21.3.5 Projekt – Konsolen-Client 'mysql'

Auch in diesem Projekt werden die Themen 'Prozess-Steuerung' und 'Prozess-Daten' im Zusammenhang mit dem Einsatz der Instruktionen SHELL und EXEC aufgegriffen. Es wird die *Basis* für eine GUI für den interaktiven Konsolen-Client 'mysql' vorgestellt.

Abbildung 21.3.5.1: Basis-GUI für das Konsolen-Programm 'mysql'

Sie können das Projekt in vielfältiger Weise zu Ihrer GUI für den Konsolen-Client 'mysql' ausbauen, die sich an den zu bearbeitenden Aufgaben sowie an den eigenen Vorstellungen orientiert. Folgende Erweiterungen wären denkbar:

- Anmelde-Formular (DB-Server, DB-User, DB-User-Passwort)
- Auswahl aller relevanten Datenbanken und der zu dieser Datenbank gehörenden Tabellen
- Ausgabe der angefragten Daten in unterschiedlichen Formaten → XML, HTML
- BackUp-Funktionen (Konsolen-Programm '*mysqldump*')
- DB-Reports
- Daten-Export und Daten-Import (csv-Dateien)

In diesem Projekt können Sie nicht nur den Konsolen-Client 'mysql' einsetzen, sondern auch die Konsolen-Clients 'sqlite3' für SQLite3 oder 'psql' für Postgres-Datenbanken, sofern Sie diese Clients installiert haben.

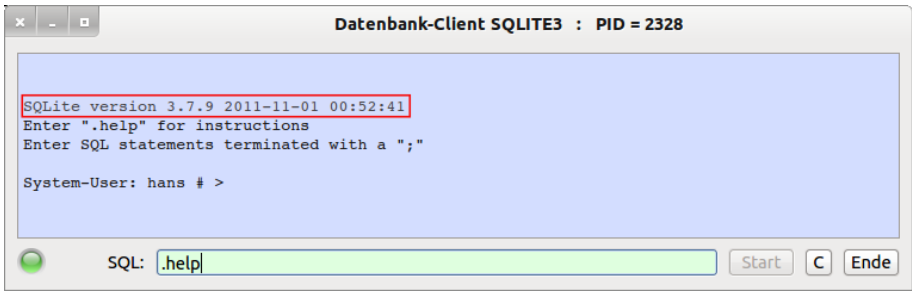

Abbildung 21.3.5.2: Basis-GUI für das Konsolen-Programm 'sqlite3'

In den folgenden Abschnitten wird der Konsolen-Client 'mysql' eingesetzt:

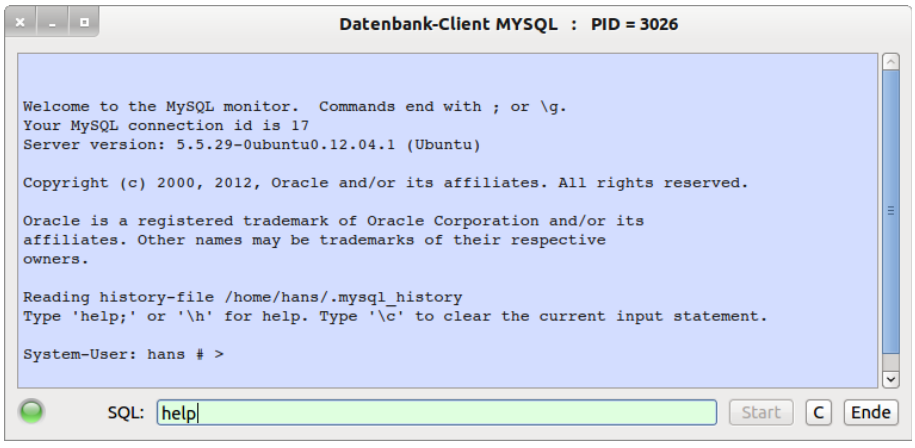

Abbildung 21.3.5.3: GUI nach dem Start des Clients 'mysql'

Nach dem Start des externen Programms 'mysql' sehen Sie den Namen des externen DB-Clients und die PID des gestarteten Prozesses sowie den Begrüßungstext des Datenbank-Clients in der Ausgabe-Komponente. Danach arbeiten Sie wie in einer normalen Konsole – mit dem Unterschied, dass Sie jetzt alle Ausgaben mit dem Gambas-Programm unter den o.a. Aspekten von Programm-Erweiterungen bearbeiten können:

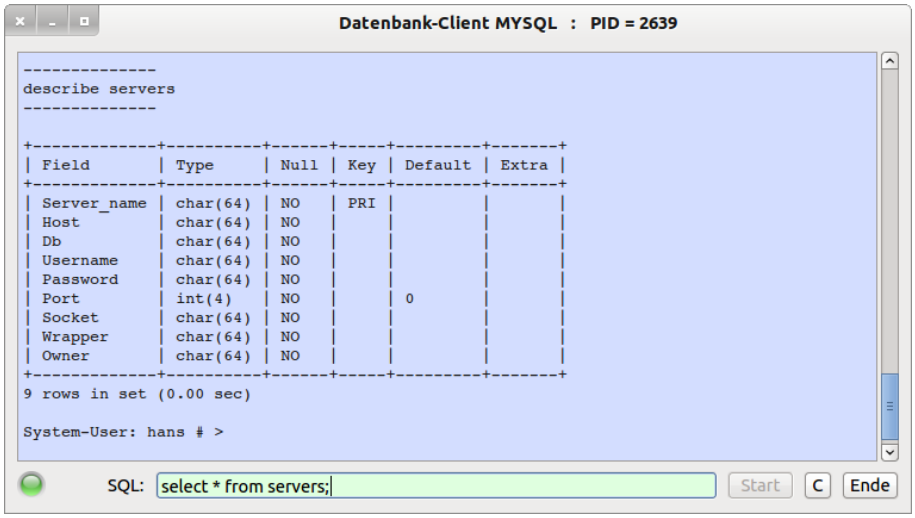

Abbildung 21.3.5.4: Anzeige einer Abfrage – Client 'mysql'

Einige Besonderheiten kennzeichnen das vorgestellte Projekt:

- Das Projekt verwendet eine neue Komponente. Die neue Komponente heißt *HistoryBox* und wird in der Klasse HistoryBox zur Verfügung gestellt.
- Das Projekt nutzt die neue Komponente, denn diese verfügt als *Erweiterung* der Komponente TextBox über eine sogenannte History.
- Sie können in der HistoryBox die Pfeiltasten ∧∨ benutzen, um wie in einer Konsole nach bereits eingegebenen Eingaben zu suchen.
- Die Klassen HistoryBox, History und _HistoryOptions wurden von Tobias Boege entwickelt. Sie wurden in der Phase der Erprobung des vorgestellten Projektes getestet.
- 

  Wie bei allen Komponenten üblich, wurde auch der HistoryBox von den Autoren ein Icon spendiert, das Sie in der IDE im Fenster Eigenschaften unter 'Form' sehen. Das Icon muss im Projektverzeichnis in einem unsichtbaren Ordner .hidden/control als Bild vom Typ 'png' eingefügt werden, damit es in der IDE im virtuellen Ordner 'Projekt' zu sehen ist.
- Alle Komponenten – außer der TextArea für die Ausgabe der Daten – wurden in einem Container *HBox* platziert, die auch ein nicht sichtbares Panel zwischen der Komponente PictureBox und den anderen Komponenten enthält. Die Weite des Panels passt sich *dynamisch* an die Fenstergröße an und hält so alle Komponenten am rechten Rand.
- Die Ausnahme bildet die PictureBox, die *permanent* am linken Rand positioniert bleibt. Als Bilder werden je nach Prozesszustand unterschiedlich farbige Bilder eingefügt, die den Eindruck einer farbigen LED erzeugen. Die LED signalisiert den Zustand des gestarteten Prozesses:

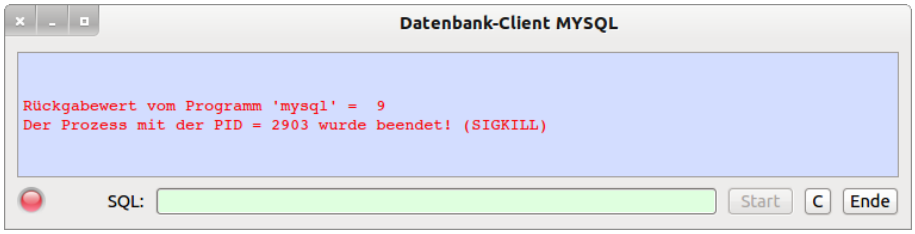

Abbildung 21.3.5.5: Der DB-Client wurde beendet

Es gibt einige Programme – wie die vorgestellten DB-Clients – die ihre Fehlerausgaben permanent auf die Standard-Ausgabe umlenken. Somit entfällt bei diesen Programmen diese Prozedur:

```
Public Sub myProcess_Error(sFehler As String)
…
End ' myProcess_Error(..)
```

Der Quelltext wird vollständig angegeben:

```
' Gambas class file

Private $hProcess As Process

' Private sDBClientName As String = "sqlite3"
' Private sStartParameter As String = ""
Private sDBClientName As String = "mysql"
Private sStartParameter As String = "-f -n -vvv -u root -pYourMySQLPassword4SQLRoot"

Public Sub Form_Open()
  FMain.Center
  FMain.Resizable = True
  FMain.Arrangement = Arrange.Vertical
  txaOutput.Expand = True
  txaOutput.Clear
  HBoxSpacing.Height = 8
  HBox.Spacing = True
  HBox.Height = 24
  pDynamicPanel.Expand = True

  txbHistoryEingabe.Foreground = Color.Gray
  txbHistoryEingabe.Alignment = Align.Center
  txbHistoryEingabe.Text = "*** SQL-Anweisung eingeben und mit ENTER-Taste aktivieren ***"
  txbHistoryEingabe.ReadOnly = True
  txbHistoryEingabe.History = New History(50)

  SetLEDColor("orange")

End ' Form_Open()

Public Sub btnProcessStart_Click()
 ProcessStart()
 FMain.Text = "Datenbank-Client " & Upper(sDBClientName) & "   :   PID = " &  $hProcess.Id
 txbHistoryEingabe.Alignment = Align.Normal
End ' btnProcessStart

Public Sub btnTextAreaClear_Click()
  txaOutput.Clear
  txbHistoryEingabe.SetFocus
End ' btnTextAreaClear

Public Sub btnClose_Click()
  FMain.Close
End ' btnClose_Click()

'*****************************************************************************************

Public Sub ProcessStart()
   Dim sCommand As String

  txbHistoryEingabe.Foreground = Color.Black
  txbHistoryEingabe.Clear
  txaOutput.Clear

  sCommand = sDBClientName & Chr(32) & sStartParameter
  $hProcess = Shell sCommand For Input Output As "myProcess"

  txaOutput.Insert(gb.NewLine)
  txbHistoryEingabe.ReadOnly = False
  txbHistoryEingabe.SetFocus
  btnProcessStart.Enabled = False
  SetLEDColor("green")
```

```
End ' ProcessStart()

Public Sub myProcess_Read()
  Dim sPuffer As String

  txaOutput.Insert(gb.NewLine)
  sPuffer = ""
  Read #$hProcess, sPuffer, Lof($hProcess)

  Select sDBClientName
  Case "mysql"
    sPuffer = Replace(sPuffer, "mysql>", "System-User: " & User.Name & " # >" & gb.NewLine)
  Case "sqlite3"
    sPuffer = Replace(sPuffer, "sqlite>", gb.NewLine & "System-User: " & User.Name & \
                      " # >" & gb.NewLine)
  End Select
  txaOutput.Insert(sPuffer & gb.NewLine)

End ' hProcess_Read()

' Public Sub myProcess_Error(sFehler As String)
'   Diese Prozedur kann entfallen, da Fehler-Ausgaben sowohl beim MySQL-Client
'   als auch beim SQLite3-Client intern auf die Standard-Ausgabe umgelenkt werden!
' End ' myProcess_Error(..)

Public Sub myProcess_Kill()
  txaOutput.Insert(gb.NewLine)
  txaOutput.Foreground = Color.Red
  txaOutput.Insert("Rückgabewert vom Programm '" & sDBClientName & "' =  " \
                   & $hProcess.Value & gb.NewLine)
  Select Case $hProcess.State
  Case 0
   txaOutput.Insert("Prozess (PID = " & $hProcess.Id & ") wurde normal beendet." & gb.NewLine)
  Case 1
   txaOutput.Insert("Prozess (PID = " & $hProcess.Id & ") wurde gestoppt!" & gb.NewLine)
  Case 2
   txaOutput.Insert("Prozess (PID = " & $hProcess.Id & ") wurde beendet!" & gb.NewLine)
  End Select ' $hProcess.State
  SetLEDColor("red")
End ' myProcess_Kill()

Public Sub WriteToMyProcess(sInput As String)
   If $hProcess Then
      If $hProcess.State = $hProcess.Running Then
         Print #$hProcess, sInput
      Endif ' Running ?
   Endif ' Process started ?
End ' WriteToMyProcess(sInput As String)

Public Sub txbHistoryEingabe_Activate()
  If txbHistoryEingabe.Text = "" Then Return
  txbHistoryEingabe.Text = Trim(txbHistoryEingabe.Text)
  WriteToMyProcess(txbHistoryEingabe.Text)
' txbHistoryEingabe.Clear   ---> Das erledigt die History-Box selbst
End ' txbHistoryEingabe_Activate()

Public Sub txaOutput_Change()
  txaOutput.Pos = Len(txaOutput.Text) ' ---> Sprung in die letzte Zeile
End ' txaOutput_Change()

Public Sub SetLEDColor(sLEDColor As String)
   PictureBox1.Picture = Picture["LED/led_" & sLEDColor & ".svg"]
End '  SetLEDColor(..)

Public Sub Form_Close()
  If $hProcess Then
     txaOutput.SetFocus
     txaOutput.Clear
     txaOutput.Insert(gb.NewLine)
     $hProcess.Kill()
     Wait 2
  Endif ' $hProcess ?
End ' Form_Close()
```