12.8.1 Wizard – Projekte

Die Klasse Wizard nutzt man vor allem, um einen Dialog zu realisieren, der aus mehreren Schritten besteht, den man u.U. abbrechen kann und der im finalen Schritt mit dem OK-Button eine bestimmte Aktion anschiebt, die Dialog-Daten liefert. Andererseits kann man die Komponente *Wizard* auch einsetzen, um durch eine Schritt-Folge zu navigieren – ohne dass ein Ergebnis zurückgeliefert wird, wie das zum Beispiel bei einem Hilfe-Assistenten der Fall wäre.

Bei der Entwicklung der vorgestellten Projekte – bei denen die beiden o.a. Einsatzfälle realisiert werden – haben sich die Vorüberlegungen in adäquater Weise bewährt, auf die im ersten Teil des Kapitels → 12.6.1 'TabStrip – Projekte' hingewiesen wurde.

12.8.1.1 Projekt 1

Das erste Projekt wird dadurch charakterisiert, dass ein *Wizard* für einen *Dialog* verwendet wird. Der Dialog wird von einem Datenbank-Programm aufgerufen und stellt diesem als Ergebnis eines erfolgreichen Dialogs eine *aktive* DB-Verbindung zur Verfügung oder keine DB-Verbindung, wenn der Dialog abgebrochen wurde oder fehlerhaft war.

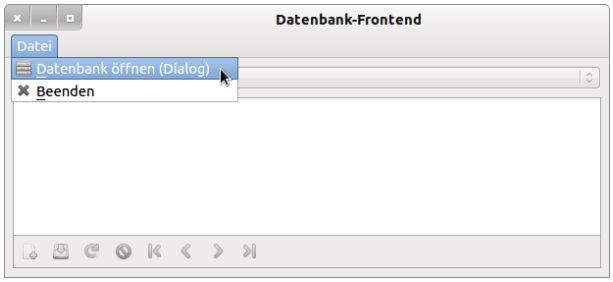

Abbildung 12.8.1.1.1: Aufruf eines Dialogs (Wizard) im Datenbank-Programm

Quelltext-Auszug aus dem Datenbank-Programm:

```
Private $hConnection As Connection

Public Sub Form_Open()
  FMain.Center
End ' Form_Open()

Public Sub mnuOpen_Click()
  Dim hNewConnection As Connection

  Try $hConnection.Close()
  hNewConnection = FMakeConnection.RunDialog()
  If Not hNewConnection Then Return
  $hConnection = hNewConnection
  Refresh()
End ' mnuOpen_Click()
```

In der Klasse *FMakeConnection.class* befindet sich in der Funktion *RunDialog()* der komplette Dialog.

Quelltext Dialog:

```
[1]  ' Gambas class file
[2]
[3]  Private $hDBConnection As New Connection
[4]  Private Const ACTIONTEXT As String = "Datenbank öffnen"
[5]
[6]  Public Function RunDialog() As Connection
[7]    Me.ShowModal()
[8]    Return $hDBConnection
[9]  End ' Run() As Connection
[10]
[11] Public Sub Form_Open()
[12]
[13]   FMakeConnection.Text = "Dialog - Datenbank-Verbindung"
[14]
[15]   wizConnect.Count = 3
[16]   wizConnect.Animated = True
[17]   wizConnect.Margin = True
[18]   wizConnect.ShowTitle = True
```

```
[19]   wizConnect.ShowButton = True
[20]
[21]   wizConnect.ActionPicture = Picture["icon:/16/view-detail"]
[22]   wizConnect.ActionText = ACTIONTEXT
[23]
[24]   wizConnect.ShowIndex = True
[25]   wizConnect[0].Text = "Wählen Sie den Datenbank-Typ aus!" ' #1
[26]   wizConnect[1].Text = "Wählen Sie eine SQLite3-Datenbank aus!" ' #2
[27]   wizConnect[2].Text = "Zugangsdaten eintragen und mit '" & ACTIONTEXT & "' zum Server verbinden!"
[28] ' Start-Verzeichnis im FileChooser (Schritt #2 - SQLite3)
[29]   fchSqlite.Root = Application.Path &/ "DBSQLite3"
[30]
[31] End ' Form_Open()
[32]
[33] Public Sub wizConnect_BeforeChange()
[34] ' Wenn SQLite3 als DBMS ausgewählt wurde, dann zeige den Schritt #2
[35] ' und blende Schritt #3 aus. Für MySQL und PostgreSQL entgegengesetzt.
[36]
[37]   If Not wizConnect[wizConnect.Index].Enabled Then Return
[38]   If radSqlite3.Value = True Then
[39]      wizConnect[1].Enabled = True  ' Schritt #2 anzeigen
[40]      wizConnect[2].Enabled = False ' Schritt #3 ausblenden
[41]      wizConnect.ActionText = "SQLite3-Datenbank öffnen"
[42]   Else
[43]      wizConnect[2].Enabled = True  ' Schritt #3 anzeigen - logisch ist das Schritt 2
[44]      wizConnect[1].Enabled = False ' Schritt #2 ausblenden
[45]    ' Port für PostgreSQL (5432) oder MySQL (3306) festlegen im Schritt #3
[46]      spbPort.Value = IIf(radPostgreSql.Value, 5432, 3306)
[47]      wizConnect.ActionText = ACTIONTEXT
[48]   Endif
[49] End ' wizConnect_BeforeChange()
[50]
[51] Public Sub wizConnect_Close()
[52]
[53]   If radSqlite3.Value Then
[54]      If Not fchSqlite.SelectedPath Then
[55]         Message.Error("Es wurde keine Datenbank-Datei ausgewählt.")
[56]         Return
[57]      Endif
[58]      $hDBConnection.Type = "sqlite3"
[59]      $hDBConnection.Name = fchSqlite.SelectedPath
[60]   Else
[61]      $hDBConnection.Type = IIf(radPostgreSql.Value, "postgresql", "mysql")
[62]      $hDBConnection.Host = txtHost.Text
[63]      $hDBConnection.Port = spbPort.Value
[64]      $hDBConnection.Login = txtUser.Text
[65]      $hDBConnection.Password = txtPassword.Text
[66]      $hDBConnection.Name = txtDatabase.Text
[67]   Endif
[68]   $hDBConnection.Open()
[69]   Me.Close()
[70]   Catch
[71]     Message.Error(Error.Text)
[72] End ' wizConnect_Close()
[73]
[74] Public Sub wizConnect_Cancel()
[75]   Me.Close()
[76] End ' wizConnect_Cancel()
[77]
[78] Public Sub wizConnect_Change()
[79] ' Print "AKTUELLER INDEX = "; wizConnect.Index
[80] End ' wizConnect_Change()
[81]
[82] Public Sub Form_Close()
[83]   If Not $hDBConnection Or If Not $hDBConnection.Opened Then $hDBConnection = Null
[84] End ' Form_Close()
```

Kommentar:

- In Abhängigkeit vom gewählten DBMS trägt das Ereignis *wizConnect_BeforeChange()* in den Zeilen 33 bis 49 die Hauptlast des Dialogs.
- Wenn der Dialog geschlossen wird, dann werden in den Zeilen 53 bis 67 die notwendigen Parameter für das Öffnen einer DB-Verbindung ermittelt und danach eine DB-Verbindung zur gewählten Datenbank hergestellt (→ Zeilen 68 und 69).
- Anschließend werden die Datenbank-Daten der ausgewählten DB-Tabelle in einem Datenbank-Browser im DB-Programm angezeigt.

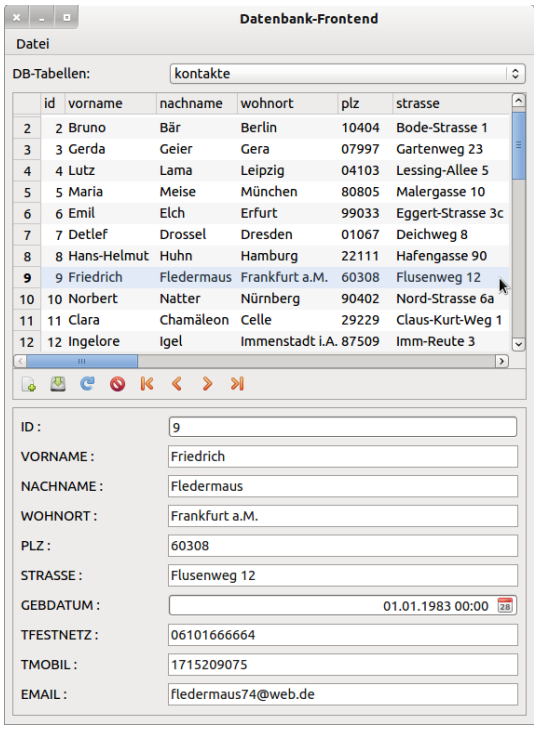

Abbildung 12.8.1.1.2: Anzeige der DB-Daten für die Tabelle 'kontakte' in der ausgewählten Datenbank

Umsetzung:

Abbildung 12.8.1.1.3: Schritt 1 – Auswahl DBMS

Im ersten Schritt des Wizards kann ein Datenbank-Management-System (DBMS) ausgewählt werden.

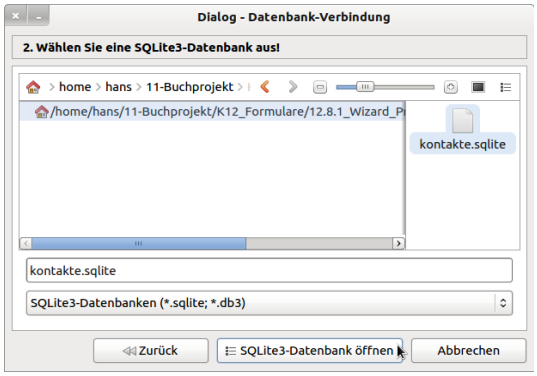

Abbildung 12.8.1.1.4: Schritt 2 – Auswahl SQLite3-Datenbank

Im 2. Schritt wird zum Beispiel eine SQLite3-Datenbank ausgewählt.

Beachten Sie: Im Projekt 1 wird Ihnen eine SQLite3-Datenbank im Projekt-Ordner für eigene Erprobungen zur Verfügung gestellt. Über das Menü des DB-Programms können Sie den Dialog jederzeit erneut starten.

12.8.1.2 Projekt 2

Das zweite Projekt setzt einen Wizard ein, um durch mehrere zusammenhängende Schritte zu navigieren. Als Beispiel dient die Anzeige eines Rezeptes in 4 Schritten. Da keine Aktion im letzten Schritt ausgelöst werden muss und die Anzeige jederzeit durch das Programm-Ende abgebrochen werden kann, wird auf die Anzeige der Button im unteren Panel des Wizards verzichtet. Setzen Sie deshalb die Eigenschaft *Wizard.ShowButton* bereits in der IDE auf *False*.

Um die Navigation zwischen den einzelnen Schritten müssen Sie sich jetzt aber selbst kümmern. Das gelingt durch den Einsatz der beiden Methoden *Wizard.MoveNext* und *Wizard.MovePrevious* ohne Probleme:

```
Public Sub btnNext_Click()
  If wizRezept.Index < wizRezept.Count Then
     wizRezept.MoveNext
     btnPrevious.Enabled = True
  Endif
  If wizRezept.Index = wizRezept.Count - 1 Then btnNext.Enabled = False
End ' btnNext_Click()
```

```
Public Sub btnPrevious_Click()
  If wizRezept.Index < wizRezept.Count Then
     wizRezept.MovePrevious
     btnNext.Enabled = True
  Endif
  If wizRezept.Index = 0 Then btnPrevious.Enabled = False
End ' btnPrevious_Click()
```

In der folgenden Abbildung sehen Sie den Inhalt des 2. Schrittes und die beiden Button für die Navigation sowie den Ende-Button:

Abbildung 12.8.1.2.1: Anzeige Schritt 2

12.8.1.3 Projekt 3

Für das Projekt 3 – das in der Anlage dem ersten Projekt gleicht – wird Ihnen zur Erprobung nur das Projekt-Archiv zur Verfügung gestellt. Das Projekt realisiert die Installation von Gambas in der aktuellen Version über SVN. Der Wizard verfügt nur über 2 Schritte:

1. Schritt Kopie des SVN-Repositoriums (1K) (Standardpfad oder selbst gewählter Pfad) oder Update des SVN-Repositoriums (1U).
2. Schritt Installation von Gambas.

- Zuerst wird geprüft, ob auf dem System das Programm *Subversion* installiert ist. Ist das nicht der Fall, dann wird das Programm mit einer Fehlermeldung beendet.
- Sonst wird beim *allerersten* Programmstart der erste Schritt 1K angezeigt und Sie können den Installationspfad festlegen oder den Standard-Pfad wählen.

- Bei jedem weiteren Programmstart wird der Schritt 1U angezeigt und ein Update angeboten, wenn die Version auf dem Server höher ist als die Version der lokalen Kopie des SVN-Repositoriums. Das Verzeichnis für das lokale SVN-Repositorium wird beim Programmende gespeichert und zum Programmstart eingelesen (Klasse Settings).

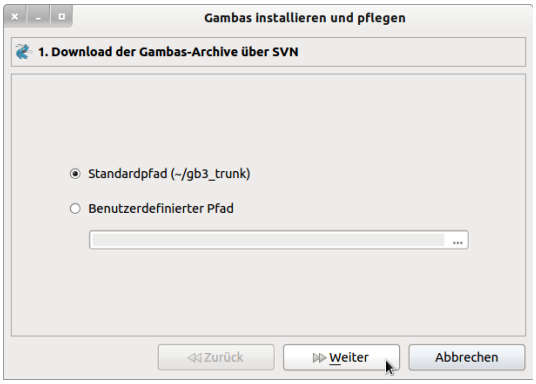

Abbildung 12.8.1.3.1: Anzeige Schritt 1K

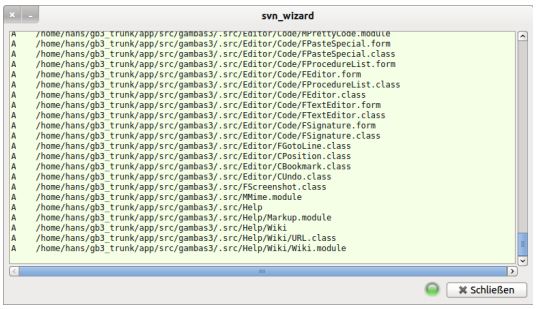

Abbildung 12.8.1.3.2: Lokale Kopie des SVN-Repositoriums

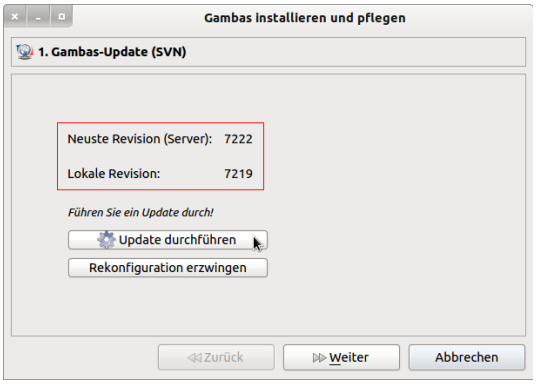

Abbildung 12.8.1.3.3: Anzeige Schritt 1U

Abbildung 12.8.1.3.4: Update des lokalen SVN-Repositoriums

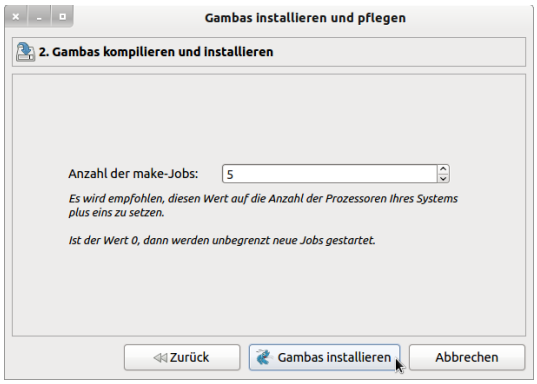

Abbildung 12.8.1.3.5: Schritt 2 – Installation von Gambas

Im Schritt 2 des Wizard können Sie noch die Anzahl der Jobs festlegen und dann die Installation von Gambas starten.

Wenn beim Programm-Start diese Oberfläche angezeigt wird, dann können Sie entweder eine Rekonfiguration der Quellen erzwingen oder das Programm abbrechen oder die angezeigte lokale Version von Gambas nach 'Weiter' installieren – sofern das noch nicht geschehen ist:

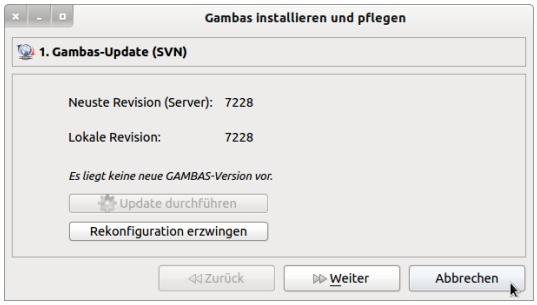

Abbildung 12.8.1.3.6: Nichts Neues zu entdecken ...