorld ist aber unbekannt.

Die Inline Sequenz \nWorld wird aus dem Datenstrom entfernt und es
wird eine Warnung ausgegeben.

Das Ergebnis ist mit dem OMS-ReportWriter 5.4:

Hello here I am.

Mit dem OMS-ReportWriter 5.3 wurde das \n als Zeilenumbruch
verstanden und das Ergebnis lautete:

Hello
World here I am.
```

UniqueAddressSourceFields

UniqueAddressSourceFields ist ein Schalter mit den Werten 0 und 1, wobei 1 der Standardwert ist. Ist der Schalter eingeschaltet, so werden für AddressSourceFields eindeutige Namen erzeugt. Werden die Sourcefelder einer Adresse mit ausgegeben und verwendet, so kommt es zu einer Inkompatibilität gegenüber der Vorgängerversion 5.3, die durch den Schalter UniqueAddressSourceFields wieder ausgeschaltet werden kann.

Seitenzähler

Im Gegensatz zu älteren Versionen werden die Seiten und Blätter der Header- und Trailer-WorkItems in der aktuellen Version mitberechnet. Dies wirkt sich auf viele Seitenzähler- und Papiertyp-Variablen aus, die in Header- und Trailer-WorkItems oder -vol-Dateien generiert werden.

Textplazierung nach AXTE

Als neue TextplacingMethod wurde jetzt AXTE zum neuen Standard. AXTE trifft nun die Zeilenabstände des Adobe LiveCycle Designers exakt. Wer in seinem Formular aber mit dem Fehler der TextplacingMethod Adobe7 gelebt hat, dem verrutschen jetzt vor allem Texte, die über viele Zeilen gehen. Hier muss entweder im Design nachgebessert werden oder die vorherige TextplacingMethod in der INI wieder eingestellt werden.

LIMITIERTE VERSIONEN

Die OMS-Produkte sind von ihrem Entwicklungsansatz dahingehend ausgelegt, den Anforderungen von High-Volume-Prozessen gerecht zu werden. Um auch "kleineren" Anwendern die Möglichkeit zu geben, die Funktionalität der Produkte nutzen zu können, gibt es leistungseingeschränkte Versionen, die einzig über die Seriennummer gesteuert werden. Im Allgemeinen wird die Limitierung über eine Geschwindigkeitsbeschränkung realisiert. Es ist aber auch möglich, die Limitierung an anderen Kenngrößen festzumachen.

Der OMS-ReportWriter kennt folgende limitierte Versionen:

50 ppm
200 ppm
500 ppm
Unlimited

Die eingeschränkten Versionen unterstützen nur die angegebene Geschwindigkeitsstufe, gemessen in ppm (pages per minute). Dabei werden alle erzeugten Seiten herangezogen, unabhängig davon, über welchen Ausgabekanal sie ausgegeben werden. Ist das Produkt schneller mit einem Datenstrom fertig als es das Speedlimit erlaubt, so werden Wartezyklen eingelegt. Der Wechsel zu einer größeren Geschwindigkeitsstufe oder einer unlimitierten Version ist jederzeit bei gleicher Funktionalität möglich. Dazu muss ein Upgrade erworben werden, das sich nur auf die Seriennummer auswirkt.

INDEX