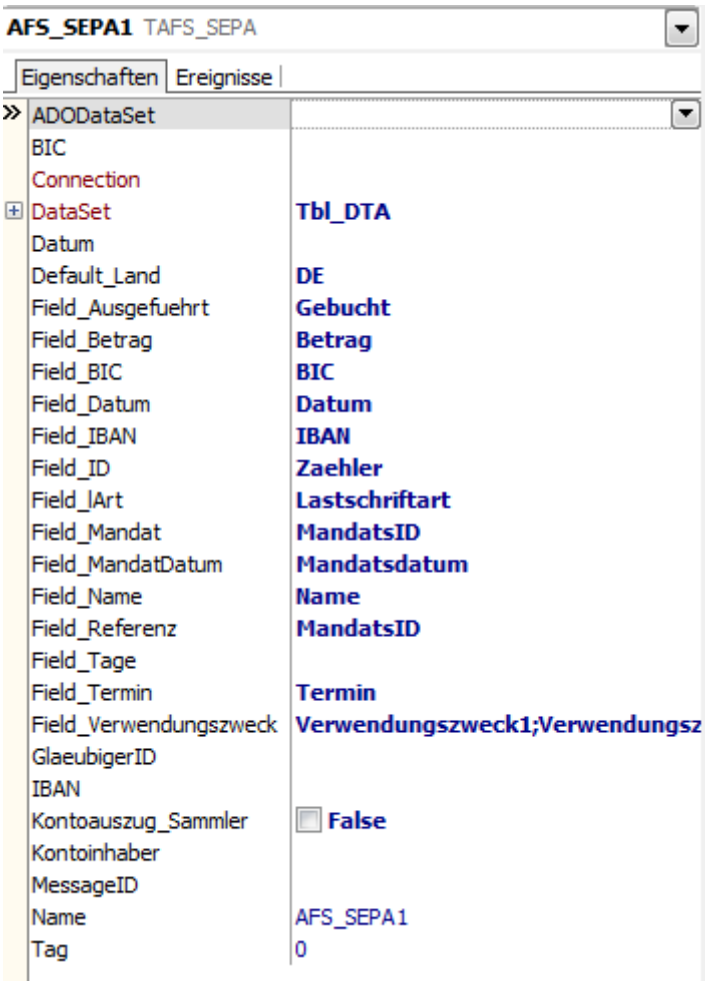
TAFS_SEPA = class(TComponent)



Funktionsumfang:

- SEPA Überweisungen

- SEPA Lastschriften (B2C)

- SEPA Firmenlastschriften (B2B) erzeugen

- IBAN Berechnungen aus Blz und Kontonummern (Länderübergreifend)

- BIC Ermittlung aus Länderdatenbanken (DE,AT und CH im Lieferumfang)

- IBAN Prüfsummencheck

- Gläubiger ID Nummerncheck

Lizenz:

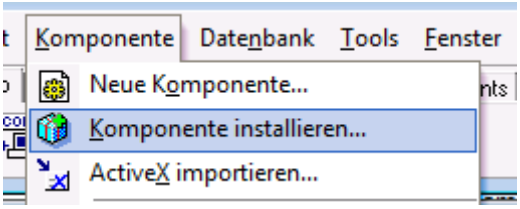
Der Käufer/Lizenznehmer erhält eine nicht ausschließliche Lizenz die Komponente innerhalb seines Produktes einzubinden und in kompilierter Form beliebiger Anzahl an seine Kunden zu vertrieben. Der Quellcode ist hiervon ausgenommen, dieser ist unter Verschluss zu halten und weder Kunden, noch in sonstiger Art und Weise zu veröffentlichen.

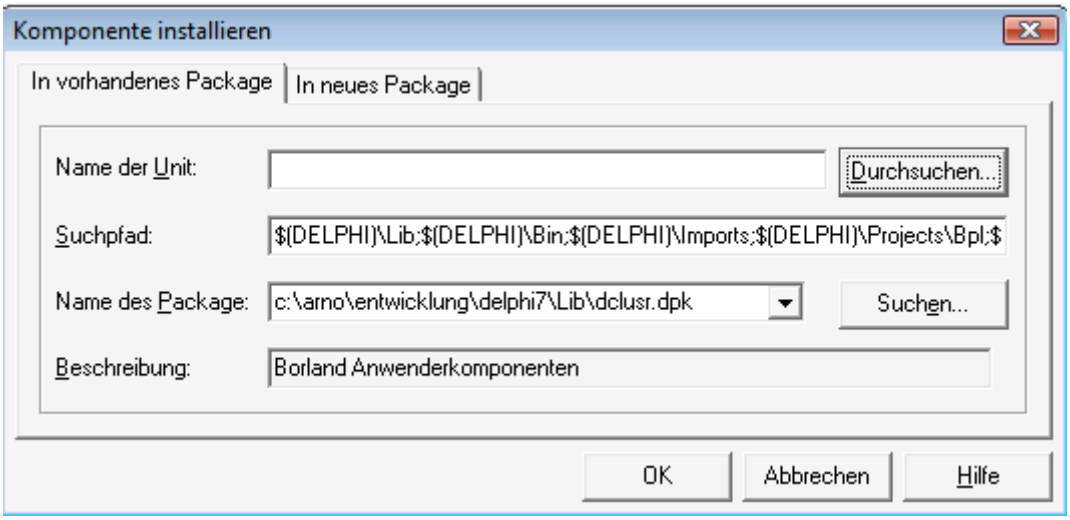
Der Lizenznehmer hat selbst dafür Sorge zu tragen, dass sein Endprodukt reibungslos funktioniert, den Anforderungen seiner Lizenznehmer entspricht und fehlerfrei arbeitet. Der Lizenzgeber übernimmt hierzu keinerlei Haftung. Der Lizenznehmer stellt den Lizenzgeber ausdrücklich von allen Haftungsansprüchen frei, die Ihm im Zuge des Einsatzes die Komponenten entstehen.

Mit dem Einbau der Komponenten oder von Teilen der Codes in sein Produkt stimmt der Lizenznehmer ausdrücklich diesen Lizenzbedingungen zu.

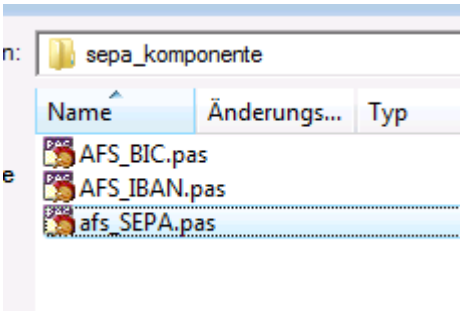
Installation:

Hier am Beispiel von Delphi7:

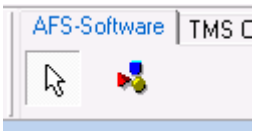




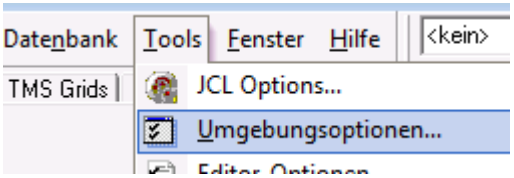
Wählen Sie bei Name der unit die Datei AFS_SEPA.PAS aus:

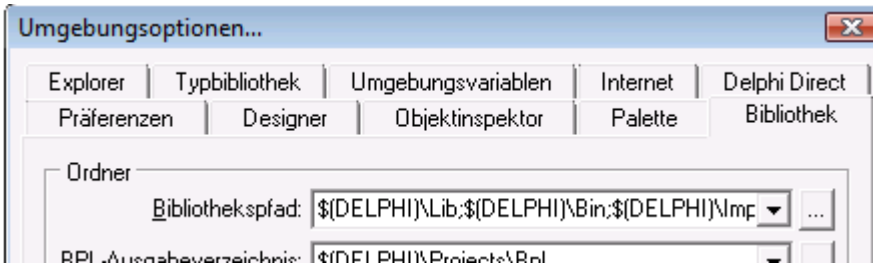


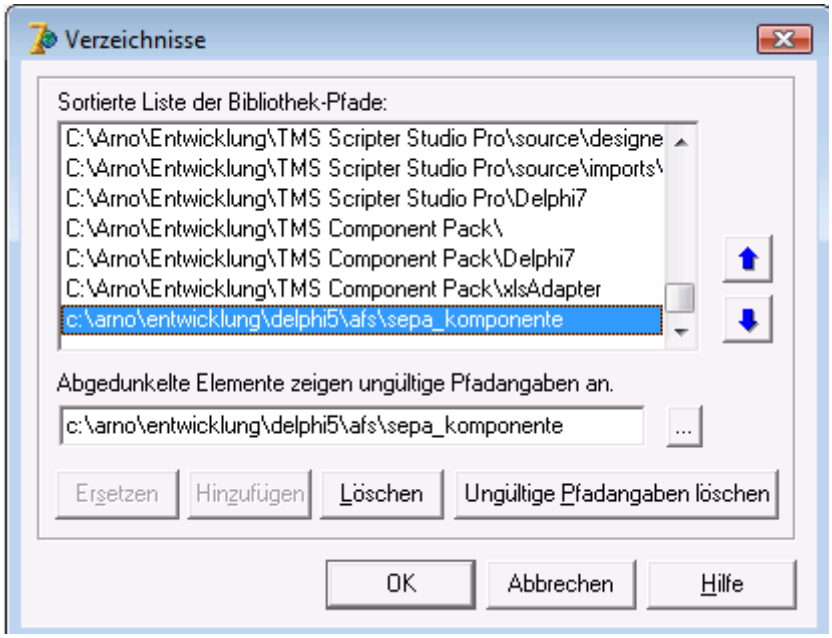
Nun wird die Komponente automatisch in das gewählte Package installiert und steht unter der karteikarte AFS-Software zur verfügung:



Damit die units beim Kompilieren Ihrer Anwendung gefunden werden, müssen Sie noch das Verzeichnis der Units in den Bibliothekspfad integrieren:



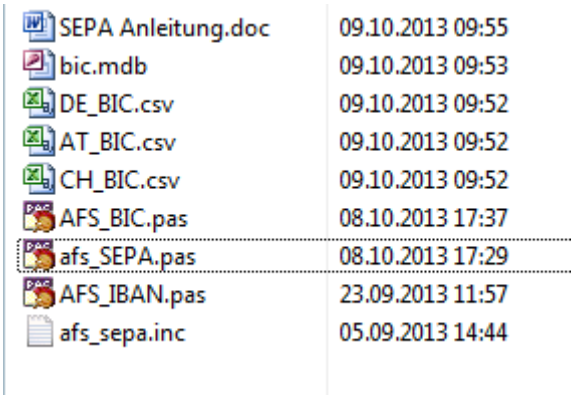




Hier als Beispiel für meinen Pfad. Hier wählen Sie das Verzeichnis Ihrer Installation aus.

Nun ist die SEPA Komponente einsatzbereit.

Interner Aufbau:



- SEPA Anleitung	*diese Anleitung*
- bic.mdb	*Access Datenbank mir Bankleitzahlen zur BIC Ermittlung*
- DE_BIC.csv	*deutsche Bankleitzahlen*
- AT-BIC.csv	*österreichische Bankleitzahlen*
- Ch_BIC.csv	*schweizer Bankleitzahlen*
- AFS_BIC.pas	*unit für Bic Ermittlungen*
- AFS_IBAN.pas	*unit für IBAN Ermittlung*
- afs_SEPA.pas	*unit der eigendlichen Komponente*
- afs_sepa.inc	*Einstellungen für die Komponente*

Die Datei afs_sepa.inc enthält einige Einstellungen, die die Eigenschaften der Komponente beeinflussen:

```
//
// Einstellung für die SEPA Komponente
//
// (c) 2013 by AFS-Software GmbH & Co.KG
//        Klaustor 3
//        36251 Bad Hersfeld (Germany)
//

{$IFDEF VER260} // XE5
 {$DEFINE NEWDELPHI}
{$ENDIF}
{$IFDEF VER250} // XE4
 {$DEFINE NEWDELPHI}
{$ENDIF}
{$IFDEF VER240} // XE3
 {$DEFINE NEWDELPHI}
{$ENDIF}
{$IFDEF VER230} // XE2
 {$DEFINE NEWDELPHI}
{$ENDIF}

//
// Bankleitzahl / BIC aus ADO DB (z.B. Access Datenbank)
//
{$DEFINE DB_ADO}
```

NEWDELPHI Definiert neuere Delphiversionen mit erweitertem Sprachumfang, wie z.B. die StringSplittfunktion, die z.B. in Delphi5 noch nicht zur Verfügung steht und diese Funktionen „nachrüstet“.

Ein wichtiger Punkt ist **DB_ADO** *. Dies gibt an, ob die BIC Informationen, aus ADO Dataset kommen können (z.B. Access, SQL-Server, usw.) oder nur aus den mitgelieferten CSV Dateien.*

Funktionen der Komponenten:

function Check_CI_ID(CIID: AFS_String): Boolean;

Prüft die GläubigerID

Beispiel:

```
procedure TMandant_Ass.DBEdit40Exit(Sender: TObject);
begin
 (Sender as TDBEDIT).Color:=clWindow;
 if length(p_Trim(Tbl_Mandant.FieldByName('GlaeubigerID').AsString))>0 then begin
  if AFS_SEPA1.Check_CI_ID(Tbl_Mandant.FieldByName('GlaeubigerID').AsString) then begin
   (Sender as TDBEDIT).Color:=$B0FFB0;
  end else begin
   (Sender as TDBEDIT).Color:=$B0B0FF;
  end;
 end;
end;
```

function Calc_IBAN(Land,Kontonr,Blz : AFS_String; var Warnung:string): AFS_String;

Berechnet die IBAN aus Kontonummer und Bankleitzahl, Die IBAN wird zurückgegeben. Falls Probleme bei der IBAN Berechnung festgestellt wurden wird dieses im String Warnung übergeben.

Beispiel:

```
procedure TMandant_Ass.DBEdit27Enter(Sender: TObject);
var IBAN,w:string;
begin
 if (Tbl_Mandant.FieldByName('IBAN1').AsString='') then begin
  if (Tbl_Mandant.FieldByName('Kontonummer1').AsString<>'') and
(Tbl_Mandant.FieldByName('Bankleitzahl1').AsString<>'') then begin

IBAN:=AFS_SEPA1.Calc_IBAN(Tbl_Mandant.FieldByName('Land').AsString,Tbl_Mandant.FieldByName('Kontonu
mmer1').AsString,Tbl_Mandant.FieldByName('Bankleitzahl1').AsString,w);
   if IBAN<>'' then begin
    Tbl_Mandant.FieldByName('IBAN1').AsString:=IBAN;
    if w<>'' then begin
     ShowMessage('Automatische IBAN Ermittlung','','Achtung es liegt eine Warnung vor'+#13+w);
    end;
   end;
```

```
  end;
 end;
end;
```

function Check_IBAN(IBAN: AFS_String):Boolean;

Prüft die IBAN

Beispiel:

```
procedure TMandant_Ass.DBEdit27Exit(Sender: TObject);
begin
 if length(p_Trim(Tbl_Mandant.FieldByName('IBAN1').AsString))>0 then begin
  if AFS_SEPA1.Check_IBAN(Tbl_Mandant.FieldByName('IBAN1').AsString) then begin
   DBEdit27.Color:=$B0FFB0;
  end else begin
   DBEdit27.Color:=$B0B0FF;
  end;
 end;
end;
```

function Find_BIC(Land,Blz: AFS_String; var Warnung:string):AFS_String;

Gibt die BIC anhand der Bankleitzahl zurück, Unterstützt DE,AT,CH

Hierzu ist es wichtig, dass die mitgelieferten CSV Dateien der Bankleitzahlen im Programmverzeichnis zu finden sind!

Beipsiel:

```
procedure TMandant_Ass.DBEdit28Enter(Sender: TObject);
var Blz,Bank,BIC:string;
begin
 if (Tbl_Mandant.FieldByName('BIC1').AsString='') then begin
  if (Tbl_Mandant.FieldByName('Bankleitzahl1').AsString<>'') then begin
    Tbl_Mandant.FieldByName('BIC1').AsString:= AFS_SEPA1.Find_BIC
(Tbl_Mandant.FieldByName('Land').AsString Tbl_Mandant.FieldByName('Bankleitzahl1').AsString,w);
  end;
 end;
end;
```

function DB_BICFind(Con: TADOConnection; Land,Blz: AFS_String; var Warnung:string):AFS_String;

*Sucht die BIC aus seiner ADO Datenquelle. Wie **Find_BIC** mit dem Unterschieddas eine TADOConnection übergeben wird, die die Blz Tabellen enthalten muss Aufbau **LAND**_BIC z.B. DE_BIC*

Im Lieferumfang ist hier die bic.mdb, die DE,AT,CH enthält

function DB_BICShow(Tbl:TADODataSet; Land,Blz: AFS_String; var Warnung:string):AFS_String;

Selektiert die gesuchte BLZ au seiner ADO Datenquelle

*wie **DB_BICFind** mit dem Unterschied, das nicht keine TADOConnectionübergeben wird, da diese der Komponente zugewiesen sein muss. Das übergebene TADODataSet enthält die gefunden Ergebnisse und kann so im weiteren Verlauf angezeigt werden.*

function DataSetToSepa(onlyaktuell:Boolean=True):integer;

Erzeugt aus einem TDataSet internen Objekte TSEPA, aus der Sepadateien erzeugt werdenkönnen.

Das TDataSet und alle andere Feldzuweisungen müssen der Komponente vorher zugewiesen worden sei.

***Onlyaktuell** gibt an, ob nur der aktuelle Datensatz benutzt wird (True) oder alle Datensätze die im TDataset vorhanden sind (False).*

Beispiel:

```
AFS_SEPA1.MessageID:=Tbl_DTA.Fields[0].AsString;
AFS_SEPA1.S_Name:=Tbl_DTA.FieldByName('Name2').AsString;
AFS_SEPA1.IBAN:=Tbl_DTA.FieldByName('IBAN2').AsString;
AFS_SEPA1.BIC:=Tbl_DTA.FieldByName('BIC2').AsString;
AFS_SEPA1.GlaeubigerID:=Tbl_DTA.FieldByName('GlaeubigerID').AsString;
AFS_SEPA1.Kontoauszug_Sammler:=False;
if Tbl_DTA.FieldByName('Termin').AsString<>'' then begin
 AFS_SEPA1.Datum:=Tbl_DTA.FieldByName('Termin').AsDateTime;
end else begin
 AFS_SEPA1.Datum:=now;
end;
//
AFS_SEPA1.Belege_Clear;
if AFS_SEPA1.DataSetToSepa(True)>0 then  memo1.Lines.Text:=AFS_SEPA1.XML_KOPF(Sepa_Art);
```

function Belege_Clear:Boolean;

Leer die TSEPA Objekteaus dem internen Array

Beispiel: siehe oben

function XML_KOPF(Art : integer): AFS_String;

Erzeugt eine SEPA Dateistring und übergibt diesen

Art = 0 -> Überweisung
Art = 10 -> Lastschrift (Basis)
Art = 11 -> Lastschrift (Firmen B2B)

Beispiel: siehe oben

Es ist wichtig: es kann nur EINE Belegart (Überweisung, Basis-Lastschrit oder B2B Lastschrift erstellt werden. Des weiteren kann in einer Datei nur ein Auftraggeberkonto angesprochen werden!

function Beleg_Add(eSepa: TSepa):Boolean;

Fügt ein TSEPA Objekt ein

Eigenschaften der Komponente

ADODataSet

Ein TAdoDataSet, die Connection auf die BIC Datenbank enthält.

Connection

Die Connection auf die BIC Datenbank enthält.

Datum

Datum der Übergabe, kann leer bleiben

Default_Land

Für die Ermittlung der BIC und IBAN wird eine Länderinformation benötigt. Dies gibt das Standardland an.

GlaebigerID

Für Lastschriften wird die GlaebigerID benötigt

Kontoinhaber

Absender Kontoinhaber

IBAN

Absender IBAN

BIC

Absender BIC

Message_ID

Eindeutige ID der SEPA Datei (z.B. Zähler des Datensatzes der Ausgeführt wird)

Kontoauszug_Sammler

Soll als Sammler gebucht werden oder Einzelbeleg. Sammler wird automatisch gesetzt, wenn mehrere Belege übergeben wird.

Achtung: die Sparkassen können laut Auskunft erst im Laufe des Nov. 2013 Einzel Belege als SEPA Lastschriften annehmen!

DataSet

TDataset, welches die SEPA Datensätze enthält, die erzeugt werden sollen

In Anschluß werden die Feldnamen übergeben, die die notwendige Feldzuordnung sicherstellen:

FIELD_Ausfegührt

z.Z. keine Funktion

FIELD_BETRAG *Feldbezeichnung des Betrags*

FIELD_BIC *Feldbezeichnung der BIC*

FIELD_DATUM *Feldbezeichnung des Belegdatum*

FIELD_IBAN *Feldbezeichnung der IBAN*

FIELD_ID *Feldbezeichnung des eindeutigen Belegs*

FIELD_LART *Feldbezeichnung der Lastschriftart*

OOFF = einmalig
FRST = erste
RCUR = regelmäßige
FNAL = letzte Lastschrift

FIELD_Mandat *Feldbezeichnung des Lastschrifts-Mandats*

FIELD_MandatDatum *Feldbezeichnung des Lastschrifts-Mandatsdatum*

FIELD_Name *Feldbezeichnung des Kontoinhaber*

FIELD_REFERENZ *Feldbezeichnung des Lastschrifts-Mandats*

FIELD_TAGE *Feldbezeichnung des Lastschrifts-Fälligkeitstage*

FIELD_Termin *Feldbezeichnung des Ausführungstermin (Terminüberweisung, Lastschriftstermin)*

FIELD_Verwendungszweck *Feldbezeichnung des Verwendungszwecks (nur ein Feld mit 70 Zeichen ist erlaubt)*

TSEPA = Belegdefinition

Mit TSEPA werden die Sepa Belege der Komponente übergeben. Diese können manuell angelegt werden oder mit der Möglichkeit die Komponente an ein DataSet zu knüpfen automatisch auf dem DataSet erzeigt werden.

```
TSEPA = class(TComponent)
private
protected
public

 S_Free              // gibt selbst frei (True oder wird automatisch freigegeben
        : Boolean;

 ID,             // ID der Belegs (fortlaufendenr)
 Art             // Art = 0  -> Überweisung
                 // Art = 10 -> Lastschrift (Basis)
                 // Art = 11 -> Lastschrift (Firmen B2B)

        : Longint;
 lArt,           // Lastschriftart (nur bei Lastschriften)
                 //  OOFF = einmalig
                 //  FRST = erste
                 //  RCUR = regelmäßige
                 //  FNAL = letzte Lastschrift
 Name,                   // Kontoinhaber
 IBAN,                   // IBAN
 BIC,                    // BIC
 Referenz,               // Referenznummer
 Mandat,                 // Mandatsnummer für Lastschriften
 Mandataenderung,        // Mandatsänderung für Lastschriften (true/false)
 Verwendungszweck        // Verwendungszweck, eine Zeile max 70 Zeichen
        : AFS_String;

 Betrag                  // Zahl/Lastschriftbetrag
        : double;

 MandatDatum,            // Datum der Unterschrift des Mandats für Lastschriften
 Termin,                 // Ausführungstermin
 Datum                   // Belegdatum
        : TDate;

 Ausgefuehrt             // Erzeugt (ja/nein)
        : Boolean;

published


end;
```