



Inhalt

1 1.1	Einleitung	
1.1	Installation	
1.3	Programmaufruf	
1.3.1	Start über die tse:nit Programmoberfläche (interner Programmaufruf)	
1.3.2	Programmaufruf über das Windows Startmenü (externer Programmaufruf)	
1.4	Unterstützte tse:nit Datenbereiche, Schnittstellenbeschreibungen	
2	Programmbeschreibung	
2.1	Grundlegende Bildschirmgestaltung	
2.1.1	Programmoberfläche	
2.1.2	Menüleiste	
2.1.3	Symbolschaltflächen	
2.1.4	Datengrid	8
2.1.5	Objektauswahlfeld	9
2.1.6	Objektexplorer	
2.1.7	Feldanzeige	
2.1.8	Formeleingabefeld	
2.1.9	Aktionseingabefeld	13
3	Funktionsweise und Features	14
3.1	Allgemeine Vorgehensweise bei externem Programmaufruf	
3.2	Allgemeine Vorgehensweise bei internem Programmaufruf	16
3.3	Arbeiten mit Variablen	18
3.4	Arbeiten mit Aktionen	
3.5	Modifizierung der Objektstruktur	
3.6	Das Datei-Verbindungsobjekt	
3.7	Das Zusammenführen mehrerer Quelldateien	
3.8	Eingaben während der Scriptausführung, die Funktion UserInput()	
3.8.1	Verwendung der Funktion USERINPUT() zur Eingabe von Property-Werten	
3.8.2	Verwendung der Funktion USERINPUT() zur Eingabe von Variablen - Werten	
3.9	Erstellen einer Protokolldatei, die Funktion WRITETOLOG()	
4	Anhang	
4.1	Der Textimportassistent	
4.1.1	Einlesen von Daten im Format MIT TRENNZEICHEN	
4.1.2	Einlesen von Daten im Format FESTE BREITE	
4.1.3	Einlesen von Daten im EXCEL - Format	
4.2	Der Formel- / Aktionseditor	
4.3	Der Funktionseditor	
4.4	Debug-Modus	52



1 Einleitung

1.1 Allgemeine Programmbeschreibung

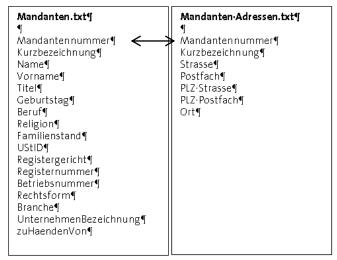
Bei dem Programm, TXTImport, handelt es sich um eine in tse:nit implementierte, frei programmierbare Schnittstelle, die es ermöglicht, Daten, die aus Fremdsystemen im ASCII/ANSI- bzw. Excel-Format1 bereitgestellt werden, in das tse:nit Austauschformat (Metafile - Format) zu konvertieren. Die konvertierte Datei erhält die Dateierweiterung "mta". Die Schnittstellendefinition für eine bestimmte Datenstruktur wird in einem Script einmalig durch den Anwender festgelegt und gespeichert und steht damit für weitere Importe zur Verfügung. Die Scriptdateien erhalten die Dateierweiterung "opt".

Es können Textdateien mit fester Breite der Spalten (sogenannten Feldern), Textdateien mit Trennzeichen, sogenannte CSV - Dateien oder Excel-Dateien in das tse:nit Austauschformat konvertiert werden.

Es kann mit Variablen, Funktionen und Prozeduren gearbeitet werden, wobei bei Funktionen ein eingeschränkter Sprachumfang von VBScript verwendet werden kann. Eine objektunterstützende Programmierung von Funktionen und Prozeduren ist nicht möglich.

Der TXTImport ermöglicht den Import von Quelldaten, die, entweder in einer einzelnen Datei vorliegen, oder aus mehr als einer Quelldatei zusammengeführt werden müssen. D.h. mit Hilfe des TXTImportes ist es möglich, mehrere Quelldateien zu einem Datenbestand zusammenzuführen. Die Zusammenführung der Quelldaten kann bedingungsabhängig durchgeführt werden.

Angenommen, die Mandantendaten liegen in zwei Dateien, Mandanten.txt und Mandanten Adressen.txt mit folgender Datenstruktur vor:



Zwischen diesen beiden Dateien wird über das Feld MANDANTENNUMMER eine Beziehung erstellt, so dass die Feldwerte der Datei MANDANTEN.TXT und die der Datei MANDANTEN ADRESSEN.TXT zusammengeführt werden, sofern der Wert des Feldes MANDANTENNUMMER in beiden Dateien übereinstimmt.

Abbildung 1

¹ ab tse:nit Version 2.73



Während der Konvertierung würden diese beiden Textdateien zu einer Datei Mandant.mta zusammengeführt, die dann in das tse:nit Mandatsverzeichnis importiert werden kann.

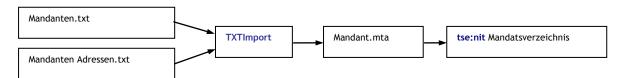


Abbildung 2

1.2 Installation

Eine gesonderte Installation des Programmes durch den Anwender ist nicht erforderlich. Die Installation erfolgt im Rahmen der tse:nit Installation bzw. des tse:nit Updates.

Zum Programmumfang gehören die Dateien

- TXTImport.exe
- srvTXTImport.dll

im tse:nit Installationsverzeichnis C:\ProgrammE\Wago-Curadata\10it\Bin².

1.3 Programmaufruf

Der Programmaufruf kann über die tse:nit Programmoberfläche (interner Programmaufruf) oder extern über das Windows Startmenü erfolgen.

1.3.1 Start über die tse:nit Programmoberfläche (interner Programmaufruf)

- 1. tse:nit starten
- 2. Öffnen des Dokumentes (z.B. KUNDENSTAMM)
- 3. Starten des TXTImportes über den Menüpunkt Datei | Importieren...
- 4. Auswahl des Importsystems Text-Datei-Import



Abbildung 3

1.3.2 Programmaufruf über das Windows Startmenü (externer Programmaufruf)



Abbildung 4

gültig ab Version 2.73 4_

² bei Standard Einzelplatzinstallation



1.4 Unterstützte tse:nit Datenbereiche, Schnittstellenbeschreibungen

Für die folgenden tse:nit Datenbereiche kann eine Konvertierung von ASCII/ANSI- bzw. Excel-Dateien in das tse:nit Austauschformat vorgenommen werden:

Datenbereich: tse:nit Dokumentpfad ³:

Kanzleiakte:

Mandanten
 DAUERAKTE\ALLGEMEINE STAMMDATEN\MANDATSVERZEICHNIS

Banken DAUERAKTE\ALLGEMEINE STAMMDATEN\BANKEN
 Finanzämter DAUERAKTE\ALLGEMEINE STAMMDATEN\FINANZÄMTER
 Gemeinden DAUERAKTE\ALLGEMEINE STAMMDATEN\GEMEINDEN
 Mitarbeiter DAUERAKTE\ALLGEMEINE STAMMDATEN\MITARBEITERLISTE

Mandanten- und Kanzleiakte:

Kunden
 Lieferanten
 Dauerakte\Grundlagen Rechnungswesen (eigenes ReWe)\Kundenstamm
 Dauerakte\Grundlagen Rechnungswesen (eigenes ReWe)\Lieferantenstamm

Stammkonten Jahresakte\Rechnungswesen\Stammdaten\Kontenstamm

Einzelbuchungen Jahresakte\Rechnungswesen\Finanzbuchhaltung\Buchungsliste

Verkehrszahlen Jahresakte\Rechnungswesen\Finanzbuchhaltung\Summen- und Saldenliste oder

JAHRESAKTE\RECHNUNGSWESEN\JAHRESABSCHLUSS\ HAUPTABSCHLUSSÜBERSICHT

Offene Posten
 Jahresakte\Rechnungswesen\OP Buchhaltung\Offene Posten Liste oder

JAHRESAKTE\RECHNUNGSWESEN\FINANZBUCHHALTUNG\SUMMEN- UND SALDENLISTE oder

JAHRESAKTE\RECHNUNGSWESEN\JAHRESABSCHLUSS\ HAUPTABSCHLUSSÜBERSICHT

Wirtschaftsgüter Jahresakte\Rechnungswesen\Anlagenbuchhaltung\ Abschreibungsverzeichnis

Die Schnittstellenbeschreibungen zu o.a. tse:nit Importformaten (Metafile, tse:nit Austauschformat) sind zum Teil auf der tse:nit Auslieferungs-DVD im Verzeichnis

<LW>\10iT_INFO\PRODUKTIONFORMATIONEN\SCHNITTSTELLEN als PDF-Dokumente abgelegt.

gültig ab Version 2.73 5_

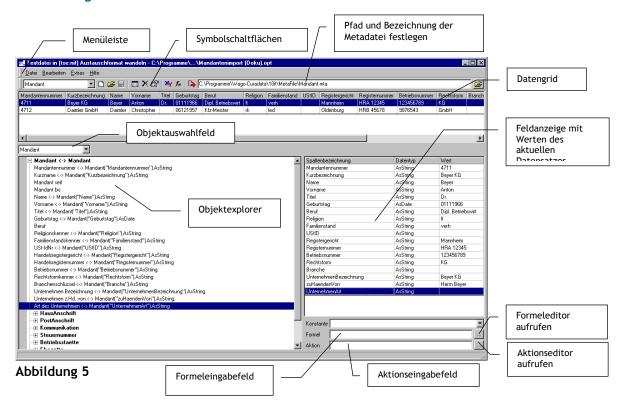
³ tse:nit Dokumentpfad nach Erstinstallation, Angabe in () für Kanzleiakte



2 Programmbeschreibung

2.1 Grundlegende Bildschirmgestaltung

2.1.1 Programmoberfläche



2.1.2 Menüleiste

DATE



- Erstellen eines neuen Scriptes
- Einlesen eines vorhandenen Scriptes
- Speichern des bearbeiteten Scriptes
- Speichern unter einem anderen Namen
- Neue Textdatei einlesen⁴
- Definition einer Textdatei entfernen⁵
- erstelltes Metafile anschauen
- Anzeige der bearbeiteten Scripte
- Programm beenden

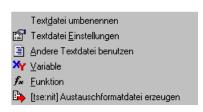
Abbildung 6

Starten des Textimportassistenten

⁵ Der logische Name der Textdatei wird aus dem Script entfernt.



BEARBEITEN



- Ändern des logischen Namens einer Textdatei
- Anzeige / Ändern der Textdatei Einstellungen⁶
- Auswahl der zu verarbeitenden Textdatei
- Variablen definieren
- Funktionen / Prozeduren bearbeiten
- tse:nit Austauschformat (Metafile) erzeugen

Abbildung 7

EXTRAS



- Anpassung der Objektstruktur
- Programmoptionen

Abbildung 8

EXTRAS | OBJEKTE



- neues Objekt erstellen
- Objekteigenschaften editieren
- Objekt löschen
- Objekt kopieren
- Objektvorschau mit den getroffenen Feldzuordnungen

Abbildung 9

EXTRAS | OPTIONEN

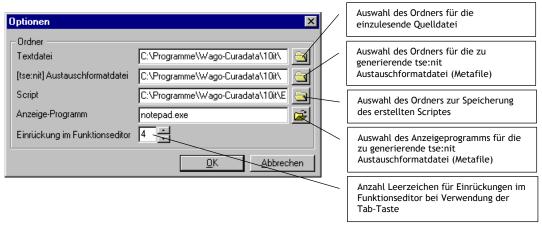


Abbildung 10

HILFE



- Anzeige der Hilfedatei

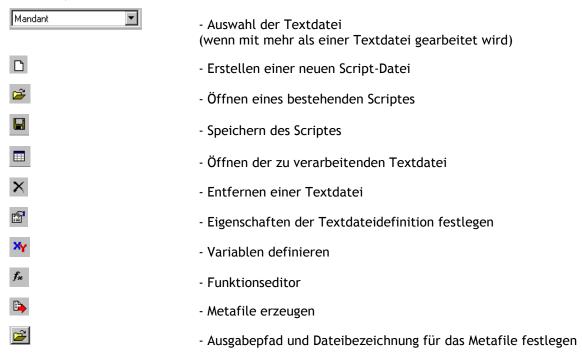
Abbildung 11

gültig ab Version 2.73 7_

⁶ abhängig vom Projektstatus (Dateidefinition bzw. Scripterstellung)



2.1.3 Symbolschaltflächen



2.1.4 Datengrid

Im Datengrid wird der Inhalt der Textdatei gemäß der getroffenen Felddefinitionen während des Textdateiimportes⁷ angezeigt.



Abbildung 12

Über das Kontextmenü im Grid können die Werte aus dem markierten Datensatz in die Feldanzeige übernommen werden, was für den Test von Formeln besonders hilfreich ist. Weiterhin besteht über das Kontextmenü die Möglichkeit, ganz gezielt nach bestimmten Werten im Quelldatenbestand (Inhalt der Textdatei) zu suchen.

Kontextmenü im Datengrid:



Abbildung 13

- Testwerte in die Feldanzeige übernehmen
- nach Wert suchen
- nächsten Wert suchen

⁷ siehe 4.1 Textimportassistent



2.1.5 Objektauswahlfeld



Abbildung 14

Über das Objektauswahlfeld erfolgt die Auswahl der zu erzeugenden Metafilestruktur.



Nach Auswahl des Datenbereiches (z.B. Mandant) wird die Metastruktur im Objektexplorer angezeigt und die Zuordnung der Datenfelder der Quelldatei zu den Properties des Metafiles kann, nachdem eine logische Verbindung mit der Textdatei vorgenommen wurde, erfolgen.

Abbildung 15

2.1.6 Objektexplorer

Im Objektexplorer erfolgt die Anzeige der zu erzeugenden Metafilestruktur mit den getroffenen Feldzuordnungen. Die Anzeige der Namen von Objekten und Properties (Eigenschaften der Objekte) kann wahlweise im Normaltext oder im tse:nit Austauschformat erfolgen. Die Umschaltung erfolgt über das Kontextmenü.

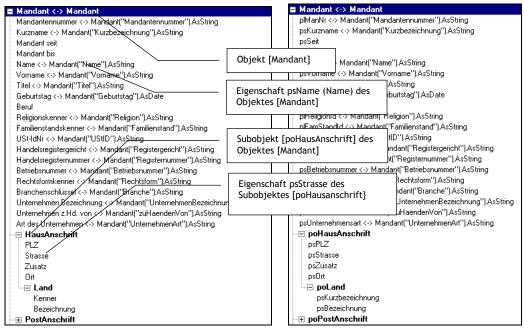


Abbildung 16

Abbildung 17

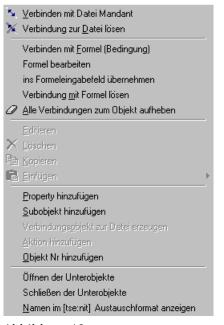
gültig ab Version 2.73 9_



Kontextmenüs im Objektexplorer

Abhängig von der ausgewählten Zeile in der Objektstruktur werden über die rechte Maustaste unterschiedliche Kontextmenüs angezeigt.

Auswahl steht auf der obersten Objektebene (z.B. MANDANT):

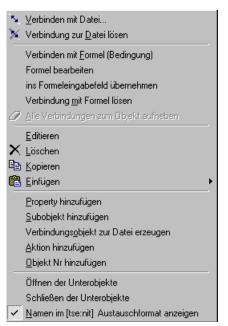


- logische Verbindung zur Datei erzeugen
- logische Verbindung zur Datei lösen
- Bedingung aus dem Formeleingabefeld übernehmen
- Formel (Bedingung) bearbeiten
- Formel (Bedingung) ins Formeleingabefeld übernehmen
- Formel (Bedingung) löschen
- alle getroffenen Zuordnungen werden gelöscht
- Hinzufügen eines Propertys zum Objekt
- Hinzufügen eines Unterobjektes zum Objekt
- Hinzufügen des Objektes ObjectNr
- Öffnen aller Unterobjekte (Expand)
- Schließen aller Unterobjekte (Collapse)
- Bezeichnungen werden im tse:nit Austauschformat gezeigt

Abbildung 18



Auswahl steht auf Subobjektebene (z.B. POHAUSANSCHRIFT):



- logische Verbindung zu einer weiteren Datei erzeugen⁸
- logische Verbindung zur Datei lösen
- Bedingung aus dem Formeleingabefeld übernehmen
- Formel (Bedingung) bearbeiten
- Formel (Bedingung) ins Formeleingabefeld übernehmen
- Formel (Bedingung) löschen
- Subobjekt bearbeiten⁹
- Subobjekt mit allen Properties löschen
- Subobjekt kopieren
- Subobjekt einfügen
- Hinzufügen eines Property zum Unterobjekt
- Hinzufügen eines Unterobjektes zum Unterobjekt
- Verbindungsobjekt zu einer Datei erzeugen
- Aktion aus dem Eingabefeld in die Struktur einfügen
- Hinzufügen des Objektes ObjectNr
- Öffnen aller Unterobjekte (Expand)
- Schließen aller Unterobjekte (Collapse)
- Bezeichnung werden im tse:nit Austauschformat gezeigt

Abbildung 19

Auswahl steht auf Property (z.B. PSORT):



- Property einem Datenfeld der Quelldatei zuordnen
- Property einer Konstante zuweisen
- Property durch Formel aus dem Eingabefeld errechnen
- Berechnungsformel bearbeiten
- aktuell verbundene Formel ins Eingabefeld übernehmen
- Verbindung Formel bzw. Datenfeld löschen
- Property bearbeiten
- Property löschen
- Aktion unterhalb des Property einfügen
- Öffnen aller Unterobjekte (Expand)
- Schließen aller Unterobjekte (Collapse)
- Bezeichnung werden im tse:nit Austauschformat gezeigt

Abbildung 20

⁸ bei Arbeit mit mehr als einer Textdatei

⁹ Bearbeiten von angezeigtem Namen, Namen im Metafile, Kommentar, Mehrfachauftreten zulassen



2.1.6.1 Das Objekt ObjectNr (Objekt Nr)

Das Objekt **ObjectNr** (Objekt Nr) wird zur Unterscheidung von mehreren Objekten / Objekteigenschaften mit gleichem Namen verwendet (Objekte oder Eigenschaften treten mehrfach auf).

Das Hinzufügen des Objektes ObjectNr hat folgende Auswirkungen:

Jedes Objekt / Objekteigenschaft erhält bei der Konvertierung in das Metafile-Format eine fortlaufende Nummer, die dieses Objekt von Objekten mit gleichem Namen unterscheidet. Die Verknüpfung (Wertzuweisung) zu dem Objekt ObjectNr erzielt man in Verbindung mit einer Zählvariablen unter Verwendung von einer Aktion, die in die Objektstruktur eingefügt wird. ¹⁰

2.1.7 Feldanzeige

In der Feldanzeige werden alle Datenfelder der Quelldatei mit ihrem festgelegten Datentyp und ihren aktuellen Werten des im Datengrid ausgewählten Datensatzes angezeigt.

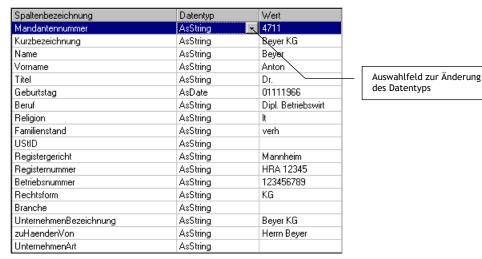


Abbildung 21

Durch Auswahl eines Feldes und Herüberziehen mit gedrückter linker Maustaste (Drag & Drop) in den Objektexplorer auf ein Property in der Objektstruktur erfolgt die Festlegung der Feldzuordnung zwischen der zuvor mit Hilfe des Textimportassistenten eingelesenen Quelldatei (Textdatei bzw. Excel-Datei). Weiterhin ist es in der Feldanzeige möglich, auch noch im Nachhinein den Datentyp eines Datenfeldes zu ändern.

Folgende Datentypen werden unterstützt:

Datentyp	Beispiel
Boolean	True, False
Currency	123.456.789,12; -123.456.789,12
Date	21.05.2002 (Werte zw. 01.01.100 u. 31.12.9999)
Float	123456789,12; -123456789,12
Integer	123456789; -123456789
String	"© 2011 ADDISON Software und Service GmbH"

¹⁰ siehe Kapitel 3.3 **Beispiel 1**



2.1.8 Formeleingabefeld



Abbildung 22

Das Formeleingabefeld dient zur Vorbereitung von Berechnungsformeln, die dann einem Property als Wert oder einem Objekt als Bedingung zugeordnet werden können. Über die Schaltfläche ■ wird der Formeleditor gestartet, mit dem komfortabel Berechnungsformeln erstellt werden können.

2.1.9 Aktionseingabefeld

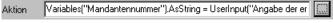


Abbildung 23

Das Aktionseingabefeld dient zur Vorbereitung von so genannten Aktionen (Formeln, Berechnungen, Eingaben des Anwenders) die dann in die Objektstruktur eingefügt werden und zur Laufzeit des Scriptes ausgeführt werden. So ist es zum Beispiel möglich, während der Scriptausführung Variablen zu berechnen und diese berechneten Variablen Properties zuzuweisen. Über die Schaltfläche wird der Aktionseditor gestartet.



3 Funktionsweise und Features

3.1 Allgemeine Vorgehensweise bei externem Programmaufruf

Den Konvertierungsprozess von Quelldaten (Textdateien), die in einfacher Datenstruktur in lediglich einer Quelldatei vorliegen, kann man, bei externem Programmaufruf, folgendermaßen beschreiben:

- 1. Starten des TXTImportes (siehe 1.3.2)
- 2. Auswahl des Datenbereiches über das Objektauswahlfeld (siehe 2.1.5)
- 3. Auswahl der Quelldatei (ASCII, ANSI, Excel) über den Textimportassistenten (Menü DATEI | NEUE TEXTDATEI)¹¹

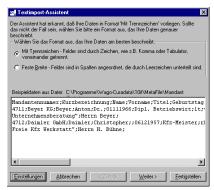


Abbildung 24

4. Festlegen der Datei- und Verarbeitungseinstellungen

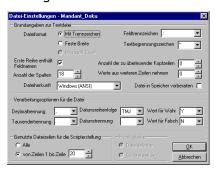


Abbildung 25

5. Festlegen der logischen Datei- und Feldnamen und der Felddatentypen.



Abbildung 26

¹¹ siehe 4.1 Der Textimportassistent



- 6. Einlesen der Quelldatei über Fertigstellen.
- 7. Über das Kontextmenü im Objektexplorer verbinden des Objektes (z.B. MANDANT) mit dem logischen Namen der eingelesenen Datei



Abbildung 27

- 8. Zuordnung der Datenfelder zu den Properties im Objektexplorer per Drag & Drop über die Feldanzeige und den Objektexplorer vornehmen
- 9. Erzeugung der tse:nit Austauschformatdatei (z.B. Mandant.mta) mittels der Schaltfläche 🕒
- 10. In tse:nit das entsprechende Dokument öffnen (z.B. MANDATSVERZEICHNIS in der Kanzleiakte) und die erzeugte Metadatei über den Menüpunkt Datei | IMPORTIEREN... als TSE:NIT AUSTAUSCHFORMAT einlesen



Abbildung 28



3.2 Allgemeine Vorgehensweise bei internem Programmaufruf

Den Konvertierungsprozess von Quelldaten (Textdateien), die in einfacher Datenstruktur in lediglich einer Quelldatei vorliegen, kann man, bei internem Programmaufruf, folgendermaßen beschreiben:

- 1. tse:nit starten und Öffnen des entsprechenden Dokumentes, in das Daten importiert werden sollen (z.B. MANDATSVERZEICHNIS)
- 2. Starten des TXTImportes über den Menüpunkt DATEI | IMPORTIEREN... | TEXT-DATEI-IMPORT (siehe 1.3.1)



Der Datenbereich (Objekt) ist bei internem Programmaufruf voreingestellt. Eine Änderung über das Objektauswahlfeld ist nicht möglich.

3. Auswahl der Quelldatei (ASCII, ANSI, Excel) über den Textimportassistenten (Menü DATEI | NEUE TEXTDATEI)

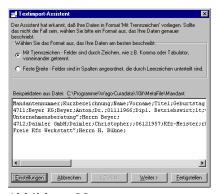


Abbildung 29

4. Festlegen der Datei- und Verarbeitungseinstellungen

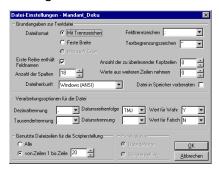


Abbildung 30

5. Festlegen der logischen Datei- und Feldnamen und der Felddatentypen.



Abbildung 31



- 6. Einlesen der Quelldatei über Fertigstellen.
- 7. Über das Kontextmenü im Objektexplorer verbinden des Objektes (z.B. MANDANT) mit dem logischen Namen der eingelesenen Datei



Abbildung 32

- 8. Zuordnung der Datenfelder zu den Properties im Objektexplorer per Drag & Drop über die Feldanzeige und den Objektexplorer vornehmen
- 9. Erzeugung der tse:nit Austauschformatdatei (z.B. Mandant.mta) mittels der Schaltfläche
- 10. Import der Daten über die Schaltfläche Übergehmen in der Metafileanzeige

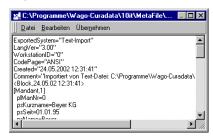


Abbildung 33



3.3 Arbeiten mit Variablen

In der Applikation TXT-Import ist es möglich, mit benutzerdefinierten Variablen zu arbeiten. Durch die Verwendung von Variablen kann eine Vereinfachung komplizierter Ausdrücke, die für die Auswertung von Bedingungen bzw. für die Verbindung von Feldern der Quelldatei mit Properties von Objekten verwendet werden, erreicht werden. Die definierten Benutzervariablen sind Bestandteil des Scriptes und werden mit diesem gespeichert.

Die Definition von Variablen erfolgt entweder über den Menüpunkt BEARBEITEN | VARIABLE oder über die Symbolschaltfläche **. Es öffnet sich das folgende Formular.

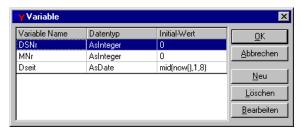


Abbildung 34

In diesem Beispiel sind drei Variablen definiert:

DSNr - Datensatznummer mit dem Initialwert 0 MNr - Mandantennummer mit dem Initialwert 0

Dseit - Datum seit (Mandant seit) Ermittlung des Initialwertes über die Kombination der

VBScript Funktionen mid() und now()
Bsp: now() = 22.05.02 14:35

Bsp: now() = 22.05.02 14:35 mid(now(),1,8) = 22.05.02

(Teilstring beginnend bei 1 mit der Länge 8)

Wie aus dem o.a. Beispiel ersichtlich wird, kann der Initialwert (Anfangswert der Variablen) entweder als eine Konstante festgelegt oder über eine Funktion ermittelt werden, wobei es **nicht** möglich ist, Verweise auf andere Variablen oder Felder von logischen Dateien zu verwenden.

Syntaxnotation für den Einsatz einer Benutzervariablen in Verknüpfungen, benutzerdefinierten Funktionen oder Prozeduren:

VARIABLES("VARIABLENNAME"). ASDATENTYP

ASDATENTYP bestimmt, welcher Datentyp der Variablen zugeordnet wird. Folgende Datentypen sind zugelassen: AsBoolean, AsCurrency, AsDate, AsFloat, AsInteger, AsString 12

¹² siehe auch 2.1.7, Datentypen



Beispiel 1

Variables("DSNr").AsInteger = Variables("DSNr").AsInteger + 1

Die Variable "DSNr" wird bei jedem Scriptdurchlauf um 1 erhöht. Mit Hilfe einer derartigen Variablen ist es zum Beispiel möglich, das Objekt **ObjectNr** ¹³ fortlaufend hochzuzählen.

Auszug aus der Objektstruktur:

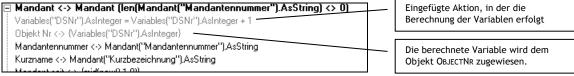


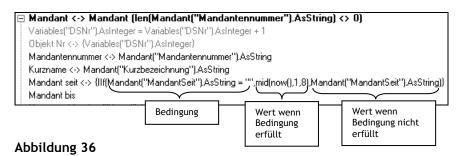
Abbildung 35

Beispiel 2

Bedingungsabhängige Verknüpfung eines Propertys unter Verwendung einer Variablen und eines Feldes einer logischen Datei.

Dem Property psSeit (Mandant seit) wird, abhängig vom Feldinhalt des Feldes "MandantSeit" der logischen Datei "Mandant", entweder der Feldinhalt (wenn Feldinhalt <> "") oder das aktuelle Systemdatum (wenn Feldinhalt = "") unter Verwendung der VBScript Funktion IIF() zugewiesen.

Auszug aus der Objektstruktur:



¹³ siehe auch 2.1.6, Kontextmenüs im Objektexplorer



3.4 Arbeiten mit Aktionen

Aktionen sind Anweisungen, die während des Konvertierungsprozesses in der Objektstruktur mit benutzerdefinierten Variablen ausgeführt werden. Die Erstellung von Aktionen kann entweder manuell im Aktionseingabefeld oder über den Aktionseditor erfolgen. Der Inhalt des Aktionseingabefeldes wird über das Kontextmenü im Objektexplorer in die Objektstruktur eingefügt, d. h. es wird so die Position im Konvertierungsprozess definiert (siehe Abbildung 37).

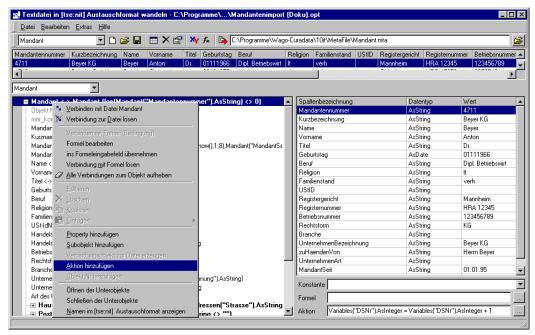


Abbildung 37

Der Menüpunkt AKTION HINZUFÜGEN im Kontextmenü des Objektexplorers ist nur aktiv, wenn eine Aktion im Aktionseingabefeld hinterlegt ist.

Durch den Menüpunkt AKTION HINZUFÜGEN wird eine neue Zeile mit der im Aktionseingabefeld hinterlegten Berechnungsformel in die Objektstruktur eingefügt. Die Position dieser Zeile bestimmt die Stelle im Script, an der die entsprechende Anweisung ausgeführt wird.

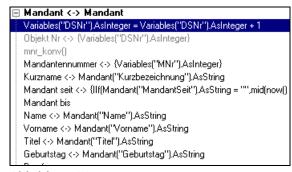


Abbildung 38

Im o.a. Beispiel (siehe Abbildung 38) wird die benutzerdefinierte Variable **DSNr** bei jedem Scriptdurchlauf um eins erhöht. Die über die Aktion

Variables("DSNr").AsInteger = Variables("DSNr").AsInteger + 1

gültig ab Version 2.73 20_



berechnete Variable wird in der nächsten Objekt-Zeile dem Objekt ObjectNr zugewiesen¹⁴.

Das Ergebnis der eingefügten Aktion sieht dann in der erzeugten Metadatei folgendermaßen aus:

```
ExportedSystem="Text-Import"
LangVer="3.00"
WorkstationID="0"
Code Page="ANSI"
Created="27.05.2002 16:33:17"
Comment=""
<Block, 27.05.02 16:33:17>
[Mandant,1] 😁
                                                Bei jedem Scriptdurchlauf wird die Objekt-
                                                nummer um 1 erhöht und dem Objekt als
[END]
                                                fortlaufende Nummerierung hinzugefügt...
[Mandant, 2] -
 [END]
<END>
END_OF_FILE
```

Abbildung 39

Als Aktionen können außer Berechnungsformeln auch benutzerdefinierte Funktionen¹⁵ in die Objektstruktur eingefügt werden, deren Berechnung dann während der Konvertierungsprozesses an der so definierten Position in der Objektstruktur ausgeführt werden. So wurde die benutzerdefinierte Funktion MNR_KONV() vor dem Property PLMANNR (Mandantennummer) in die Objektstruktur eingefügt (siehe Abbildung 38). Diese Funktion berechnet die benutzerdefinierte Variable MNR. Der Wert dieser berechneten Variablen wird dann in der nächsten Zeile dem Property plManNr (Mandantennummer) zugewiesen. Auf diese Art und Weise wurde eine Konvertierung ehemals vierstelliger Mandantennummern auf fünfstellige erreicht.

Das Ergebnis der als Aktion eingefügten Funktion sieht dann folgendermaßen aus:

```
ExportedSystem="Text-Import"
LangVer="3.00"
WorkstationID="0"
CodePage="ANSI"
Created="27.05.2002 16:33:17"
Comment=""
<Block, 27.05.02 16:33:17>
[Mandant, 1]
  plManNr=14711 ~
                                                  Die ehemals 4-Stelligen Mandantennum-
  . .
                                                  mern der Quelldaten wurden in 5-stellige
[END]
                                                  konvertiert.
                                                             4711 -> 14711
[Mandant, 2]
  plManNr=14712 -
                                                             4712 -> 14712
 [END]
<END>
END OF FILE
```

Abbildung 40

¹⁴ siehe auch Abschnitt 3.3 Arbeiten mit Variablen, **Beispiel 1**

¹⁵ Zur Erstellung von Funktionen siehe Abschnitt 4.3 Der Funktionseditor.



3.5 Modifizierung der Objektstruktur

Die über das Objektauswahlfeld festgelegte Objektstruktur kann modifiziert werden. So ist es möglich, einzelne Properties aber auch ganze Unterobjekte aus der Objektstruktur zu entfernen, wenn es dafür in den Quelldaten keine Entsprechung gibt und diese nicht benötigt werden. Die Objektstruktur wird dadurch vereinfacht und übersichtlicher. Es können aber auch Properties und Subobjekte hinzugefügt oder kopiert werden. Das Kopieren ist jedoch nur möglich, wenn in den Objekteigenschaften (Kontextmenü EDITIEREN) die Eigenschaft OBJEKT TRITT MEHRFACH AUF gesetzt ist.



Abbildung 41

Beispiel 3

In einer Quelldatei MANDANT KOMMUNIKATION. CSV mit folgendem Dateiinhalt

Mandantennummer;Kurzbezeichnung;TelefonFirma;TelefaxFirma;TelefonPrivat;TelefaxPrivat;Mobiltelefon 4711;Beyer KG;0123-12345;0123-12346;0123-7890;0123-7891;0170-123456789 4712;Daimler GmbH;0456-98765;0456-98766;0456-976410;0456-976422;0171-987654321

Abbildung 42

sind die Kommunikationsdaten der Mandanten hinterlegt. Um diese verschiedenen Kommunikationsdaten in tse:nit einzulesen, muss das Objekt POKOMMUNIKATION (Kommunikation) mehrfach kopiert werden. Im Objektexplorer wird die Objektzeile POKOMMUNIKATION (Kommunikation) markiert und über das Kontextmenü das Objekt kopiert und wieder eingefügt. Auf diese Art werden vier Kopien des Objektes POKOMMUNIKATION (Kommunikation) erstellt.

Kopieren und Einfügen eines Objektes:

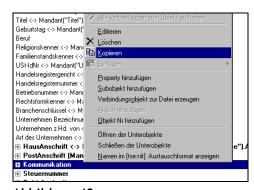


Abbildung 43

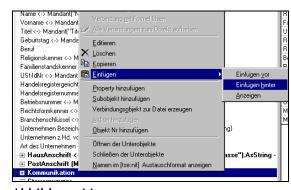


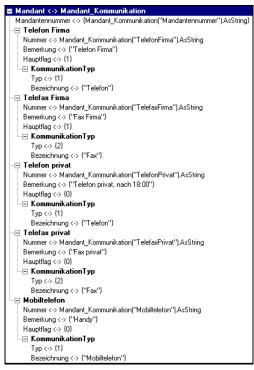
Abbildung 44

gültig ab Version 2.73 22_



Nachdem das Objekt POKOMMUNIKATION (Kommunikation) mehrfach kopiert, die Datei MANDANT KOMMUNIKATION.CSV über den Textimportassistenten eingelesen, die Datei mit dem Objekt Mandant verbunden und die Felder der Textdatei den Properties der Objekte zugeordnet wurden, ergibt sich die unten dargestellte Objektstruktur. Zur besseren Übersicht wurden die Anzeigenamen der Kommunikationsobjekte editiert und nicht benötigte Objekte und Properties aus der Objektstruktur entfernt, so dass die Feldzuordnung einfacher vorgenommen werden kann. Der tse:nit Name der Objekte bleibt jedoch unverändert, da eine Änderung beim Import zu einem Fehler führt.

Normalansicht:



Anzeige im tse:nit Austauschformat:

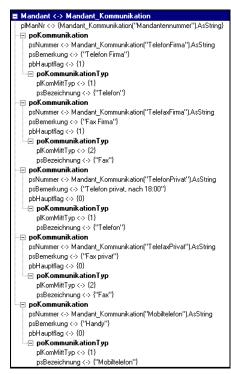


Abbildung 45

Abbildung 46

Die Ausführung des so erstellten Scriptes mittels der Symbolschaltfläche [5] liefert das auf der folgenden Seite dargestellte Metafile.



Modifizierte Objekte müssen unbedingt der tse:nit Schnittstellenbeschreibung¹⁶ entsprechen, da Formatverletzungen beim tse:nit Import zu Fehlern führen.

gültig ab Version 2.73 23_

¹⁶ siehe tse:nit Auslieferungs-CD im Verzeichnis <LW>\10it-Info\Produktinformation\Schnittstellen



Erstellte tse:nit Austauschformatdatei aus Beispiel 3:

```
ExportedSystem="Text-Import"
LangVer="3.00"
WorkstationID="0"
CodePage="ANSI"
Created="28.05.2002 13:27:33"
Comment=""
<Block, 28.05.02 13:27:33>
[Mandant]
  plManNr=4711
  [poKommunikation]
    psNummer=0123-12345
    psBemerkung=Telefon Firma
    pbHauptflag=1
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0123-12346
    psBemerkung=Fax Firma
    pbHauptflag=1
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
    psNummer=0123-7890
    psBemerkung=Telefon privat, nach 18:00
    pbHauptflag=0
    [poKommunikationTyp]
      {\tt plKomMittTyp=1}
      psBezeichnung=Telefon
  [END]
  [poKommunikation]
    psNummer=0123-7891
    psBemerkung=Fax privat
    pbHauptflag=0
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
    psNummer=0170-123456789
    psBemerkung=Handy
    pbHauptflag=0
    [poKommunikationTyp]
      {\tt plKomMittTyp=1}
      psBezeichnung=Mobiltelefon
    [END]
  [END]
[END]
[Mandant]
  plManNr=4712
  [poKommunikation]
    psNummer=0456-98765
    psBemerkung=Telefon Firma
    pbHauptflag=1
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0456-98766
    psBemerkung=Fax Firma
    pbHauptflag=1
    [poKommunikationTyp]
```

gültig ab Version 2.73 24_



```
plKomMittTyp=2
    psBezeichnung=Fax
[END]
  [END]
  [poKommunikation]
    psNummer=0456-976410
    psBemerkung=Telefon privat, nach 18:00
    pbHauptflag=0
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0456-976422
    psBemerkung=Fax privat
    pbHauptflag=0
    [poKommunikationTyp]
     plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
    psNummer=0171-987654321
    psBemerkung=Handy
    pbHauptflag=0
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Mobiltelefon
    [END]
  [END]
[END]
<END>
END_OF_FILE
```

gültig ab Version 2.73 25_



3.6 Das Datei-Verbindungsobjekt

Das Datei-Verbindungsobjekt ist ein virtuelles Objekt, das zur Verbindung von Unterobjekten mit Textdateien verwendet wird. Es ermöglicht, die Erstellung der Bedingungen für die Verbindung von Objekten zu Quelldateien zu vereinfachen. Bei diesem Objekt handelt es sich um ein spezielles Objekt, das nicht in das tse:nit-Austauschformat exportiert wird, sondern dass lediglich der Scriptsteuerung dient. Zur Veranschaulichung der Wirkungsweise soll das folgende Beispiel dienen.

Beispiel 4

Die Kommunikationsdaten (Telefon, Fax) für die Mandanten liegen einer Datei mit folgendem Inhalt vor:

```
Mandantennummer;Kurzbezeichnung;Kommunikationsnummer;Kommunikationstyp;Bemerkung
4711;Beyer KG;0123-12345;TelefonFirma;
4711;Beyer KG;0123-12346;TelefaxFirma;
4711;Beyer KG;0123-7890;TelefonPrivat;nach 18:00
4711;Beyer KG;0123-7891;Telefax;
4711;Beyer KG;0170-123456789;Mobiltelefon;
4711;Beyer KG;0170-123456789;Mobiltelefon;
4712;Daimler GmbH;0456-98765;TelefonFirma;
4712;Daimler GmbH;0456-98766;TelefaxFirma;
4712;Daimler GmbH;0456-976410;TelefonPrivat;nur im Notfall
4712;Daimler GmbH;0456-976422;TelefaxPrivat;
4712;Daimler GmbH;0456-976422;TelefaxPrivat;
```

Abbildung 47

Zum Import dieser Daten empfiehlt sich der Einsatz des Datei Verbindungs-Objektes, da in diesem Fall die zu einem Mandanten gehörenden Kommunikationsdaten nicht in Spalten sondern zeilenweise hinterlegt sind.

gültig ab Version 2.73 26_



Es wird ein Script mit folgendem Aufbau erstellt, wobei auch hier nicht benötigte Unterobjekte und Properties zur besseren Übersicht aus der Objektstruktur entfernt wurden. Außerdem wurde in diesem Beispiel mit einer weiteren Textdatei (MANDANT) gearbeitet, die lediglich als Mandantenverzeichnis dient. ¹⁷ Die Kommunikationsdaten wurden über den Textimportassistenten in die logische Datei MANDANT_KOMMDATEN und das Mandatsverzeichnis in die logische Datei MANDANT eingelesen.



Die logische Datei Mandant wurde über das Kontextmenü mit dem Objekt Mandant verbunden.

Es wurde über das Kontextmenü ein Datei Verbindungsobjekt über dem Objekt Kommunikation eingefügt, das über das Kontextmenü mit der Datei Mandant_KommDaten verbunden wurde. Als Verbindungsformel wurde Mandant("Mandantennummer").AsString =

MANDANT("MANDANTENNUMMER").ASSTRING =
MANDANT_KOMMDATEN("MANDANTENNUMMER").ASSTRING
hinterlegt.

Für jeden Kommunikationstyp (TelefonFirma, TelefaxFirma, TelefonPrivat, TelefaxPrivat, Mobiltelefon) wurde eine Kopie des Objektes KOMMUNIKATION in der Objektstruktur erstellt und bei den einzelnen Kopien eine Bedingung zur Erzeugung des Objektes hinterlegt (Doppelklick auf das Objekt und Eingabe der Bedingungsformel z.B.

MANDANT_KOMMDATEN("KOMMUNIKATIONSTYP").ASSTRING = "TELEFONFIRMA")

Abbildung 48

Die einzelnen Properties der Objekte wurden mit den entsprechenden Feldern der Datei MANDANT_KOMMDATEN verbunden.

gültig ab Version 2.73 27_

¹⁷ siehe auch Abschnitt 3.7 Das Zusammenführen mehrerer Quelldateien



Als Ergebnis von Beispiel 4 wird bei der Ausführung des Scriptes folgendes Metafile erstellt:

```
ExportedSystem="Text-Import"
LangVer="3.00"
WorkstationID="0"
CodePage="ANSI"
Created="30.05.2002 13:22:34"
Comment=""
<Block, 30.05.02 13:22:34>
[Mandant]
  plManNr=4711
  [poKommunikation]
    psNummer=0123-12345
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0123-12346
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
    psNummer=0123-7890
    psBemerkung=nach 18:00
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0170-123456789
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
[END]
[Mandant]
  plManNr=4712
  [poKommunikation]
    psNummer=0456-98765
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0456-98766
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
    psNummer=0456-976410
    psBemerkung=nur im Notfall
    [poKommunikationTyp]
      {\tt plKomMittTyp=1}
      psBezeichnung=Telefon
    [END]
  [poKommunikation]
    psNummer=0456-976422
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
```

gültig ab Version 2.73 28_



```
psNummer=0171-987654321
[poKommunikationTyp]
    plKomMittTyp=1
    psBezeichnung=Telefon
[END]
[END]
[END]
<END>
END_OF_FILE
```



3.7 Das Zusammenführen mehrerer Quelldateien

In der Applikation TXTImport ist es möglich, Informationen aus mehreren Textdateien in eine tse:nit-Austauschformatdatei zusammenzuführen. Die Vorgehensweise dazu lässt sich folgendermaßen beschreiben:

- Einlesen der ersten Datei über den Menüpunkt DATEI | NEUE TEXTDATEI (Textimportassistent)
- Verbinden des Hauptobjektes mit dieser Textdatei (Kontextmenü der Objektstruktur, Menüpunkt
 Verbinden mit Dalei xxx)
- Einlesen weiterer Textdateien über den Menüpunkt Datei | Neue Textdatei (Textimportassistent)
- Verbinden der weiteren Textdateien mit den Unterobjekten des Hauptobjektes (Kontextmenü der Objektstruktur, Menüpunkt 🔼 Verbinden mit Datei...)

Folgendes Beispiel soll die Arbeit mit mehreren Dateien veranschaulichen.

Beispiel 5

Die Mandantendaten liegen in folgenden Dateien mit folgendem Inhalt vor:

• Mandant Adressen. CSV (allgemeine Mandantenadressdaten)

```
Mandantennummer;Kurzbezeichnung;Strasse;Postfach;PLZ_Strasse;PLZ_Postfach;Ort
4711;Beyer KG;An der Mühlenbreite;3520;12345;12346;Imlande
4712;Daimler GmbH;Zur Stadtmauer;9911;78945;78946;Nirgendwo
```

Abbildung 49

Mandant Kommunikation.CSV (Kommunikationsdaten der Mandanten)

```
Mandantennummer;Kurzbezeichnung;Kommunikationsnummer;Kommunikationstyp;Bemerkung
4711;Beyer KG;0123-12345;TelefonFirma;
4711;Beyer KG;0123-12346;TelefaxFirma;
4711;Beyer KG;0123-7890;TelefaxFirma;
4711;Beyer KG;0123-7891;Telefax;
4711;Beyer KG;0170-123456789;Mobiltelefon;
4712;Daimler GmbH;0456-98765;TelefonFirma;
4712;Daimler GmbH;0456-98766;TelefaxFirma;
4712;Daimler GmbH;0456-976410;TelefonPrivat;nur im Notfall
4712;Daimler GmbH;0456-976422;TelefaxPrivat;
4712;Daimler GmbH;0456-976422;TelefaxPrivat;
```

Abbildung 50

Mandant Bank.CSV (Bankverbindungen der Mandanten)

```
Mandantennummer;Kurzbezeichnung;BLZ;BankKurzname;Bankname;Kontonummer
4711;Beyer KG;58550130;Sparkasse;Sparkasse Trier (Züsch);801411900
4711;Beyer KG;72050101;Kreissparkasse;Kreissparkasse Augsburg (Zusmarshausen);8999777
4711;Beyer KG;72069274;Raiffeisenbank;Raiffeisenbank Zusmarshausen-Altenmünster;9111333
4712;Daimler GmbH;57051870;Kreissparkasse;Kreissparkasse (Alf);1000200
4712;Daimler GmbH;58761343;Raiffeisenbank;Raiffeisenbank Zeller Land (Alf);5000000
```

Abbildung 51

Über ein geeignetes Script sollen diese drei Textdateien zu einer tse:nit Austauschformatdatei zusammengefasst werden.



Die Quelldaten sollten immer in sortierter Form vorliegen.

gültig ab Version 2.73 30_



Im ersten Schritt wird die Datei MANDANT ADRESSEN.CSV über den Textimportassistent in das Datengrid eingelesen und über das Kontextmenü des Objektexplorers mit dem Hauptobjekt MANDANT verbunden

Leich Mandant Adressen

Nicht benötigte Properties werden entfernt, so dass die folgende
Objektstruktur entsteht. Da die Datei MANDANT ADRESSEN in diesem Beispiel mit dem Hauptobjekt MANDANT verbunden wurde, dient sie gleichzeitig als Verzeichnisdatei für die Verbindung mit den weiteren Ouelldateien.



Abbildung 52

Das Property Mandantennummer (plManNr) wird mit dem Feld Mandantennummer der Adressdatei verbunden.

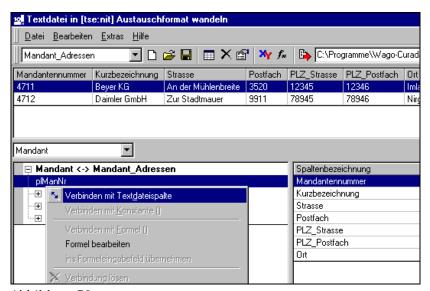


Abbildung 53

Die Objektstruktur stellt sich jetzt so dar.

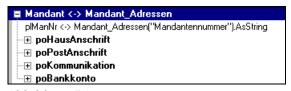


Abbildung 54

Im nächsten Schritt werden die weiteren Textdateien über den Menüpunkt Datei | NEUE TEXTDATEI eingelesen und die logischen Namen, MANDANT_KOMMDATEN und MANDANT_BANK vergeben. Über das Textdatei-Auswahlfeld in der Symbolleiste können jetzt die eingelesenen Dateien ausgewählt und deren Inhalt im Datengrid dargestellt werden.



Abbildung 55



Um das gewünschte Ergebnis zu erzielen, sind noch einige Änderungen an der Objektstruktur notwendig. Das Objekt HAUSANSCHRIFT wird markiert und über den Kontextmenüpunkt Verbindungsobjekt zur Datei erzeugen ein Verbindungsobjekt über diesem eingefügt.

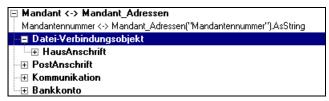


Abbildung 56

Das Objekt HAUSANSCHRIFT ist damit um eine Hierarchiestufe tiefer gesetzt worden. Um jetzt das Objekt POSTANSCHRIFT ebenfalls als Unterobjekt des Datei Verbindungs-Objektes zu erhalten, wird es an der jetzigen Position gelöscht



Abbildung 57

und als neues Subobjekt des Datei Verbindungs-Objektes eingefügt (Kontextmenüpunkt Subobjekt hinzufügen).



Abbildung 58

Die Objektstruktur sieht dann so aus:



Abbildung 59

In der weiteren Bearbeitung werden die Properties mit den Feldern verbunden und ggf. Formeln hinterlegt (siehe Objekt POSTANSCHRIFT, Property STRASSE in der folgenden Abbildung).

gültig ab Version 2.73 32_



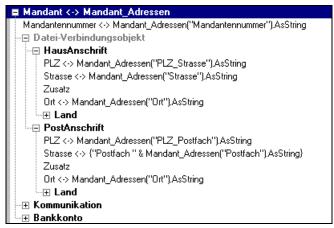
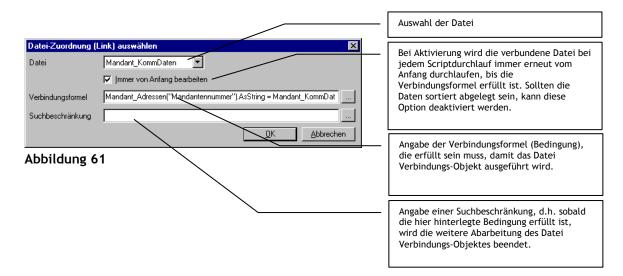


Abbildung 60

Im weiteren Verlauf der Scripterstellung wird über dem Objekt Kommunikation ein Datei Verbindungs-Objekt erzeugt und dieses Datei Verbindungs-Objekt über das Kontextmenü Schlieber werbunden. Als Verbindungsformel wird

MANDANT_ADRESSEN("MANDANTENNUMMER"). AsSTRING = MANDANT_KOMMDATEN("MANDANTENNUMMER"). AsSTRING hinterlegt.



Um alle Kommunikationstypen einzulesen, wird für jeden Typ eine Kopie erstellt und für deren Ausführung eine Bedingung hinterlegt.

(z.B. MANDANT_KOMMDATEN("KOMMUNIKATIONSTYP").AsSTRING = "TELEFONFIRMA" wenn das Feld KOMMUNIKATIONSTYP der Datei MANDANT_KOMMDATEN den Wert "TelefonFirma" hat)

gültig ab Version 2.73 33_



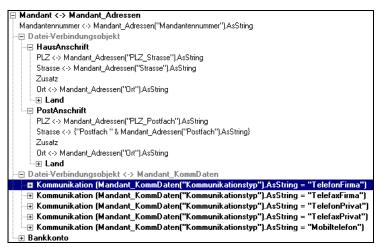


Abbildung 62

Die Properties der Objekte KOMMUNIKATION werden mit den Feldern verbunden. Die Objektstruktur sieht dann folgendermaßen aus:



Abbildung 63

Im Weiteren wird das Objekt BANKKONTO mit der Datei MANDANT_BANK verbunden. Als Verbindungsformel wird hier ebenfalls

MANDANT_ADRESSEN("MANDANTENNUMMER").ASSTRING = MANDANT_KOMMDATEN("MANDANTENNUMMER").ASSTRING

hinterlegt und die Feldzuordnung der Properties des Objektes BANKKONTO zu den Feldern der Datei MANDANT_BANK vorgenommen.

gültig ab Version 2.73 34_



Abbildung 64

Damit ist die Scripterstellung abgeschlossen und die Scriptausführung über die Symbolschaltfläche liefert als Ergebnis die tse:nit Austauschformatdatei auf der folgenden Seite.

gültig ab Version 2.73 35_



tse:nit Austauschformatdatei nach Ausführung des Scriptes aus Beispiel 5:

```
ExportedSystem="Text-Import"
LangVer="3.00"
WorkstationID="0"
CodePage="ANSI"
Created="30.05.2002 18:12:03"
Comment=""
<Block, 30.05.02 18:12:03>
[Mandant]
  plManNr=4711
  [poHausAnschrift]
    psPLZ=12345
    psStrasse=An der Mühlenbreite
    psOrt=Imlande
  [END]
  [poPostAnschrift]
    psPLZ=12346
    psStrasse=Postfach 3520
    psOrt=Imlande
  [END]
  [poKommunikation]
    psNummer=0123-12345
    pbHauptflag=1
    [poKommunikationTyp]
      {\tt plKomMittTyp=1}
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0123-12346
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
  [END]
  [poKommunikation]
    psNummer=0123-7890
    psBemerkung=nach 18:00
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0170-123456789
    [poKommunikationTyp]
      plKomMittTyp=1
    psBezeichnung=Telefon [END]
  [END]
  [poBankkonto]
    psKontonummer=801411900
    [poBank]
      psBLZ=58550130
      psName=Sparkasse Trier (Züsch)
      psKurzname=Sparkasse
    [END]
  [END]
  [poBankkonto]
    psKontonummer=8999777
    [poBank]
      psBLZ=72050101
      psName=Kreissparkasse Augsburg (Zusmarshausen)
      psKurzname=Kreissparkasse
    [END]
  [END]
  [poBankkonto]
    psKontonummer=9111333
    [poBank]
      psBLZ=72069274
      psName=Raiffeisenbank Zusmarshausen-Altenmünster
      psKurzname=Raiffeisenbank
    [END]
  [END]
[END]
[Mandant]
  plManNr=4712
```

gültig ab Version 2.73 36_



```
[poHausAnschrift]
   psPLZ=78945
    psStrasse=Zur Stadtmauer
    psOrt=Nirgendwo
  [END]
  [poPostAnschrift]
   psPLZ=78946
    psStrasse=Postfach 9911
   psOrt=Nirgendwo
  [END]
  [poKommunikation]
   psNummer=0456-98765
    pbHauptflag=1
    [poKommunikationTyp]
     plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0456-98766
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
   psNummer=0456-976410
    psBemerkung=nur im Notfall
    [poKommunikationTyp]
      plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poKommunikation]
    psNummer=0456-976422
    [poKommunikationTyp]
      plKomMittTyp=2
      psBezeichnung=Fax
    [END]
  [END]
  [poKommunikation]
    psNummer=0171-987654321
    [poKommunikationTyp]
     plKomMittTyp=1
      psBezeichnung=Telefon
    [END]
  [END]
  [poBankkonto]
   psKontonummer=1000200
    [poBank]
      psBLZ=57051870
      psName=Kreissparkasse (Alf)
   psKurzname=Kreissparkasse [END]
  [END]
  [poBankkonto]
    psKontonummer=5000000
    [poBank]
      psBLZ=58761343
      psName=Raiffeisenbank Zeller Land (Alf)
      psKurzname=Raiffeisenbank
    [END]
  [END]
[END]
<END>
END_OF_FILE
```

gültig ab Version 2.73 37_



3.8 Eingaben während der Scriptausführung, die Funktion UserInput()

Die Funktion UserInput() ermöglicht es dem Anwender, Eingaben während der Scriptausführung vorzunehmen. Die Funktion USERINPUT() besitzt eine variable Parameteranzahl.

Syntaxnotation:

USERINPUT(PROMPT[, PROMPTFELDNAME1, PROMPTFELDNAME2, ...])

PROMPT Textabfrage (Angabe in Hochkomma), die dem Anwender angezeigt wird PROMPTFELDNAME Feldnamen aus der aktuellen Datei, deren Werte angezeigt werden sollen

Der Aufruf der Funktion USERINPUT() erfolgt über den Formeleditor.

3.8.1 Verwendung der Funktion USERINPUT() zur Eingabe von Property-Werten

Beispiel 6

Es liegen Bankendaten in einer Datei mit folgendem Inhalt vor:

```
BankKurzname;Bankname
Sparkasse;Sparkasse Trier (Züsch)
Kreissparkasse;Kreissparkasse Augsburg (Zusmarshausen)
Raiffeisenbank;Raiffeisenbank Zusmarshausen-Altenmünster
Kreissparkasse;Kreissparkasse (Alf)
Raiffeisenbank;Raiffeisenbank Zeller Land (Alf)
```

Abbildung 65

In dieser Datei sind lediglich die Bankbezeichnungen und Bankkurzbezeichnungen hinterlegt. Während der Konvertierung in das tse:nit Austauschformat sollen die Bankleitzahlen durch Eingaben des Anwenders ergänzt werden. Dazu wird die Funktion USERINPUT() verwendet.

Im Objektauswahlfeld wird das Objekt BANK als Hauptobjekt ausgewählt. Die Textdatei wird über den Menüpunkt DATEI | NEUE TEXTDATEI in das Datengrid eingelesen und der logische Dateiname BANKEN vergeben. Das Objekt BANK wird mit der Datei BANKEN verbunden und die Zuordnungen für die Felder BANKNAME und BANKKURZNAME (Name) zu den Properties PSNAME und PSKURZNAME (Bezeichnung) vorgenommen. Die Objektstruktur sieht jetzt folgendermaßen aus:



Abbildung 66

Durch Doppelklick auf das Property PSBLZ (BLZ) wird der Formeleditor geöffnet und aus der Liste der Funktionen kann die Funktion USERINPUT() ausgewählt und in die Eingabezeile übernommen werden. In der folgenden Abbildung wurde im Formeleditor die Funktion USERINPUT() mit den Parametern "BANKLEITZAHL" als Prompt und als PromptFeldName wird das Datei-Feld BANKEN("BANKNAME"). ASSTRING verwendet.

gültig ab Version 2.73 38_



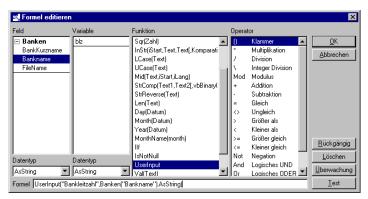


Abbildung 67

Durch betätigen der Schaltfläche Lest kann die Funktion überprüft werden.





Abbildung 68

Abbildung 69

Mit der Schaltfläche im Formeleditor wird die Formel aus der Eingabezeile des Formeleditors dem Property zugeordnet.



Abbildung 70

Bei Verwendung der Funktion USERINPUT() in einem Script durchläuft der Konvertierungsprozess zwei Phasen. In der 1. Phase werden die Textdateien durchlaufen und eine Liste der einzugebenden Werte wird erstellt und in einer Tabelle angezeigt. Der Anwender kann darin die notwendigen Angaben vornehmen.



Abbildung 71

Nach betätigen der Schaltfläche erfolgt dann in der 2. Phase die eigentliche Konvertierung in das tse:nit Austauschformat. Als Ergebnis der Ausführung des so erstellten Scriptes wird das Metafile auf der folgenden Seite erstellt.

gültig ab Version 2.73



tse:nit Austauschformat Datei aus Beispiel 6:

```
ExportedSystem="Text-Import"
LangVer="3.00"
WorkstationID="0"
CodePage="ANSI"
Created="03.06.2002 14:23:20"
Comment=""
<Block, 03.06.02 14:23:20>
[Bank]
 psBLZ=12345678
  psName=Sparkasse Trier (Züsch)
  psKurzname=Sparkasse
[END]
[Bank]
  psBLZ=78945612
  psName=Kreissparkasse Augsburg (Zusmarshausen)
  psKurzname=Kreissparkasse
[END]
[Bank]
 psBLZ=32165498
  psName=Raiffeisenbank Zusmarshausen-Altenmünster
  psKurzname=Raiffeisenbank
[END]
[Bank]
  psBLZ=14725836
  psName=Kreissparkasse (Alf)
  psKurzname=Kreissparkasse
[END]
[Bank]
 psBLZ=95175314
  psName=Raiffeisenbank Zeller Land (Alf)
  psKurzname=Raiffeisenbank
[END]
<END>
END_OF_FILE
```

gültig ab Version 2.73 40_



3.8.2 Verwendung der Funktion USERINPUT() zur Eingabe von Variablen - Werten

Die Funktion USERINPUT() kann außer zur Eingabe von Property-Werten auch zur Wertzuweisung von Variablen zur Laufzeit des Scriptes verwendet werden. Die Anwendung der Funktion unterscheidet sich in diesem Fall geringfügig.

Beispiel 7

Beim Import von Buchungsdaten sollen zur Laufzeit eines Scriptes das Buchungsjahr und der aktuelle Buchungsmonat für die einzulesenden Buchungsdaten angegeben werden.

Es werden zwei Variablen, BUCHUNGSJAHR und BUCHUNGSMONAT definiert, deren Werte zur Laufzeit des Scriptes eingegeben werden sollen.



Abbildung 72

Im Aktionseditor wird die folgende Aktion erstellt

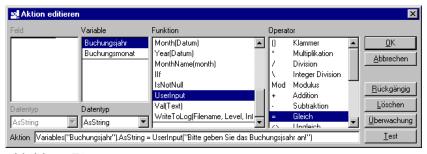


Abbildung 73

und diese dann in die Objektstruktur eingefügt. Ebenso wird für die Angabe des Buchungsmonates eine Aktion erstellt und diese dann in die Objektstruktur eingefügt. Dann werden die Variablen den entsprechenden Properties (Monat des Wirtschaftsjahres und Buchungszeitraum) zugewiesen.

Auszug aus der Objektstruktur Mandant_Buchungen nach Einfügen der Aktionen:



Abbildung 74

Zur Laufzeit des Scriptes werden jetzt die beiden Variablen - Werte abgefragt.



Abbildung 75

gültig ab Version 2.73



3.9 Erstellen einer Protokolldatei, die Funktion WRITETOLOG()

Unter Verwendung der Funktion WRITETOLOG() ist es möglich, während der Laufzeit eines Scriptes eine Protokolldatei zu erstellen und in diese Feldwerte, Variablenwerte, Kommentare etc. zu schreiben. Der Aufruf dieser Funktion erfolgt innerhalb von benutzerdefinierten Funktionen oder Prozeduren über den Befehl CALL.

Syntaxnotation: WRITETOLOG(FILENAME, [LEVEL], INFO, [CLEARATINIT])

FILENAME: Name der LOG - Datei in Hochkomma (z.B. "c:\Protokoll.txt")

LEVEL: Einrückung, Anzahl Leerzeichen

INFO: Text, der geschrieben wird, in Hochkomma CLEARATINIT: 1-Inhalt wird bei Initialisierung gelöscht

0-Inhalt bleibt erhalten (anfügen)

Beispiel 8

Das Beispiel 5 wird dahingehend abgewandelt, dass eine Prüfung der Kontonummern und Bankleitzahlen erfolgt. Es erfolgt ein Eintrag in der Protokolldatei "C:\Bankenimport.log", sofern keine Kontonummer übergeben wird oder die Länge der Bankleitzahl <> 8 ist. Im Funktionseditor¹⁸ wird die folgende Funktion erstellt:

```
function log()

if len(Mandant_Bank("Kontonummer").AsString) = 0 then

call WriteToLog("C:\Bankenimport.log",0,Mandant_Bank("Mandantennummer").AsString & ", Die
Bankverbindung ist unvollständig",1)
end if

if len(Mandant_Bank("BLZ").AsString) <> 8 then

call WriteToLog("C:\Bankenimport.log",0,Mandant_Bank("Mandantennummer").AsString & ", " &
Mandant_Bank("Bankname").AsString & ", Die BLZ ist fehlerhaft",0)
end if

end function
```

Der Aufruf dieser Funktion erfolgt in der Objektstruktur unter dem Objekt BANKKONTO als eingefügte Aktion.



Abbildung 76

Als Ergebnis liefert diese Funktion in der Datei "C:\Bankenimport.log" Protokolleinträge, sofern fehlerhafte Bankverbindungen in der Textdatei übergeben werden.

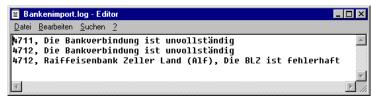


Abbildung 77

gültig ab Version 2.73 42_

¹⁸ siehe Abschnitt 4.3 Der Funktionseditor



4 Anhang

4.1 Der Textimportassistent

Über den Textimportassistenten erfolgt das Einlesen von Quelldateien (Textdateien, Excel-Dateien), die in das tse:nit Austauschformat konvertiert werden sollen. Nachdem über den Menüpunkt DATEI | NEUE TEXTDATEI eine Textdatei im Format *.asc, *.chr, *.csv, *.dat, *.txt ausgewählt wurde, wird das nebenstehende Formular geöffnet.

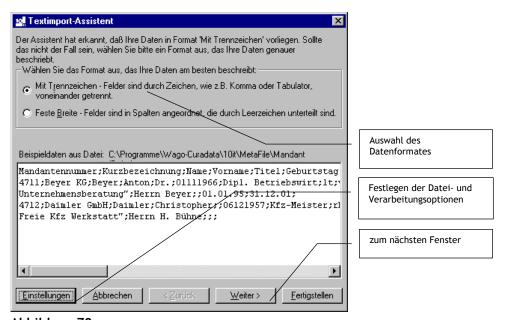


Abbildung 78

In diesem Formular wird das Eingabeformat der Quelldatei festgelegt.

MIT TRENNZEICHEN Verwendung bei CSV-Dateien¹⁹, d.h. die einzelnen Feldwerte in den Datensätzen

sind durch Trennzeichen getrennt

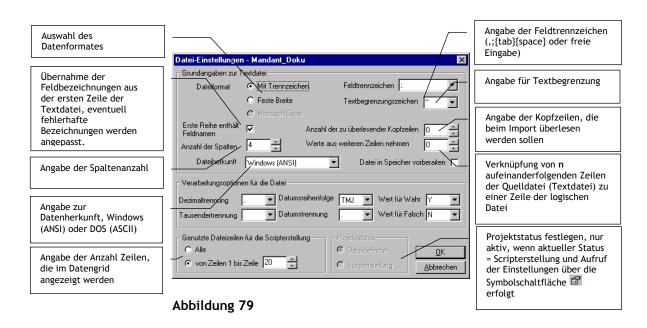
FESTE Breite Anordnung der Feldwerte in den Datensätzen mit fester Spaltenbreite

Über die Schaltfläche können weitere Angaben zum Format der Quelldaten und zu den Verarbeitungsoptionen vorgenommen werden.

gültig ab Version 2.73 43

¹⁹ CSV - Character Separated Value





Im Bereich VERARBEITUNGSOPTIONEN FÜR DIE DATEI können die folgenden Optionen eingestellt werden:

DEZIMALTRENNUNG: Dezimaltrennung erfolgt durch:

"." (Punkt) z.B. 1.23 "," (Komma) z.B. 1,23

TAUSENDERTRENNUNG: Trennzeichen zwischen Tausendern ist:

" (Leerzeichen) z.B. 1 234 567,89 ." (Punkt) z.B. 1.234.567,89 ," (Komma) z.B. 1,234,567.89

DATUMSREIHENFOLGE: Reihenfolge der Datumsangabe (in Verbindung mit Datumstrennzeichen):

(Tag Monat Jahr) 30052001 TMJ z.B. 30-2001-05 TJM(Tag Jahr Monat) z.B. (Monat Tag Jahr) MTJ z.B. 05.30.2001 MJT (Monat Jahr Tag) z.B. 05/2001/30 **JTM** (Jahr Tag Monat) z.B. 2001:30:05 JMT (Jahr Monat Tag) z.B. 20010530

DATUMSTRENNUNG: Trennzeichen in Datumsangaben:

"." "<u>-</u>" (Punkt) z.B. 29.10.2001 (Bindestrich) z.B. 15-9-01 (Schrägstrich) 4/6/99 z.B. ":" (Doppelpunkt) z.B. 9:12:87 (kein Trennzeichen) z.B. 150698 {no}

WERT FÜR WAHR: logischer Wert für "Wahr" ist repräsentiert als:

"True"
"Ja"
"Yes"
"On"
"T"

"J" "1"

gültig ab Version 2.73 44_



WERT FÜR FALSCH: logischer Wert für "Falsch" ist repräsentiert als:

"False"
"Nein"
"No"
"Off"
"F"
"N"

"0" (Ziffer 0)

Erläuterung zur Option PROJEKTSTATUS:

Wenn der Dialog EINSTELLUNGEN aus dem Textimport-Assistenten aufgerufen wird, ist die Option Dateidefinition aktiviert und es können alle vorhandenen Einstellungen geändert werden. Im anderen Fall, dem Aufruf aus dem Menü oder über die Symbolschaltfläche [67], ist die Option Scripterstellung eingestellt und man kann nur Einstellungen vornehmen, deren Änderung die Anzahl der Felder (Spalten) und deren Namen nicht beeinflusst. Alle Einstellungen können vom Anwender explizit geändert werden, wenn man in die Option Dateidefinition umschaltet. In diesem Fall wird der Anwender vor der Umschaltung gewarnt, da jetzt die Feldnamen modifiziert werden können und es so zu einer "Entwertung" der früher definierten Verbindungen kommen kann und damit das bereits erstellte Script unbrauchbar wird.

4.1.1 Einlesen von Daten im Format MIT TRENNZEICHEN



Nachdem über die Schaltfläche Intellungen alle erforderlichen Angaben gemacht wurden, gelangt man über die Schaltfläche weber in das nächste Fenster (siehe Abbildung 80), in dem noch einmal Einstellungen zu dem gewählten Format vorgenommen werden können.

Abbildung 80



Wiederum mit der Schaltfläche gelangt man in das folgende Fenster (siehe Abbildung 81), in dem der logische Name der Textdatei und die logischen Namen der Datenfelder und deren Datentypen festgelegt werden. In der unteren Hälfte dieses Fensters werden zur Kontrolle der Einstellungen Beispielwerte aus der Quelldatei angezeigt.

Abbildung 81

Über die Schaltfläche Fertigstellen werden die Datensätze aus der Textdatei in das Datengrid eingelesen.

gültig ab Version 2.73 45_



4.1.2 Einlesen von Daten im Format FESTE BREITE



Nachdem über die Schaltfläche Intelligen alle erforderlichen Angaben gemacht wurden, gelangt man über die Schaltfläche in das nächste Fenster (siehe Abbildung 82), in dem die Festlegung der Spaltenbreite der Datenfelder vorgenommen wird. Zum Erstellen einer Feldtrennlinie wird die gewünschte Position mit der linken Maustaste angeklickt. Zum Löschen einer Trennlinie wird diese erneut mit der linken Maustaste angeklickt. Zum Verschieben einer Trennlinie wird diese mit gedrückter linker Maustaste an ihre neue Position gezogen.

Abbildung 82



Wiederum mit der Schaltfläche gelangt man in das folgende Fenster (siehe Abbildung 83), in dem der logische Name der Textdatei und die logischen Namen der Datenfelder und deren Datentypen festgelegt werden. In der unteren Hälfte dieses Fensters werden zur Kontrolle der Einstellungen Beispielwerte aus der Quelldatei angezeigt.

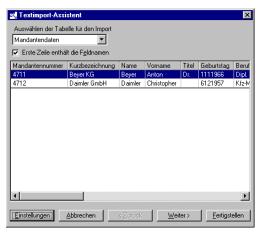
Abbildung 83

Über die Schaltfläche Fertigstellen werden die Datensätze aus der Textdatei in das Datengrid eingelesen.

gültig ab Version 2.73 46_



4.1.3 Einlesen von Daten im EXCEL - Format



Bei der Auswahl von Excel (*.xls) Dateien über den Menüpunkt DATEI | NEUE TEXTDATEI öffnet sich das nebenstehende Fenster. Die Auswahl des Excel-Tabellenblattes, aus dem die Daten eingelesen werden sollen, erfolgt über ein Listenfeld.

Abbildung 84

Über die Schaltfläche Enselween sind in der Regel keine Angaben zu machen, da die Datei- und Verarbeitungsoptionen schon durch die verwendeten Formate im Excel-Sheet vorgegeben und durch den Importassistenten erkannt werden. Bis auf die u.a. Optionen sind beim Import von Excel-Daten keine Angaben möglich.

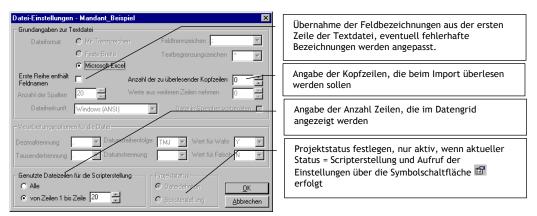


Abbildung 85



Abbildung 86

Nachdem über die Schaltfläche Entellingen alle erforderlichen Angaben gemacht wurden, gelangt man über die Schaltfläche in das nächste Fenster (siehe Abbildung 86), in dem der logische Name des Excel-Datenblattes und die logischen Namen der Datenfelder und deren Datentypen festgelegt werden. In der unteren Hälfte dieses Fensters werden zur Kontrolle der Einstellungen Beispielwerte aus der Quelldatei angezeigt. Über die Schaltfläche Ferigstellen wird die unter Einstellungen festgelegte Anzahl Zeilen in das Datengrid eingelesen.

gültig ab Version 2.73 47_



4.2 Der Formel- / Aktionseditor

Der Formel - Editor ermöglicht es,das Erstellen von Ausdrücken (Berechnungsformeln) zu vereinfachen. Der Aufruf erfolgt mittels der Symbolschaltfläche In neben dem Formeleingabefeld bzw. dem Aktionseingabefeld, durch Doppelklick auf eine Zeile im Objektexplorer oder über das Kontextmenü im Funktionseditor (Menüpunkt FORMEL EINFÜGEN oder AKTION EINFÜGEN).

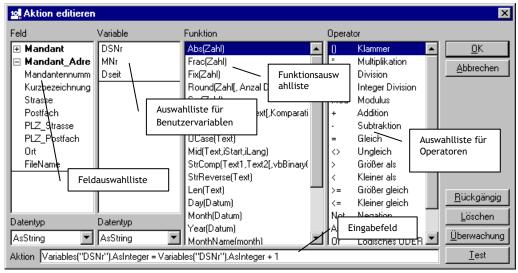


Abbildung 87

Das Editorfenster enthält der Reihe nach: Liste der Felder aus den aktuellen Textdatein, Liste der Benutzervariablen, Liste der wichtigsten Funktionen²⁰ und Liste der Operatoren.

Mittels Doppelklick auf einen Eintrag in den Auswahllisten, durch Betätigen der Leertaste oder via Drag & Drop bei gleichzeitigem Drücken der [Strg] Taste werden die jeweiligen Einträge in das Eingabefeld (je nach Aufruf des Formeleditors mit der Bezeichnung FUNKTION oder AKTION) zur weiteren Ausdrucksbearbeitung übernommen.

Der aktuelle Wert von Feldern und Benutzervariablen wird als Tooltip angezeigt, sobald der Mauszeiger auf den jeweiligen Eintrag positioniert wird.

Der Inhalt des Listenfeld DATENTYP zeigt immer den aktuellen Datentyp des markierten Feldes / der markierten Variable. Wird ein anderes Feld oder eine andere Variable ausgewählt, ändert sich entsprechend auch der Inhalt dieses Listenfeldes.

Die Schaltfläche Dietet dem Anwender die Möglichkeit, aktuelle Werte von mehreren Ausdrücken zu überwachen. Sie dient ebenfalls als Schalter für das Ein- und Ausblenden der Tabelle zur Ausdrucksüberwachung.

Mit der Schaltfläche Rückgängig werden die letzten ausgeführten Eingaben gelöscht.

Über die Schaltfläche wird der Formeleditor geschlossen, ohne die vorgenommenen Änderungen zu speichern.

gültig ab Version 2.73 48_

²⁰ Außer den enthaltenen Funktionen können weitere Funktionen des VBScript Sprachumfanges genutzt werden.



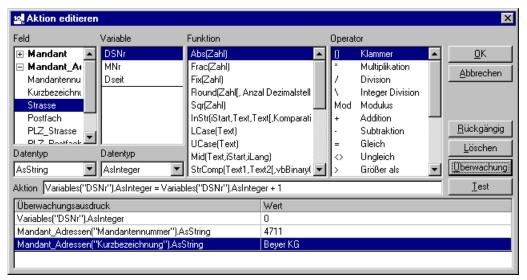


Abbildung 88

Überwachungsausdrücke werden über das Kontextmenü im Grid oder per Drag & Drop bei gleichzeitigem Drücken der [Strg] Taste von Listeneinträgen hinzugefügt.

Kontextmenü:



- Möglichkeit der Modifizierung des aktuellen Ausdrucks
- Dialog zur Eingabe eines neuen Ausdrucks
- Ausdruck aus der Eingabezeile übernehmen
- Hinzufügen der aktuell gewählten Variablen
- Hinzufügen des aktuell gewählten Feldes
- Ausgewählte Überwachung Löschen
- Alle Überwachungen aus der Liste entfernen
- Aktualisieren der Überwachungen in der Liste

Abbildung 89

Durch Betätigen der Schaltfläche wird versucht, die in der Eingabezeile hinterlegte Formel zu berechnen, was zur Kontrolle des erstellten Berechnungsausdrucks dient. Es erfolgt eine Syntaxprüfung und Berechnung der eingegebenen Formel.

Nach Betätigung der Schaltfläche wird der erstellte Ausdruck in das Formeleingabefeld, das Aktionseingabefeld oder den Funktionseditor übernommen.

gültig ab Version 2.73 49_



4.3 Der Funktionseditor

Zur Vereinfachung von Ausdrücken oder zur Berechnung komplizierter Ausdrücke ist es im TXTImport möglich, benutzerdefinierte Funktionen und Prozeduren zu erstellen, und diese mit dem Script, genau wie die benutzerdefinierten Variablen, zu speichern.

Der Aufruf des Funktionseditors erfolgt entweder über den Menüpunkt BEARBEITEN | FUNKTION oder über die Symbolschaltfläche 🗲 in der Programmoberfläche. Es öffnet sich das folgende Formular.

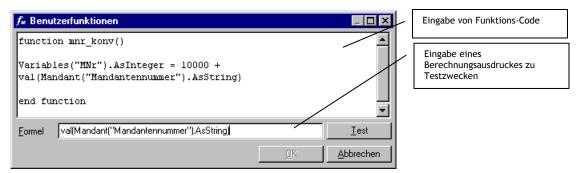
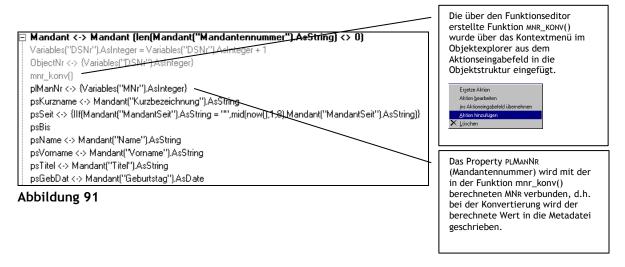


Abbildung 90

Die Abbildung zeigt die Funktion MNR_KONV(), mit deren Hilfe man zum Beispiel eine Konvertierung der vierstelligen Mandantennummern in fünfstellige Mandantennummern erreichen könnte. Die Berechnung wird in dieser Funktion der Variablen Variables ("MNR"). ASINTEGER zugewiesen, die dann im Objekt-explorer mit dem Property PLMANNR (Mandantennummer) verknüpft wird. Die benutzerdefinierte Funktion wird an geeigneter Stelle in die Objektstruktur als Aktion eingefügt (siehe Abbildung 91).

Im Eingabefeld FORMEL im unteren Teil des Formulars kann ein Testausdruck eingegeben werden, der dann über die Schaltfläche let berechnet wird. Falls die Funktion fehlerhaft ist, wird eine Fehlermeldung angezeigt und der Cursor wird in die Code Zeile gesetzt, in der der Fehler erkannt wurde.

Auszug aus der Objektstruktur für das Objekt MANDANT (Mandant):



gültig ab Version 2.73 50_



Kontextmenü im Funktionseditor:

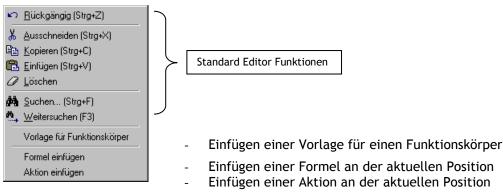


Abbildung 92

Das Kontextmenü enthält neben den Standard Editor Bearbeitungsbefehlen folgende Menüpunkte:

VORLAGE FÜR FUNKTIONSKÖRPER: An der aktuellen Cursorposition wird ein Muster für einen Funktionskörper eingefügt:

Function Function_Name()
'Name durch Funktionsnamen ersetzen!

Function_Name =
End Function

Danach ist die eingefügte Funktion umzubenennen (Function_Name() durch wirklichen Funktionsnamen ersetzen) und der Funktionsinhalt in Form von Programmanweisungen (Befehlen) einzugeben.

FORMEL EINFÜGEN: Der Formeleditor wird aufgerufen. Bilden und testen eines

Berechnungsausdrucks im Formeleditor. Der so gebildete

Berechnungsausdruck wird an der aktuellen Cursorposition in den

Formeleditor eingefügt.

AKTION EINFÜGEN: Der Aktionseditor wird aufgerufen. Die gebildete Aktion wird an der

aktuellen Cursorposition in den Funktionseditor eingefügt.

gültig ab Version 2.73 51_



4.4 Debug-Modus

Über die Funktionstaste **[F8]** wird der Debug-Modus eingeschaltet. In diesem speziellen Programm-Modus besteht die Möglichkeit, Scriptdurchläufe zu testen und anhand von Überwachungsausdrücken die Wertzuweisung zu Properties oder Variablen zu überprüfen. Weiterhin besteht im Debug-Modus die Möglichkeit, in der Objektstruktur so genannte Haltepunkte zu setzen, bei denen die Abarbeitung eines Scriptes unterbrochen wird.

Funktionstastenbelegung im Debug-Modus:

- [F8] Ausführen Einzelschritt (Symbolschaltfläche 🕦)
- [F5] Ausführen bis Haltepunkt bzw. Scriptende (Symbolschaltfläche)
- [ESC] Scriptausführung beenden, Rückkehr in den Definitions-Modus (Symbolschaltfläche 🔳

Bildschirmansicht im Debug-Modus:

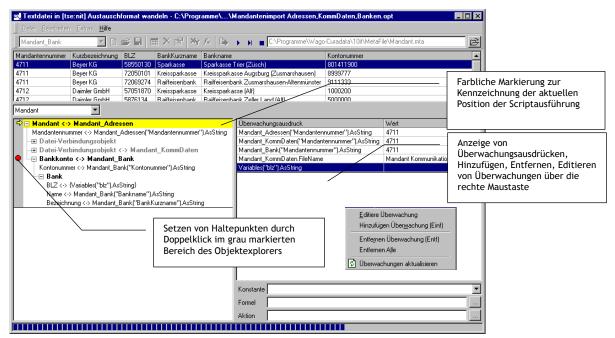


Abbildung 93

gültig ab Version 2.73 52_