Ob OLE DB, ADO oder Zeos die Möglichkeiten für Free Pascal um auf Datenbanken zuzugreifen

Zugriff mit Free Pascal über OLE DB (Object Linking and Embedding, Database Library)

Sicherlich gibt es mehrere Methoden um über Free Pascal auf Datenbanken zuzugreifen aber als Erstes nun die kürzeste und zugleich schlecht dokumentierte Variante. OLE DB - Object Linking and Embedding Databases, (Low Level) COM (Component Object Model) basiertes Interface wurde erstellt um standardisierten Zugriff auf Datenquellen zu ermöglichen.

Diese Variante stellt nur sehr geringe Anforderungen an Free Pascal, es werden lediglich die Units ComObj und Variants benötigt, die Anforderungen an den Entwickler sind jedoch sehr hoch denn unter Lazarus bietet diese Variante keine Codevervollständigung, man sollte alle Befehle bereits kennen denn das was von Lazarus vorgeschlagen wird hilft einem nicht weiter.

Im folgenden Beispiel sind auch direkt Beispiele für weitere Datenbanken eingefügt damit ich dieses Beispiel nicht bei jeder weiteren Datenbank aufführen muss.

OLE DB -> Jet OLE DB Provider -> Jet -> MSAccess

OleDB.lpr (/code/OleDB.lpr?mode=download) Pascal (2,48 kByte) 01.01.2014 20:35

```
program Ole;
{$APPTYPE CONSOLE}
{$mode objfpc}{$H+}
uses ComObj, Variants;

function IIF(b: boolean; sTrue: string; sFalse: string = ''): string;
begin if b then result := sTrue else result := sFalse; end;

const
  adOpenForwardOnly = $00000000;
  adLockReadOnly = $00000001;
var
  cn, rs: OleVariant;
  s: string;
  i: integer;
begin
//  CoInitialize(NIL);
//connect db
  cn := CreateOleObject('ADODB.Connection');
{$ifdef access} // Access over MS Jet OLE DB 4.0
  cn.Open('Provider=Microsoft.Jet.OLEDB.4.0;Data Source=test.mdb');
{$endif}
{$ifdef mdbODBC} // Access 97-2003, Microsoft Access ODBC Driver
cn.Open('Driver={Microsoft Access Driver (*.mdb)};Dbq=D:\test.mdb;Locale
Identifier=1031;Uid=Admin;Pwd=;');
{$endif}
{$ifdef accdbODBC} // Access 2007-2013, Microsoft Access accdb ODBC Driver
cn.Open('Driver={Microsoft Access Driver (*.accdb)};Dbq=d:\test.accdb;Locale
Identifier=1031;Uid=Admin;Pwd=;');
{$endif}
```

```
{$ifdef accessNETWORK} // OLE DB Provider for Microsoft Directory Services
// uses Microsoft Access Database Engine 2010
cn.Open('Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=\\server\shareDB\test.mdb;');
{$endif}
{$ifdef mysqlnew} // MySQL ODBC 5.2
  cn.Open('Provider=MSDASQL;DRIVER={MySQL ODBC 5.2a
Driver};SERVER=localhost;DATABASE=testsql;UID=root;PWD=root;');
{$endif}
{$ifdef mysqlold} // MySQL ODBC 3.51
  cn.Open('Provider=MSDASQL.1;Extended Properties="Driver={MySQL ODBC 3.51
Driver};Server=localhost;Port=3306;Database=testsql;User=root;Password=root;Opt
on=3;"');
{$endif}
{$ifdef mysql} // MSSQL
  cn.Open('Provider=SQLOLEDB.1;Initial Catalog=TestDB;Data
Source=localhost;Persist Security Info=False;User ID=root;Password=root;');
{$endif}
{$ifdef mysqlexpress} // MSSQL Express 2008
  cn.Open('Provider=SQLNCLI10.1;Initial Catalog=TestDB;Data
Source=localhost\SQLEXPRESS;Persist Security Info=False;User
ID=root;Password=root;');
{$endif}
//select
  rs := CreateOleObject('ADODB.Recordset');
  rs.Open('SELECT * FROM testTable', cn, adOpenForwardOnly, adLockReadOnly);
  s := '';
  //list fieldnames
  for i:=0 to rs.Fields.count-1 do
    s := s + IIF(i>0, ';') + rs.Fields[i].Name;
  s := s + #13#10;
  //list values
  while not rs.eof do
  begin
    for i:=0 to rs.Fields.count-1 do
      s := s + IIF(i>0, ';') + VarToStr(rs.Fields[i].Value);
    s := s + #13#10;
    rs.MoveNext;
  end;
  WriteLn(s);
  //close recordset
  rs.Close;
  rs := unassigned;
//close connection;
  cn.Close;
  cn := unassigned;
//  CoUnInitialize;
end.
```

Zugriff mit Free Pascal über ADODB (Microsoft ActiveX Data Objects 2.8 Library)

Eine Abstraktionsschicht mehr ADO - ActiveX Data Objects, (High Level) einfach zu nutzende, Sprachen unabhängige API für den Zugriff auf von OLE DB unterstützte Datenbanken.

ADODB -> OLE DB -> Jet OLE DB Provider -> Jet -> MSAccess

Diese Variante benötigt die Units ADODB die sich mit dem Werkzeug "Import Type Library" von Lazarus aus der Datei msado15.dll (C:\Programme\Gemeinsame Dateien\System\ado\) erzeugen lässt.

Import ADODB : Microsoft ActiveX Data Objects 2.x-Objektbibliothek
C:\Program Files\Common Files\System\ado\msado15.dll adodb 6 1 tlb.pas (/downloads/adodb_6_1_tlb.pas) (189,84 kByte) 30.12.2018 21:29

Ado.lpr (/code/Ado.lpr?mode=download) Pascal (1,1 kByte) 21.12.2013 18:31

```
program Ado;
{$APPTYPE CONSOLE}
{$mode objfpc}{$H+}
uses Variants, adodb_6_1_tlb;

function IIF(b: boolean; sTrue: string; sFalse: string = ''): string;
begin if b then result := sTrue else result := sFalse; end;

var
  cn: _Connection;
  rs: _Recordset;
  s: string;
  i: integer;
begin
//  CoInitialize(NIL);
//connect db
  cn := CoConnection.Create;
{ $ifdef access} // Access over MS Jet OLEDB
  cn.Open('Provider=Microsoft.Jet.OLEDB.4.0;Data Source=test.mdb', '', '', 0);
{ $endif}
//select
  rs := CoRecordset.Create;
  rs.Open('SELECT * FROM testTabelle', cn, adOpenForwardOnly, adLockReadOnly,
adCmdText);
  s := '';
  //list fieldnames
  for i:=0 to rs.Fields.count-1 do
    s := s + IIF(i>0, ';') + rs.Fields[i].Name;
  s := s + #13#10;
  //list values
  while not rs.EOF_ do
  begin
    for i:=0 to rs.Fields.count-1 do
      s := s + IIF(i>0, ';') + VarToStr(rs.Fields[i].Value);
    s := s + #13#10;
    rs.MoveNext;
  end;
  WriteLn(s);
  //close recordset
```

```
  rs.Close;
  rs := unassigned;
//close connection;
  cn.Close;
  cn := unassigned;
//  CoUnInitialize;
end.
```

Wenn man sich dann noch ein wenig Mühe macht und die Objekte optimiert...

ADO (eigene Klassen) -> ADODB -> OLE DB -> Jet OLE DB Provider -> Jet -> MSAccess

myAdo.lpr (/code/myAdo.lpr?mode=download) Pascal (1,73 kByte) 21.12.2013 18:36

```
program myAdo;
{$APPTYPE CONSOLE}
{$IFDEF FPC}
  {$mode objfpc}{$H+}
{$ENDIF}
uses ado;

function IIF(b: boolean; sTrue: string; sFalse: string = ''): string;
begin if b then result := sTrue else result := sFalse; end;

var
  cn: TADOConnection;
  rs: TADORecordset;
  s: string;
  i: integer;
begin
  //connect db
  cn := TADOConnection.Create(nil);
{ $ifdef access} // Access over MS Jet OLEDB
  cn.Open('Provider=Microsoft.Jet.OLEDB.4.0;Data Source=test.mdb');
{ $endif}
{$ifdef mysqlnew} // MySQL ODBC 5.2
  cn.Open('Provider=MSDASQL;DRIVER={MySQL ODBC 5.2a
Driver};SERVER=localhost;DATABASE=testsql;UID=root;PWD=root;');
{$endif}
{$ifdef mysqlold} // MySQL ODBC 3.51
  cn.Open('Provider=MSDASQL.1;Extended Properties="Driver={MySQL ODBC 3.51
Driver};Server=localhost;Port=3306;Database=testsql;User=root;Password=root;Opt
on=3;"');
{$endif}
{$ifdef mysql} // MSSQL
  cn.Open('Provider=SQLOLEDB.1;Initial Catalog=TestDB;Data
Source=localhost;Persist Security Info=False;User ID=root;Password=root;');
{$endif}
{$ifdef mysqlexpress} // MSSQL Express 2008
  cn.Open('Provider=SQLNCLI10.1;Initial Catalog=TestDB;Data
Source=localhost\SQLEXPRESS;Persist Security Info=False;User
ID=root;Password=root;');
{$endif}
```

```
//select
  rs := TADORecordset.Create(cn);
  rs.Open('SELECT * FROM testTable');
  s := '';
  //list fieldnames
  for i:=0 to rs.Fields.count-1 do
    s := s + IIF(i>0, ';') + rs.Fields[i].Name;
  s := s + #13#10;
  //list values
  while not rs.eof do
  begin
    for i:=0 to rs.Fields.count-1 do
      s := s + IIF(i>0, ';') + rs.Fields[i].AsString;
    s := s + #13#10;
    rs.MoveNext;
  end;
  WriteLn(s);
  //close recordset
  rs.Close;
  rs.Free;
  //close connection;
  cn.Close;
  cn.Free;
end.
```

Zugriff mit Free Pascal über Zeos

Der große Vorteil von Zeos besteht darin das ein späterer Umstieg auf eine andere Datenbank leichter möglich ist, besonders wenn hierzu keine ADO Schnittstelle existiert oder wenn ein Systemumstieg geplant ist. Des weiteren existieren etliche visuelle Komponenten, die natürlich in den zuvor aufgeführten Varianten nicht zur Verfügung stehen.

Leider steht zur Zeit jedoch Zeos mit ADO Unterstützung nur für Delphi zur Verfügung, mit einem Patch und Unit OleDB lässt sich Zeos jedoch auch unter Free Pascal nutzen. Die Unit OleDB lässt sich aus der Datei oledb.h aus dem Microsoft Data Access SDK 2.8 herleiten was aber nicht einfach ist. Wie legitim der Download der oledb.pas von Google http://cbuilder-vcl.googlecode.com/svn/trunk/Source/vcl/oledb.pas ist, ist fraglich.

ZeosAdo.patch (/downloads/ZeosAdo.patch) (8,45 kByte) 30.12.2018 21:29

Im Beispiel wurden auch Zugriffe die nicht über ADO laufen eingebaut, hierfür werden unterschiedliche Librarys (DLLs) je Datenbank benötigt.

zeos.lpr (/code/zeos.lpr?mode=download) Pascal (1,46 kByte) 01.12.2013 15:06

```
program zeos;
{$APPTYPE CONSOLE}
{$mode objfpc}{$H+}
uses Classes, SysUtils, {Variants,} ZConnection, ZDataset;

function IIF(b: boolean; sTrue: string; sFalse: string = ''): string;
begin if b then result := sTrue else result := sFalse; end;
```

```
var
  cn: TZConnection;
  rs: TZQuery;
  i: integer;
  s: string;
begin
//connect db
  cn := TZConnection.Create(nil);
{$ifdef access} // Access over MS Jet OLEDB
  cn.Protocol := 'ado';
  cn.Database := 'Provider=Microsoft.Jet.OLEDB.4.0;Data Source=test.mdb';
  cn.Port := 0;
{$endif}
{$ifdef mysqlnew} // MYSQL 5
  cn.Protocol := 'mysql-5';
  cn.Database := 'testsql';
  cn.HostName := 'localhost';
  cn.Port := 3306;;
  cn.User := 'root';
  cn.Password := 'root';
{$endif}
{$ifdef mysql} // MSSQL
  cn.Protocol := 'mssql';
  cn.Database := 'TestDB';
  cn.Host := 'localhost'
  cn.Port := 1433;
  cn.User := '';
  cn.Password := '';
{$endif}
  cn.Connect();
//select
  rs := TZQuery.Create(nil);
  rs.Connection := cn;
  rs.SQL.Add('SELECT * FROM testTabelle');
  rs.Open;
  s := '';
//list fieldnames
  for i:=0 to rs.Fields.count-1 do
    s := s + IIF(i>0, ';') + rs.Fields[i].Name;
  s := s + #13#10;
//list values
  while not rs.eof do
  begin
    for i:=0 to rs.Fields.count-1 do
      s := s + IIF(i>0, ';') + rs.Fields[i].AsString;
    s := s + #13#10;
    rs.Next;
  end;
  WriteLn(s);
//close recordset
  rs.Close;
```

```
  rs.Free;
//close connection;
  cn.Disconnect;
  cn.Free;
end.
```

Autor: Udo Schmal (#udo.schmal) , veröffentlicht: 30.11.2013, letzte Änderung: 06.09.2023

(/)