Beschreibung TXT-Import

Inhaltsverzeichnis

Beschreibung TXT-Import

1 Einleitung

1.1 Allgemeine Programmbeschreibung

Bei dem Programm TXTImport, handelt es sich um eine in SBS Rewe neo® implementierte, frei programmierbare Schnittstelle, die es ermöglicht, Daten, die aus Fremdsystemen im ASCII/ANSI- bzw. Excel-Format bereitgestellt werden, in das SBS Rewe neo® 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 SBS Rewe neo® Austausch-format 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 Firmendaten liegen in zwei Dateien, **Firma.txt** und **Firma Adresse.txt** mit folgender Datenstruktur vor:

Firma.txt	Firma Adresse.txt
Mandantennummer	Mandantennummer
Kurzbezeichnung	Kurzbezeichnung
Name	Straße
Vorname	Postfach
Title	PLZ Straße
Geburtstag	PLZ Postfach
Beruf	Ort
Religion	
Familienstand	
UStID	
Rechtsform	
Branche	

Zwischen diesen beiden Dateien wird über das Feld Mandantennummer (Firmennummer) 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.

Während der Konvertierung würden diese beiden Textdateien zu einer Datei Firma.mta zusammengeführt, die dann in das SBS Rewe neo® Mandatsverzeichnis importiert werden kann.

1.2 Installation

Eine gesonderte Installation des Programms durch den Anwender ist nicht erforderlich. Die Installation erfolgt im Rahmen der SBS Rewe neo® Installation bzw. des SBS Rewe neo® Updates.

Zum Programmumfang gehören die Dateien:

- TXTImport.exe
- srvTXTImport.dll

im SBS Rewe neo® Installationsverzeichnis <C:\Program Files (x86)\SBS-Software\SBS-Rewe>.

1.3 Programmaufruf

Der Programmaufruf kann über die SBS Rewe neo® Programmoberfläche oder extern über das Windows Startmenü erfolgen.

1.3.1 Start über die SBS Rewe neo® Programmoberfläche (interner Programmaufruf)

- SBS Rewe neo® starten
- Öffnen des Dokumentes (z.B. Kundenstamm)
- Starten des TXTImportes über den Punkt Daten | Import.
- Auswahl des Importsystems Text-Datei-Import

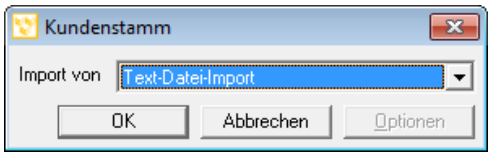


Abb. 1: Txt-Datei-Import

1.4 Unterstützte SBS Rewe neo® Datenbereiche

Für die folgenden SBS Rewe neo® Datenbereiche kann eine Konvertierung von ASCII/ANSI- bzw. Excel-Dateien in das SBS Rewe neo® Austauschformat vorgenommen werden:

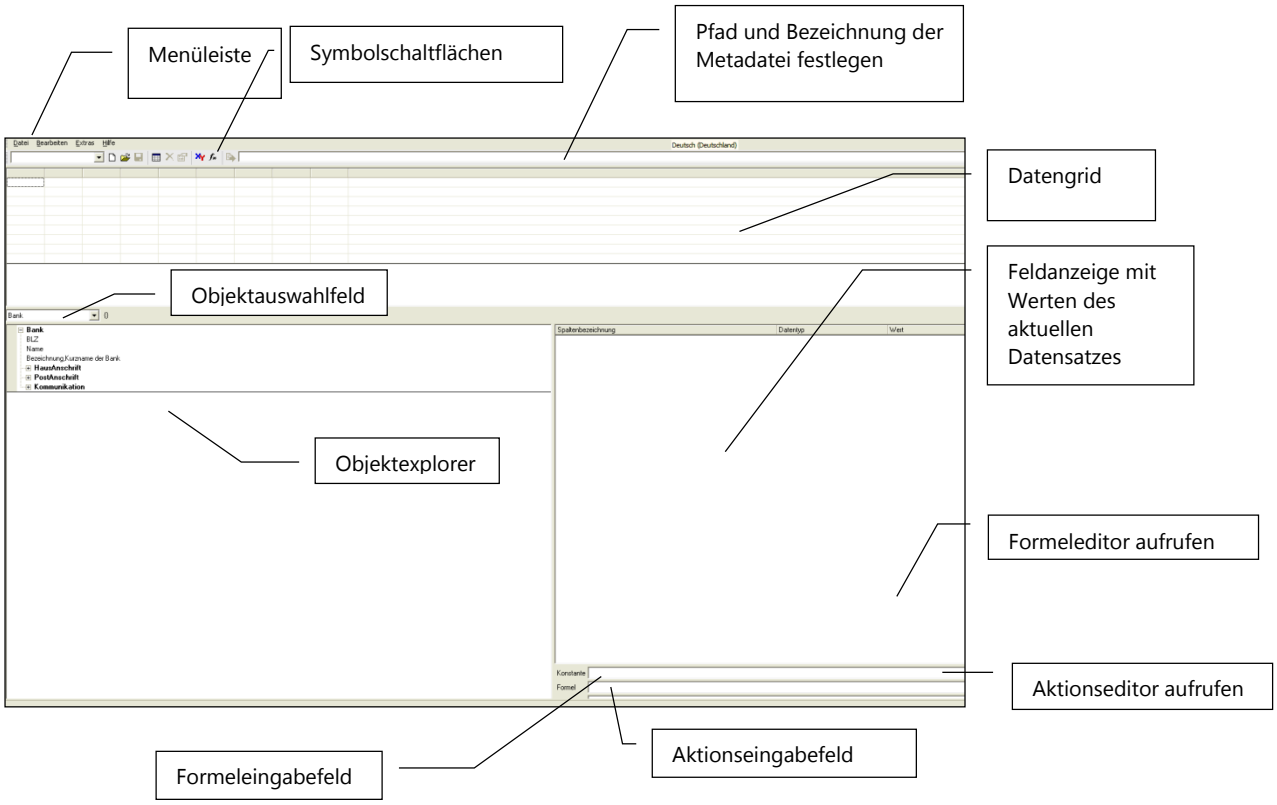
Datenbereich:	SBS Rewe neo® Dokumentpfad:
Kanzleiakte:	
Mandanten (Firmen)	Stammdaten\Mandatsverzeichnis
Banken	Stammdaten\Banken
Finanzämter	Stammdaten\Finanzämter
Gemeinden	Stammdaten\Gemeinden
Mitarbeiter	Stammdaten\Mitarbeiterliste
Mandanten- und Kanzleiakte:	
Kunden	Stammdaten\Grundlagen Rechnungswesen\Kundenstamm
Lieferanten	Stammdaten\Grundlagen Rechnungswesen\Lieferantenstamm
Stammkonten	jahresabhängige Stammdaten\Rechnungswesen\Stammdaten\Kontenstamm
Einzelbuchungen	Dokumente\Rechnungswesen\Finanzbuchhaltung\Buchungsliste
Verkehrszahlen	Dokumente\Rechnungswesen\Finanzbuchhaltung\Summen- und Saldenliste
Oder	Dokumente\Rechnungswesen\Jahresabschluss\Hauptabschlussübersicht
Offene Posten	Dokumente\Rechnungswesen\OP Buchhaltung\Offene Posten
Oder	Dokumente\Rechnungswesen\Finanzbuchhaltung\Summen- und Saldenliste
Oder	Dokumente\Rechnungswesen\Jahresabschluss\Hauptabschlussübersicht
Wirtschaftsgüter	Dokumente\Rechnungswesen\Anlagenbuchhaltung\ Abschreibungsverzeichnis

Die Schnittstellenbeschreibungen zu o.a. SBS Rewe neo® Importformaten (Metafile, SBS Rewe neo® Austauschformat) sind zum Teil auf der SBS Rewe neo® Auslieferungs-DVD abgelegt.

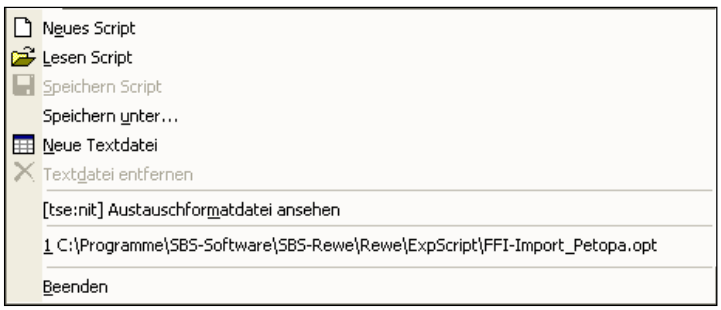
2 Programmbeschreibung

2.1 Grundlegende Bildschirmgestaltung

2.1.1 Programmoberfläche

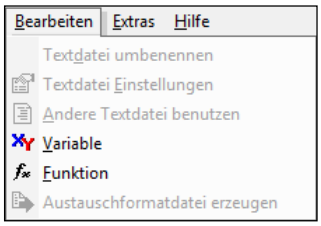


2.1.2 Menüleiste



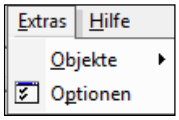
- Erstellen eines neuen Scripts
- Einlesen eines vorhandenen Scripts
- Speichern des bearbeiteten Scripts
- Speichern unter einem anderen Namen
- Neue Textdatei einlesen
- Definition einer Textdatei entfernen
- erstelltes Metafile anschauen
- Anzeige der bearbeiteten Scripte
- Programm beenden

Abb. 2: Datei



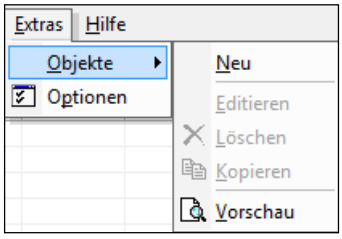
- Ändern des logischen Namens einer Textdatei
- Anzeige / Ändern der Textdatei Einstellungen
- Auswahl der zu verarbeitenden Textdatei
- Variablen definieren
- Funktionen / Prozeduren bearbeiten
- SBS Rewe neo® Austauschformat (Metafile) erzeugen

Abb. 3: Bearbeiten



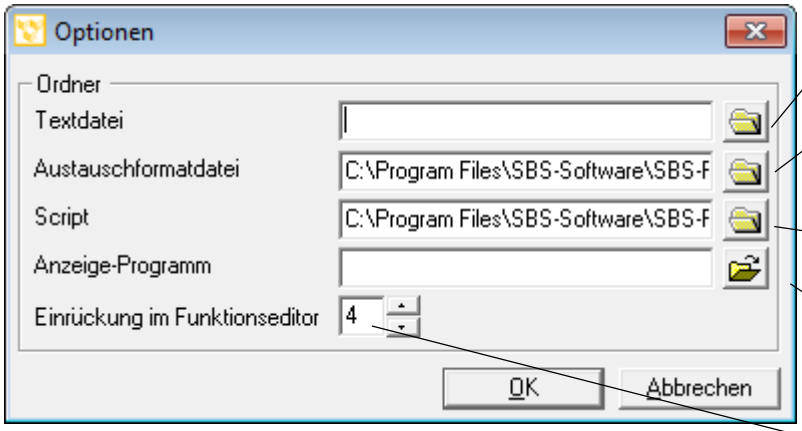
- Anpassung der Objektstruktur
- Programmoptionen

Abb. 4: Extras



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

Abb. 5: Extras -> Objekte



Auswahl des Ordners für die einzulesende Quelldatei

Auswahl des Ordners für die zu generierende SBS Rewe neo® Austauschformatdatei (Metafile)

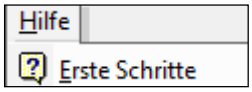
Auswahl des Ordners zur Speicherung des erstellten Scripts

Auswahl des Anzeigeprogramms für die zu generierende Rewe Austauschformatdatei

Anzahl Leerzeichen für Einrückungen im Funktionseditor bei Verwendung der Tab-Taste

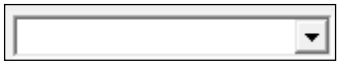
Abb. 6: Optionen Txt-Import

Hilfe



- Anzeige der Hilfedatei

2.1.3 Symbolschaltflächen



- Auswahl der Textdatei (wenn mit mehr als einer Textdatei gearbeitet wird)

- Erstellen einer neuen Script-Datei

- Öffnen eines bestehenden Scriptes

- Speichern des Scriptes

- Öffnen der zu verarbeitenden Textdatei

- Entfernen einer Textdatei

- Eigenschaften der Textdateidefinition festlegen

- Variablen definieren

- Funktionseditor

- Metafile erzeugen

2.1.4 Datengrid

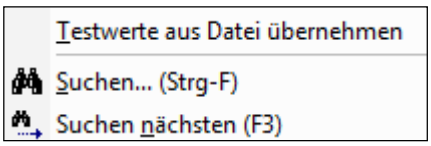
Im Datengrid wird der Inhalt der Textdatei gemäß der getroffenen Felddefinitionen während des Textdateiimportes angezeigt.

BuchungssatzID	MandantenNummer	Beginn_Wirtschaftsjahr	Ende_Wirtschaftsjahr	Buchungsnummer	Whrung	BuchungskreisNummer
10070	999	01.01.2010	31.12.2010	391	EUR	0
10071	999	01.01.2010	31.12.2010	391	EUR	0

Abb. 7: Datengrid

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.

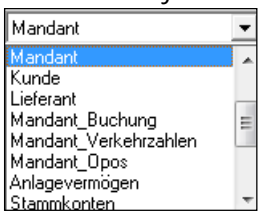
Kontextmenü im Datengrid:



- Testwerte in die Feldanzeige übernehmen
- Nach Wert suchen
- Nächsten Wert suchen

2.1.5 Objektauswahlfeld

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.

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 SBS Rewe neo® Austauschformat erfolgen. Die Umschaltung erfolgt über das Kontextmenü.

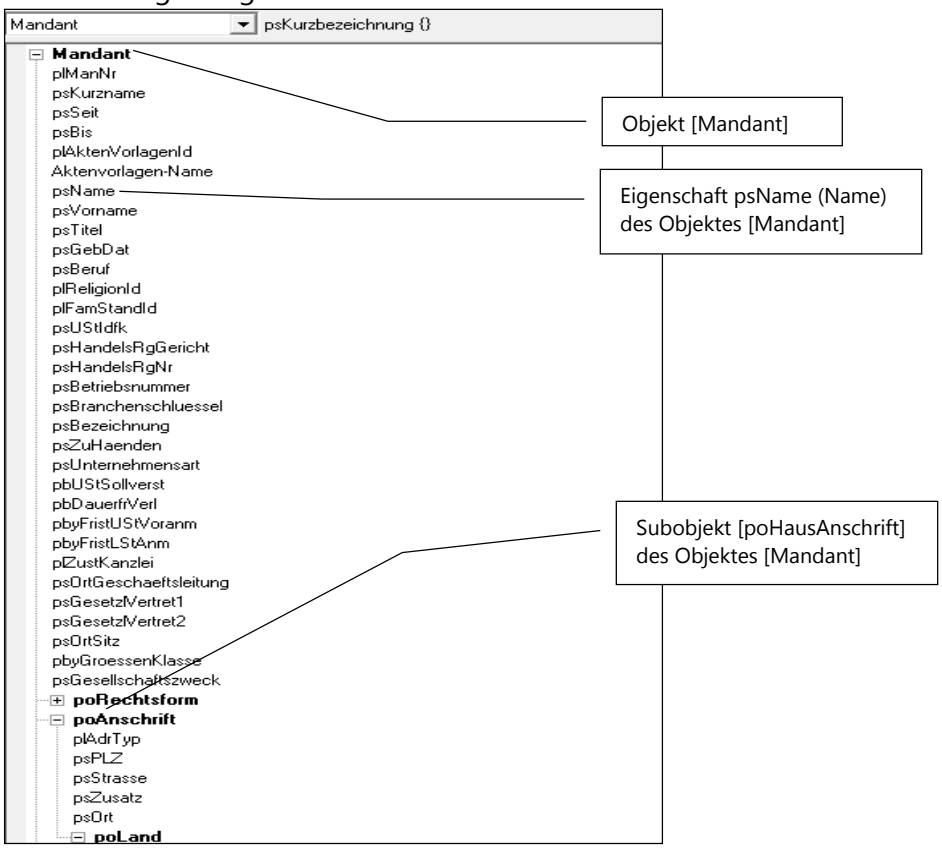
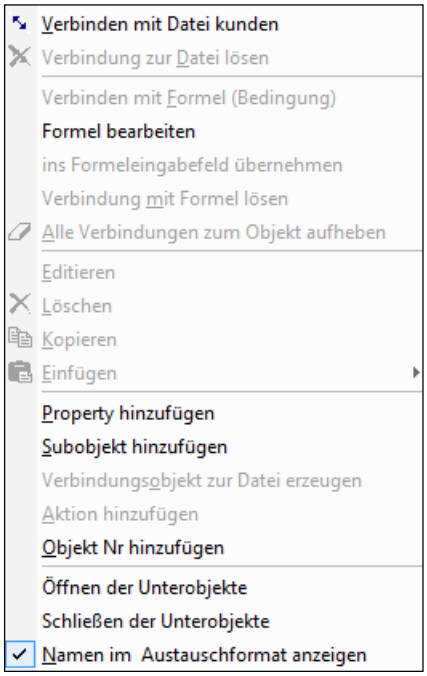


Abb. 8: Objektexplorer

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):

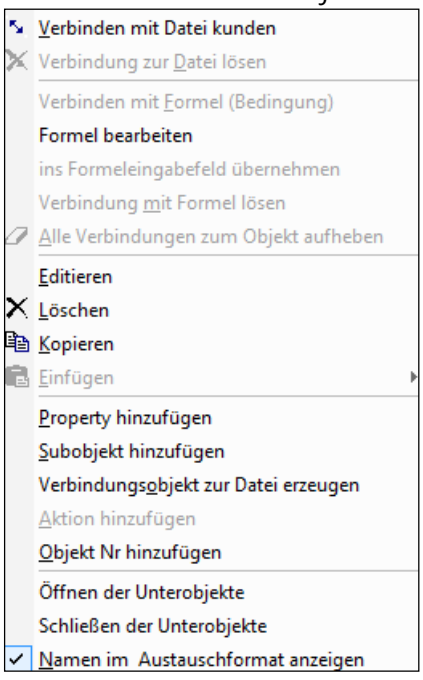


- logische Verbindung zur Datei erzeugen
- logische Verbindung zur Datei lösen
- Bedingung aus dem Formeleingabefeld übernehmen
- Formel (Bedingung) bearbeiten
- Formel (Bedingung) ins Formeleingabefeld übernehmen
- Formel (Bedingung) löschen
- alle getroffenen Zuordnungen werden gelöscht

- Hinzufügen eines Propertys zum Objekt
- Hinzufügen eines Unterobjektes zum Objekt

- Hinzufügen des Objektes **ObjectNr**
- Öffnen aller Unterobjekte
- Schließen aller Unterobjekte
- Bezeichnung werden im SBS Rewe neo® Austauschformat gezeigt

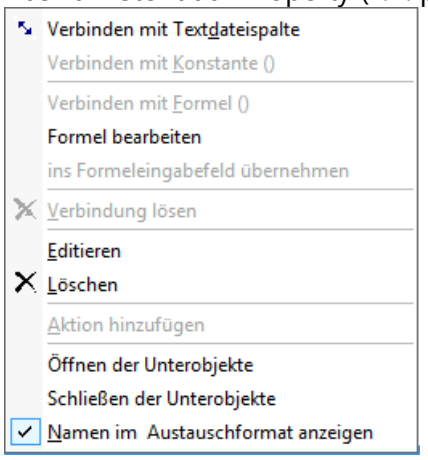
Auswahl steht auf Subobjektebene (z.B. poHausAnschrift):



- logische Verbindung zu einer weiteren Datei erzeugen
- logische Verbindung zur Datei lösen
- Bedingung aus dem Formeleingabefeld übernehmen
- Formel (Bedingung) bearbeiten
- Formel (Bedingung) ins Formeleingabefeld übernehmen
- Formel (Bedingung) löschen

- Subobjekt bearbeiten
- 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
- Schließen aller Unterobjekte
- Bezeichnung werden im SBS Rewe neo® Austauschformat gezeigt

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
- Schließen aller Unterobjekte
- Bezeichnung werden im SBS Rewe neo® Austauschformat gezeigt

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.

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.

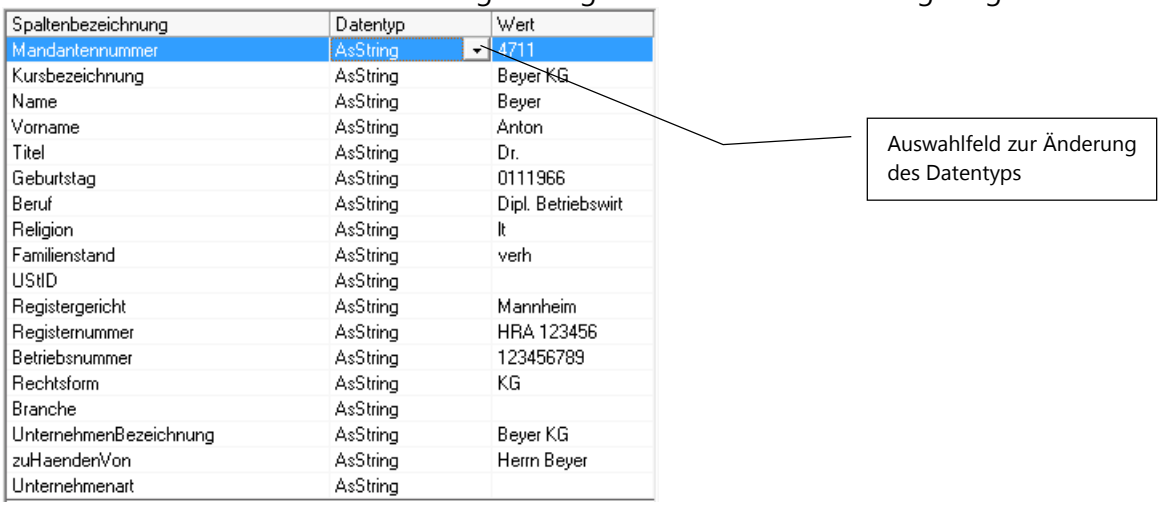


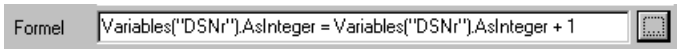
Abb. 9: Feldanzeige

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:

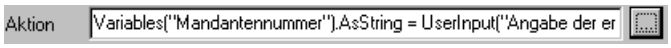
Datentyp	Beispiel
Boolean	True, False
Currency	123.456.789,12; -123.456.789,12
Date	21.05.2002 (Werte zw. 01.01.100 u. 31.12.9999)
Float	123456789,12; -123456789,12
Integer	123456789; -123456789

2.1.8 Formeleingabefeld



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



Das Aktionseingabefeld dient zur Vorbereitung von sogenannten 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:

- Starten des TXTImportes
- Auswahl des Datenbereiches über das Objektauswahlfeld.
- Auswahl der Quelldatei (ASCII, ANSI, Excel) über den Textimportassistenten (Menü **Datei** ->**Neue Textdatei**)

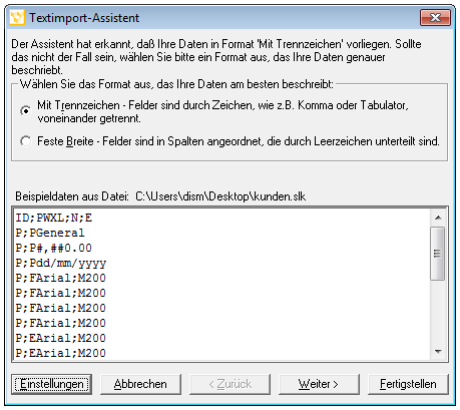


Abb. 10: Assistent Txt-Import

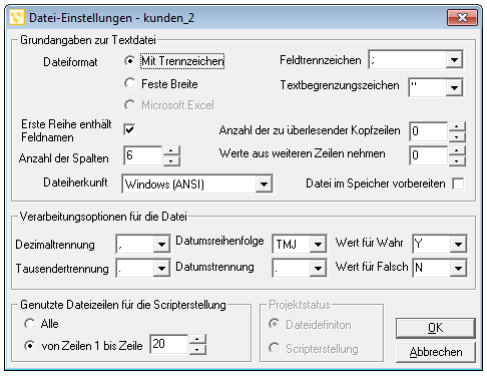


Abb. 11: Festlegen der Datei- und Verarbeitungseinstellungen

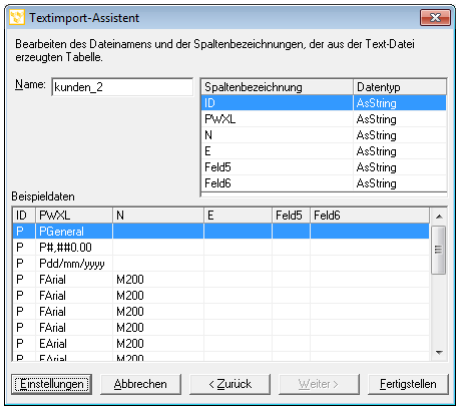


Abb. 12: Festlegen der logischen Datei- und Feldnamen und der Felddatentypen.

Einlesen der Quelldatei über 

.
Über das Kontextmenü im Objektexplorer verbinden des Objektes (z.B. Mandant) mit dem logischen Namen der eingelesenen Datei

Zuordnung der Datenfelder zu den Properties im Objektexplorer per Drag & Drop über die Feldanzeige und den Objektexplorer vornehmen
Erzeugung der SBS Rewe neo® Austauschformatdatei (z.B. Mandant.mta) mittels der Schaltfläche 

In SBS Rewe neo® das entsprechende Dokument öffnen (z.B. Mandatsverzeichnis in der Kanzleiakte) und die erzeugte Metadatei über den Menüpunkt **Datei** -> **Importieren**. als SBS Rewe neo® Austauschformat einlesen.

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:

SBS Rewe neo® starten und Öffnen des entsprechenden Dokumentes, in das Daten importiert werden sollen (z.B. Mandatsverzeichnis)
Starten des TXTImportes über den Menüpunkt **Datei** -> **Importieren**... -> **Text-Datei-Import**

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

Auswahl der Quelldatei (ASCII, ANSI, Excel) über den Textimportassistenten (Menü **Datei** -> **Neue Textdatei**)

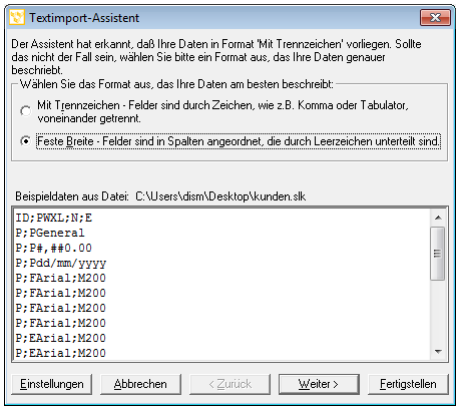


Abb. 13: Assistent Txt-Import

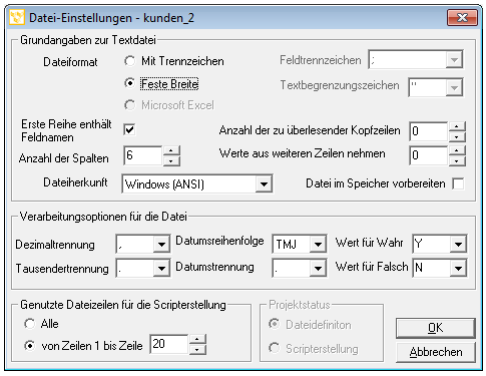


Abb. 14: Festlegen der Datei- und Verarbeitungseinstellungen

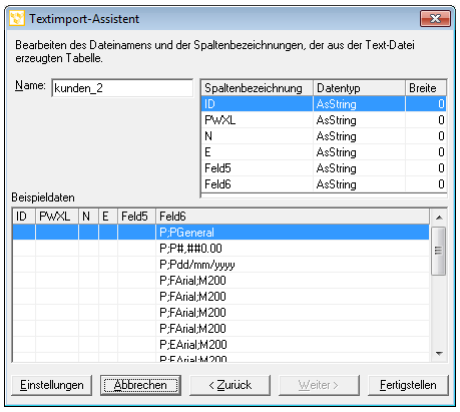


Abb. 15: Festlegen der logischen Datei- und Feldnamen und der Felddatentypen.

Einlesen der Quelldatei über Fertigstellen.
Über das Kontextmenü im Objektexplorer verbinden des Objektes (z.B. Mandant) mit dem logischen Namen der eingelesenen Datei Verbinden mit Datei Mandant
Zuordnung der Datenfelder zu den Properties im Objektexplorer per Drag & Drop über die Feldanzeige und den Objektexplorer vornehmen

- Erzeugung der SBS Rewe neo® Austauschformatdatei (z.B. Mandant.mta) mittels der Schaltfläche
- Import der Daten über die Schaltfläche Übernehmen in der Metafileanzeige

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.

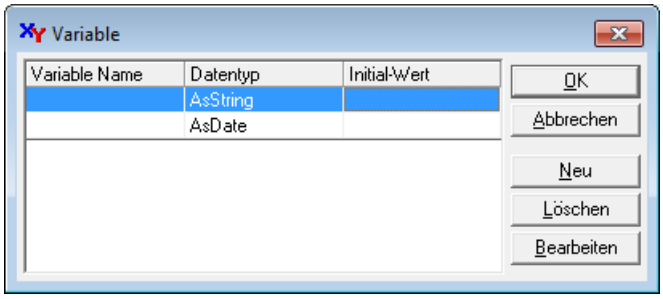


Abb. 16: Variable

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() Beispiel: now() = 22.05.10 14:35 mid(now(),1,8) = 22.05.10 [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:

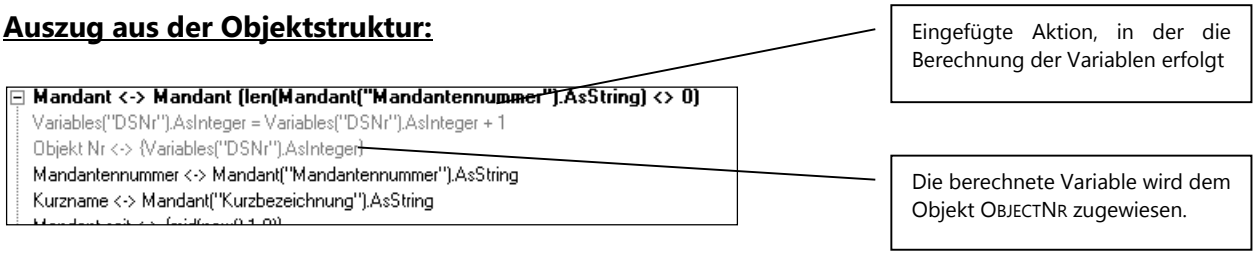
Variables("Variablenname").AsDatentyp

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

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 fortlaufend hochzuzählen.

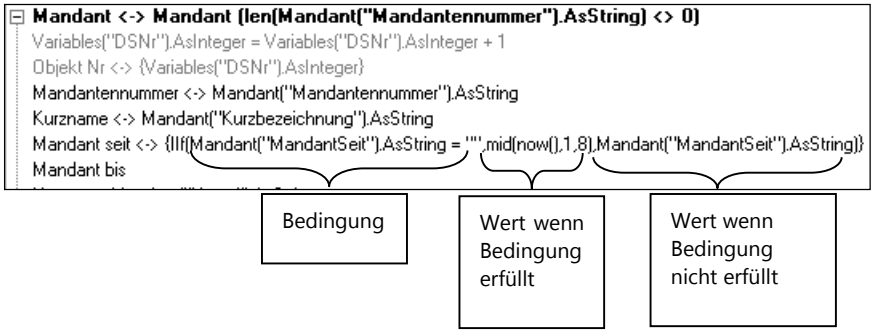
Auszug aus der Objektstruktur:



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:



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.

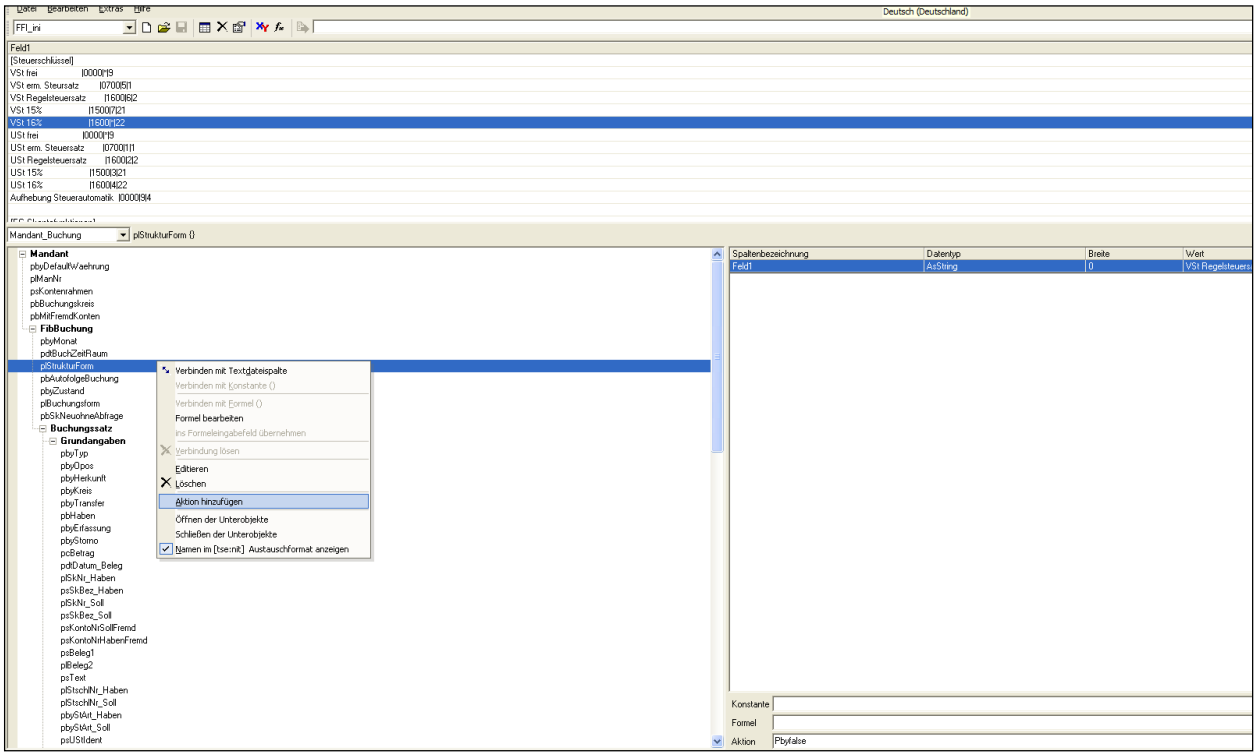
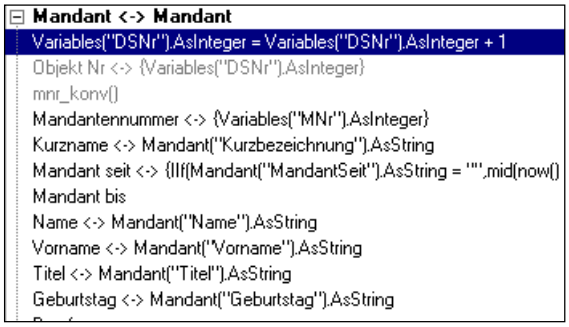


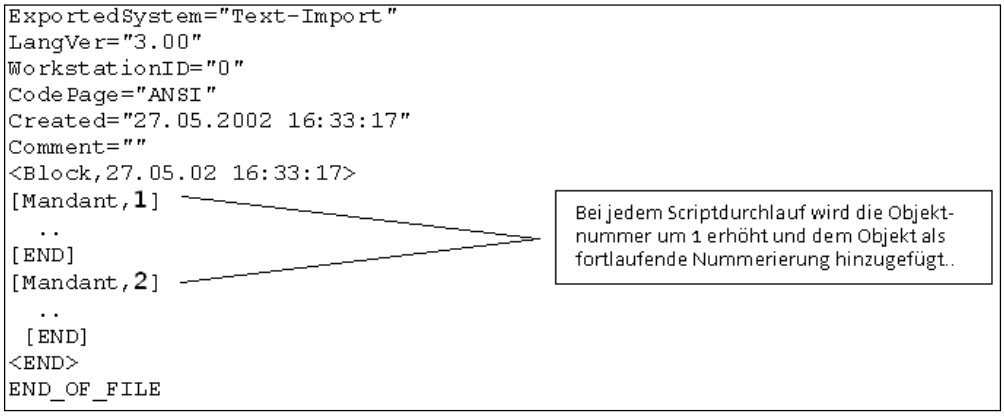
Abb. 17: Ansicht Txt-Import

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.



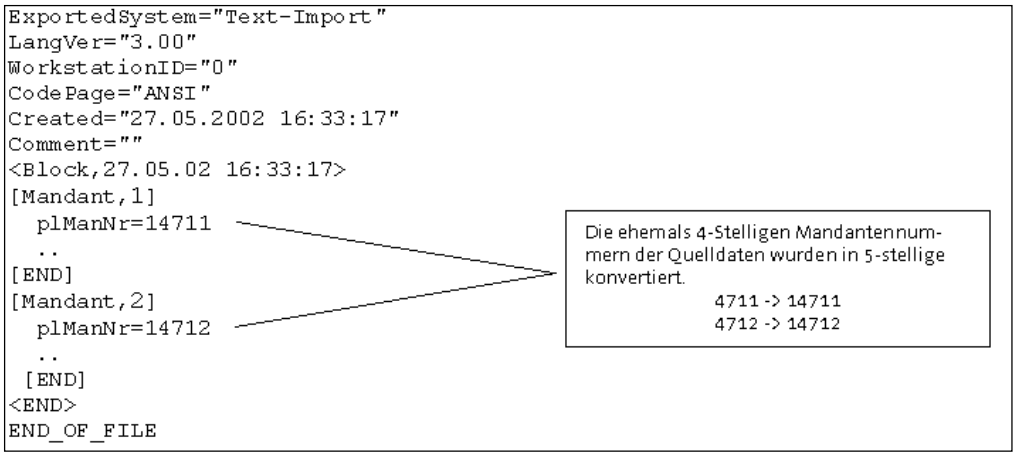
Im o.a. Beispiel wird die benutzerdefinierte Variable **DSNr** bei jedem Scriptdurchlauf um eins erhöht. Die über die Aktion
Variables("DSNr").AsInteger = Variables("DSNr").AsInteger + 1

berechnete Variable wird in der nächsten Objekt-Zeile dem Objekt ObjectNr zugewiesen. Das Ergebnis der eingefügten Aktion sieht dann in der erzeugten Metadatei folgendermaßen aus:



Als Aktionen können außer Berechnungsformeln auch benutzerdefinierte Funktionen 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. 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.

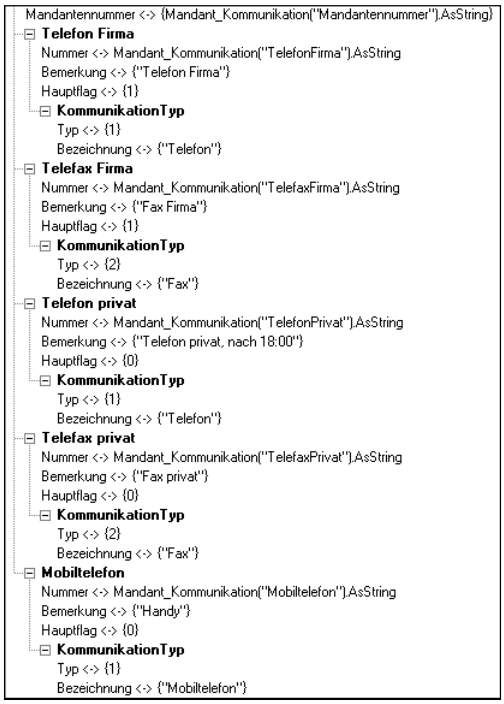
Das Ergebnis der als Aktion eingefügten Funktion sieht dann folgendermaßen aus:



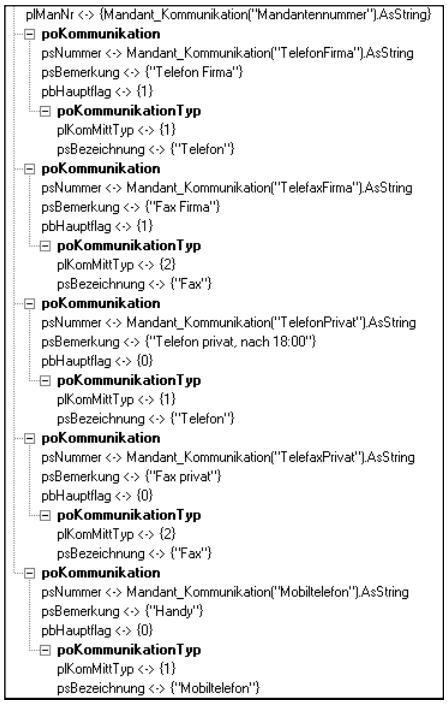
3.5 Modifizierung der Objektstruktur

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.

Normalansicht:



Anzeige im SBS Rewe neo® Austauschformat:



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

Achtung: Modifizierte Objekte müssen unbedingt der SBS Rewe neo® Schnittstellenbeschreibung entsprechen, da Formatverletzungen beim SBS Rewe neo® Import zu Fehlern führen.

3.5.1 Erstellte SBS Rewe neo® Austauschformatdatei:

```
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]
   plKomMittTyp=1
   psBezeichnung=Telefon
  [END]
 [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]
      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]
      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
```

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 SBS Rewe neo® 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;
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;0171-987654321;Mobiltelefon;
```

Kommunikationsdaten
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.
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. Die Kommunikationsdaten wurden über den Textimportassistenten in die logische Datei Mandant_KommDaten und das Mandatsverzeichnis in die logische Datei Mandant eingelesen.

Beschreibung TXT-Import

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_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"**)

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

Als Ergebnis wird bei der Ausführung des Scripts 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]
      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]
[END]
<END>
END_OF_FILE
```

3.7 Das Zusammenführen mehrerer Quelldateien

In der Applikation TXTImport ist es möglich, Informationen aus mehreren Textdateien in eine SBS Rewe neo® 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 Datei)
- 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:

```
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
```

Abb. 18: Mandant Adressen.CSV (allgemeine Mandantenadressdaten)

```
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;
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;0171-987654321;Mobiltelefon;
```

Abb. 19: Mandant Kommunikation.CSV (Kommunikationsdaten 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
```

Abb. 20: Mandant Bank.CSV (Bankverbindungen der Mandanten)

Über ein geeignetes Script sollen diese drei Textdateien zu einer SBS Rewe neo® Austauschformatdatei zusammengefasst werden.

Hinweis: Die Quelldaten sollten immer in sortierter Form vorliegen.

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 

. 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 Quelldateien.

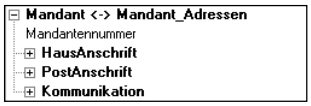


Abb. 21: Mandant

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

Die Objektstruktur stellt sich jetzt so dar.

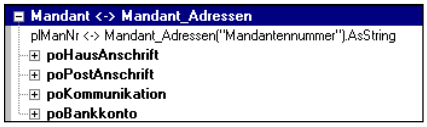
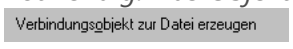


Abb. 22: Objekt Mandant

*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.*

Um das gewünschte Ergebnis zu erzielen, sind noch einige Änderungen an der Objektstruktur notwendig. Das Objekt Hausanschrift wird markiert und über den Kontextmenüpunkt 

ein Verbindungsobjekt über diesem eingefügt.

Abb. 23: Objektexplorer

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

Abb. 24: Objektexplorer

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

Die Objektstruktur sieht dann so aus:

Abb. 25: Objektstruktur

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

Abb. 26: Objektstruktur

Im weiteren Verlauf der Scripterstellung wird über dem Objekt Kommunikation ein Datei Verbindungs-Objekt erzeugt und dieses Datei Verbindungs-Objekt über das Kontextmenü ***Verbinden mit Datei*** *mit der Datei Mandant_KommDaten verbunden. 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)*

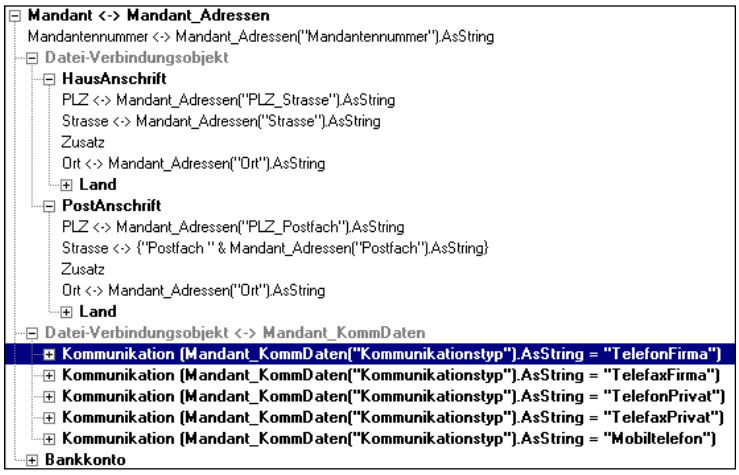


Abb. 27: Feldverknüpfung

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

siehe Dokument Kommunikationsmittel in der Kanzleiakte

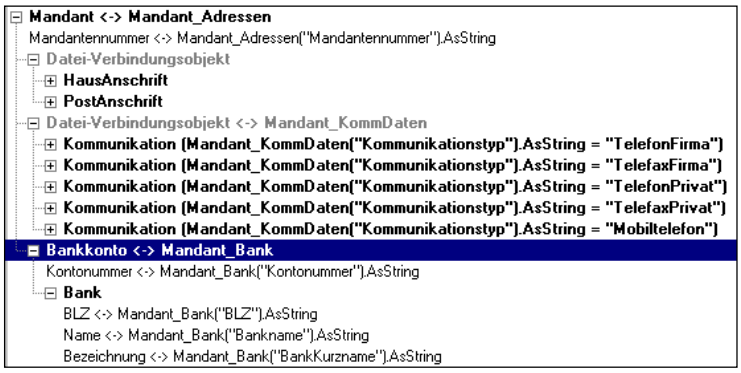
Typ = 1 - Telefon
Typ = 2 - Fax
Typ = 3 - eMail
Typ = 4 - Telex
Typ = 5 - Daten

Abb. 28: Verbindungsobjekte

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.



Damit ist die Scripterstellung abgeschlossen und die Scriptausführung über die Symbolschaltfläche  *liefert als Ergebnis die SBS Rewe neo® Austauschformatdatei auf der folgenden Seite.*

SBS Rewe neo® Austauschformatdatei nach Ausführung des Scripts:

```
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]
      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]
  [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
  [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
```

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

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)
```

Abb. 29: Inhalt einer Bankdatei

In dieser Datei sind lediglich die Bankbezeichnungen und Bankkurzbezeichnungen hinterlegt. Während der Konvertierung in das SBS Rewe neo® 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:

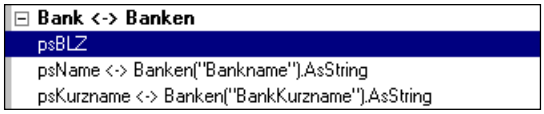


Abb. 30: Banken

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.

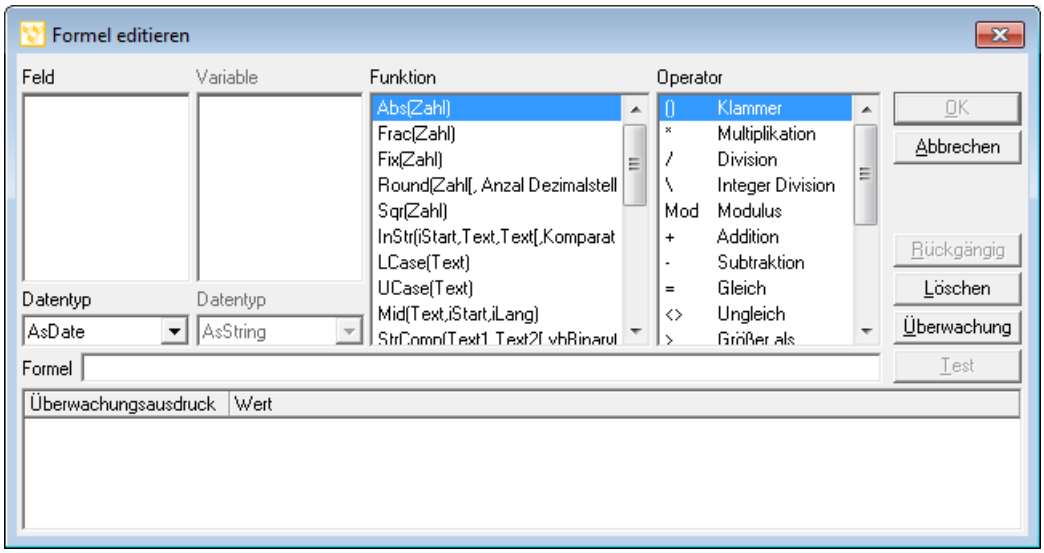
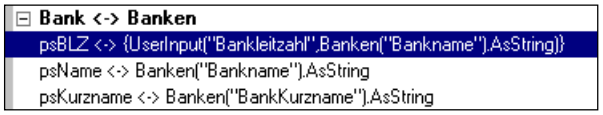


Abb. 31: Formel editieren

Durch betätigen der Schaltfläche **Test** kann die Funktion überprüft werden.

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



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.
Nach betätigen der Schaltfläche **Ok** erfolgt dann in der 2. Phase die eigentliche Konvertierung in das SBS Rewe neo® Austauschformat. Als Ergebnis der Ausführung des so erstellten Scriptes wird das Metafile auf der folgenden Seite erstellt.

3.8.2 SBS Rewe neo® Austauschformat Datei:

```
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
```

3.8.3 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.

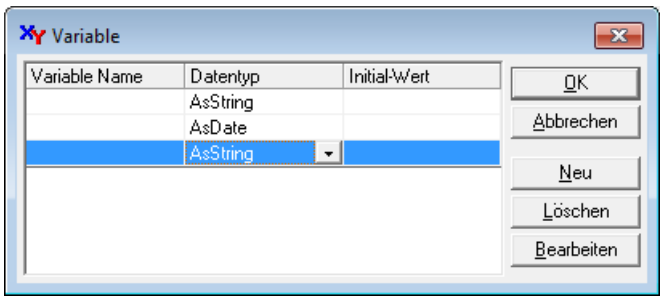


Abb. 32: Variable

Im Aktionseditor wird eine Aktion erstellt, 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:

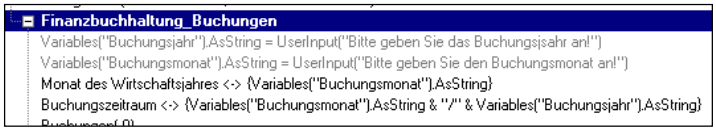


Abb. 33: Objektstruktur

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

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 Nr. 5 (Seite 26) 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 Funktionseditor 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.

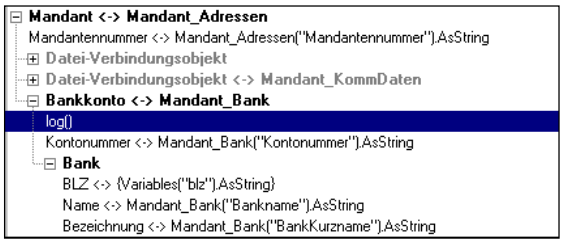


Abb. 34: Objektstruktur

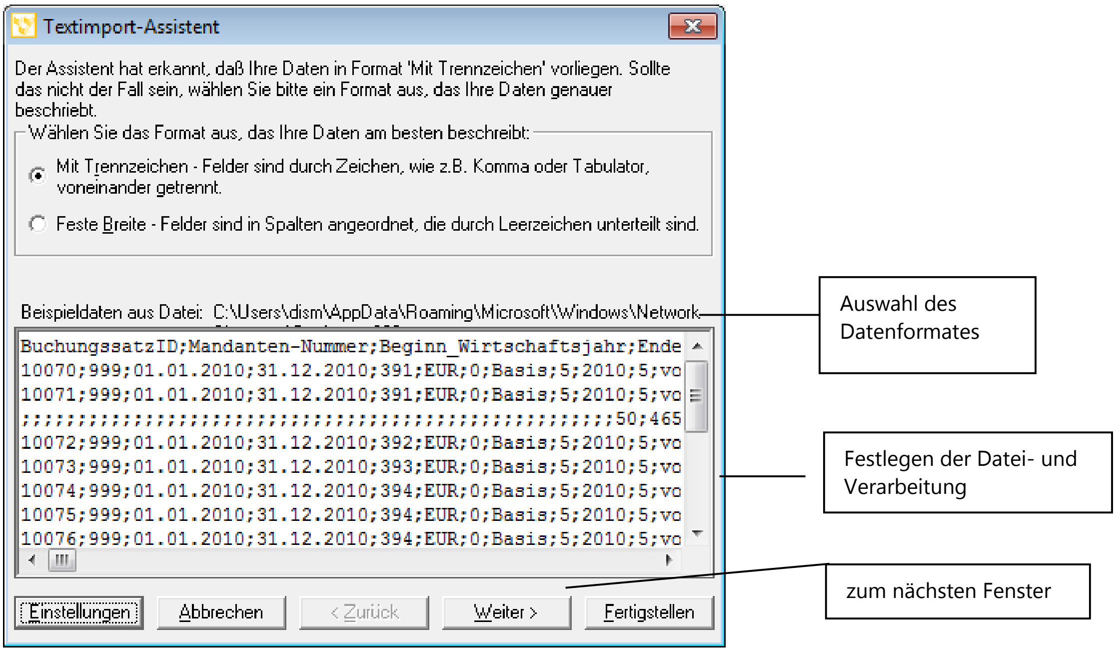
Als Ergebnis liefert diese Funktion in der Datei <C:\Bankenimport.log> Protokolleinträge, sofern fehlerhafte Bankverbindungen in der Textdatei übergeben werden.

4 Anhang

4.1 Der Textimportassistent

Über den Textimportassistenten erfolgt das Einlesen von Quelldateien (Textdateien, Excel-Dateien), die in das SBS Rewe neo® Austauschformat konvertiert werden sollen.

Nachdem über den Menüpunkt Datei -> Neue Textdatei eine Textdatei im Format *.asc, *.chr, *.csv, *.dat, *.txt ausgewählt wurde, wird das folgende Formular geöffnet.

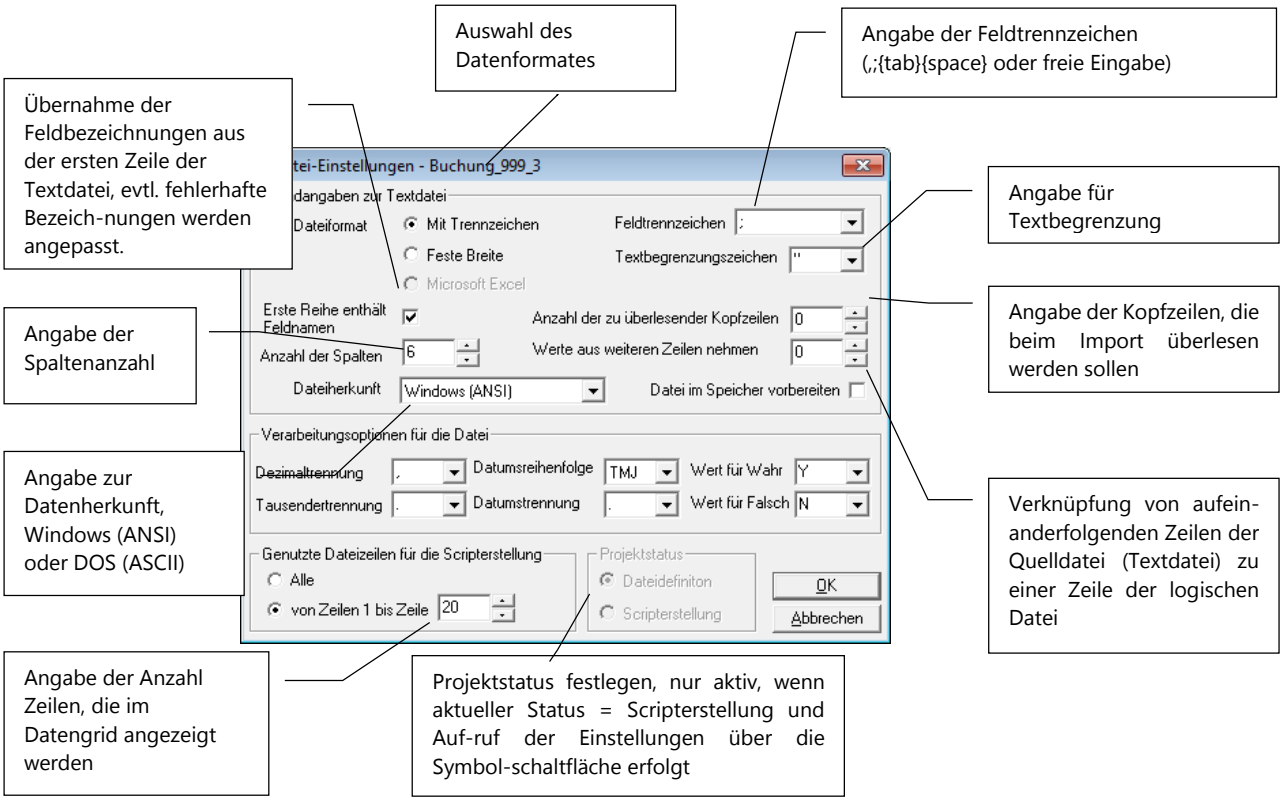


In diesem Formular wird das Eingabeformat der Quelldatei festgelegt.

Mit Trennzeichen	Verwendung bei CSV-Dateien, 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 **Einstellungen** können weitere Angaben zum Format der Quelldaten und zu den Verarbeitungsoptionen vorgenommen werden.

Beschreibung TXT-Import



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):

TMJ	(Tag Monat Jahr)	z.B.	30052001
TJM	(Tag Jahr Monat)	z.B.	30-2001-05
MTJ	(Monat Tag Jahr)	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:		
"."	(Punkt)	z.B.	29.10.2001
"-"	(Bindestrich)	z.B.	15-9-01
"/"	(Schrägstrich)	z.B.	4/6/99
":"	(Doppelpunkt)	z.B.	9:12:87
{no}	(kein Trennzeichen)	z.B.	150698

Wert für Wahr: logischer Wert für "Wahr" ist repräsentiert als:
"True"
"Ja"
"Yes"
"On"
"T"
"Y"
"J"
"1"

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 das Symbol  , 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.

Einlesen von Daten im Format Mit Trennzeichen

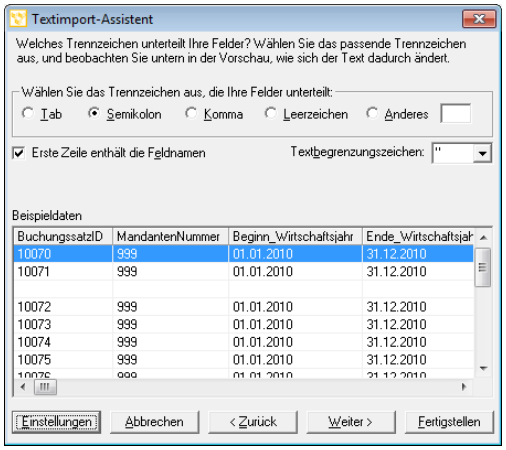


Abb. 35: Assistent Txt-Import

Nachdem über die Schaltfläche **Einstellungen** alle erforderlichen Angaben gemacht wurden, gelangt man über **Weiter** in das nächste Fenster in dem noch einmal Einstellungen zu dem gewählten Format vorgenommen werden können.

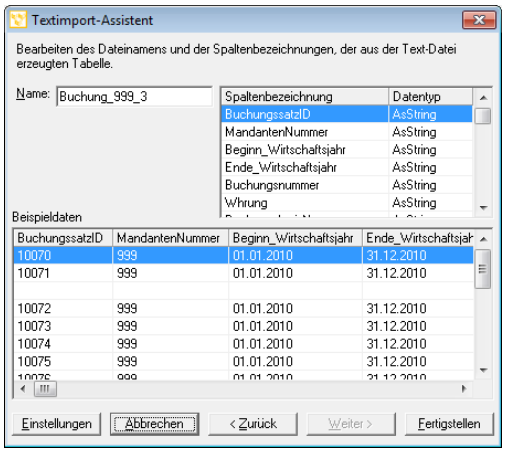


Abb. 36: Assistent Txt-Import

Mit **Weiter** gelangt man in das folgende Fenster, 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 Beispiel-werte aus der Quelldatei angezeigt.

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

4.1.1 Einlesen von Daten im Format Feste Breite

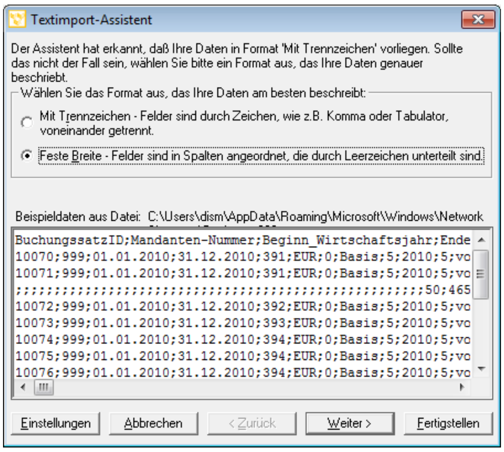


Abb. 37: Assistent Txt-Import

Nachdem über die Schaltfläche **Einstellungen** alle erforderlichen Angaben gemacht wurden, gelangt man über **Weiter** in das nächste Fenster, in dem die Festlegung der Spaltenbreite der Datenfelder vorgenommen wird. Zum Erstellen einer Feldtrenn-linie 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.

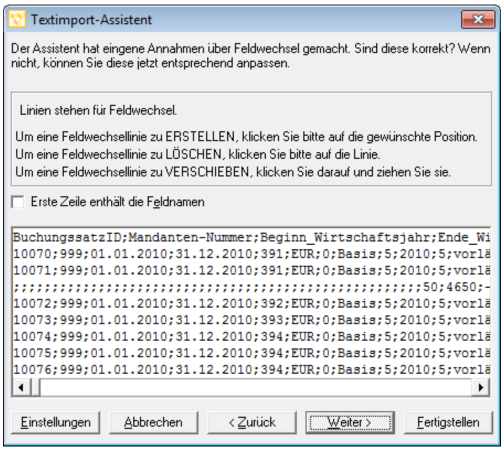


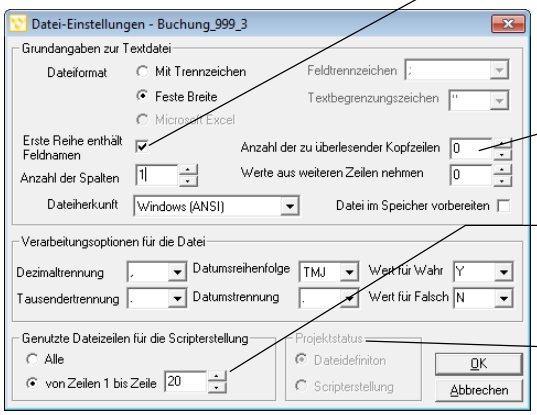
Abb. 38: Assistent Txt-Import

Mit **Weiter** gelangt man in das folgende Fenster, 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.

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

4.1.2 Einlesen von Daten im EXCEL - Format

Über die Schaltfläche **Einstellungen** 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.



Übernahme der Feldbezeichnungen aus der ersten Zeile der Textdatei, eventuell fehlerhafte Bezeichnungen werden angepasst.

Angabe der Kopfzeilen, die beim Import überlesen werden sollen

Angabe der Anzahl Zeilen, die im Datengrid angezeigt werden

Projektstatus festlegen, nur aktiv, wenn aktueller Status = Scripterstellung und Aufruf der Einstellungen über die Symbol-schaltfläche erfolgt

Abb. 39: Assistent Txt-Import

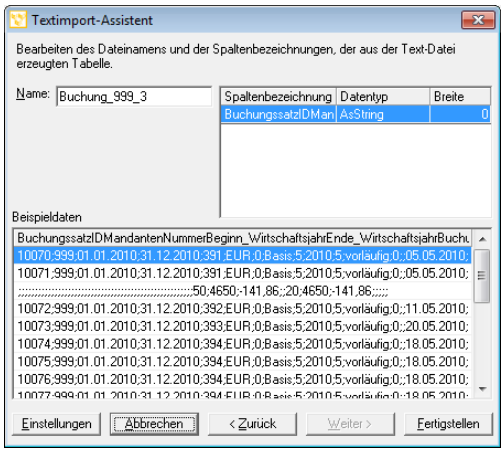


Abb. 40: Assistent Txt-Import

Nachdem über **Einstellungen** alle erforderlichen Angaben gemacht wurden, gelangt man über die **Weiter** in das nächste Fenster, 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 **Fertigstellen** wird die unter Einstellungen festgelegte Anzahl Zeilen in das Datengrid eingelesen.

4.2 Der Formel- / Aktionseditor

Der Formel - Editor ermöglicht es, das Erstellen von Ausdrücken (Berechnungsformeln) zu vereinfachen. Der Aufruf erfolgt mittels der Symbolschaltfläche ... 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).

Das Editorfenster enthält der Reihe nach: Liste der Felder aus den aktuellen Textdateien, Liste der Benutzervariablen, Liste der wichtigsten Funktionen 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.

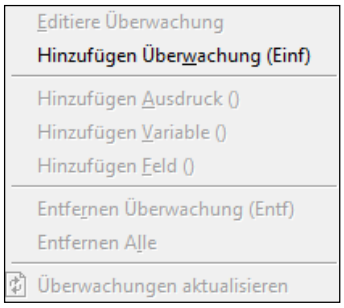
Die Schaltfläche **Überwachung** bietet dem Anwender die Möglichkeit, aktuelle Werte von mehreren Ausdrücken zu überwachen. Sie dient ebenfalls als Schalter für das Ein- und Ausblenden der Tabelle zur Ausdrucksüberwachung.

Mit **Rückgängig** werden die letzten ausgeführten Eingaben gelöscht.

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

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

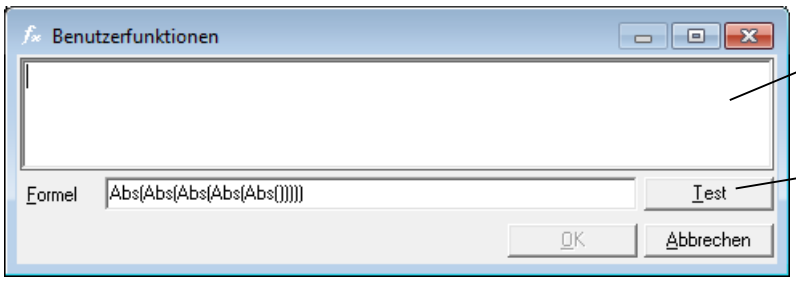
Durch Betätigen der Schaltfläche **Test** wird versucht, die in der Eingabezeile hinterlegte Formel zu berechnen, was zur Kontrolle des erstellten Berechnungsausdrucks dient. Es erfolgt eine Syntaxprüfung und Berechnung der eingegebenen Formel.

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

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.

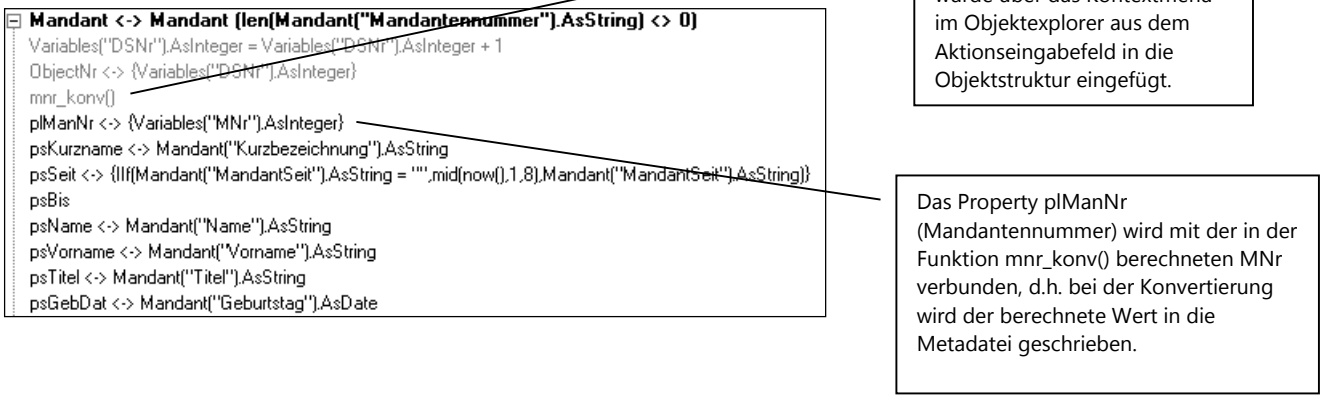
Der Aufruf des Funktionseditors erfolgt entweder über den Menüpunkt **Bearbeiten** -> **Funktion** oder über die Symbolschaltfläche in der Programmoberfläche. Es öffnet sich das folgende Formular.



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.

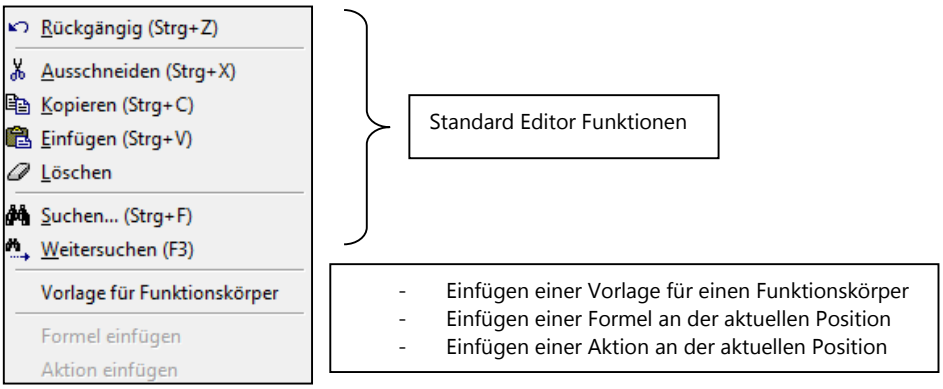
Im Eingabefeld **Formel** im unteren Teil des Formulars kann ein Testausdruck eingegeben werden, der dann über die Schaltfläche **Test** berechnet wird. Falls die Funktion fehlerhaft ist, wird eine Fehlermeldung angezeigt und der Cursor wird in die Code Zeile gesetzt, in welcher der Fehler erkannt wurde.

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



Beschreibung TXT-Import

Kontextmenü im Funktionseditor:



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 Formel-editor eingefügt.

Aktion einfügen: Der Aktionseditor wird aufgerufen. Die gebildete Aktion wird an der aktuellen Cursorposition in den Funktionseditor eingefügt.

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 sogenannte 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)

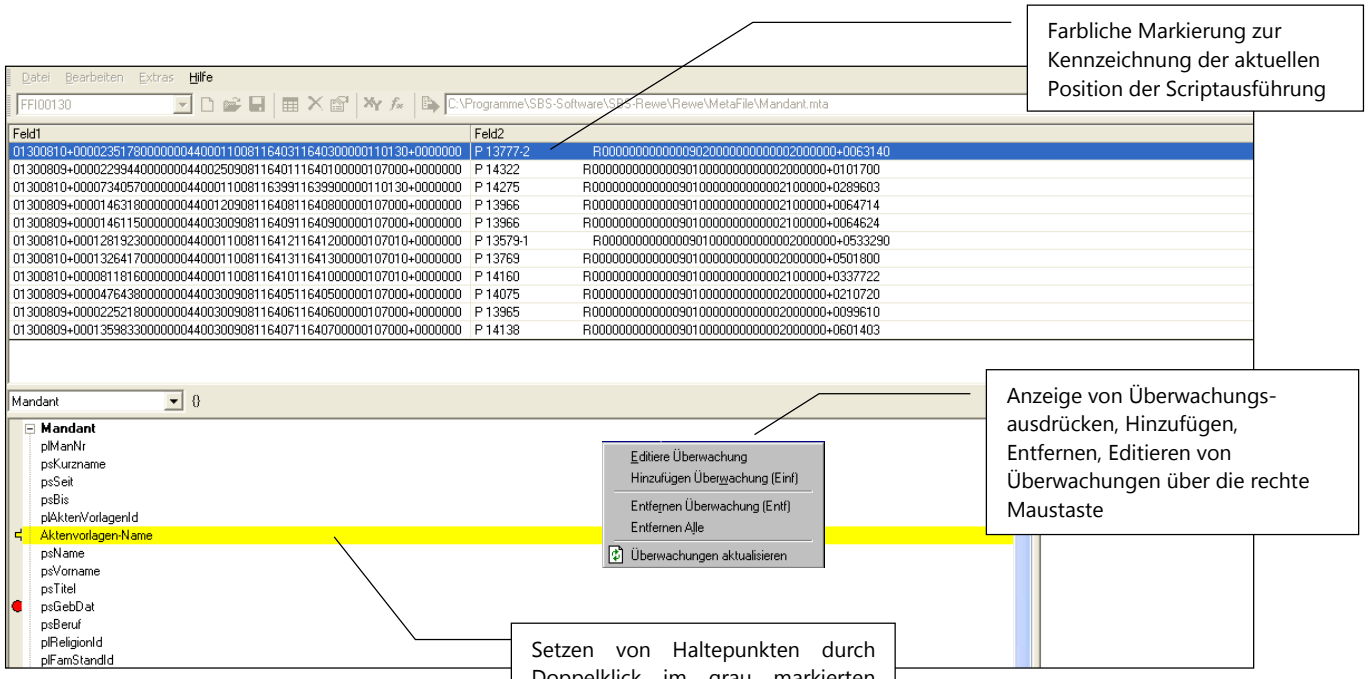


Abb. 41: Bildschirmansicht im Debug-Modus: