16.14.1 Projekt – Suchen und ersetzen von Text in einer TextArea

Das Suchen und Ersetzen von Text gelingt auch in einer TextArea mit regulären Ausdrücken sehr gut. Informationen dazu finden Sie im Kapitel 19.6.6 Suchen und Ersetzen von Zeichenketten in einem Text.

In diesem Kapitel wurde der Schwerpunkt jedoch darauf gelegt, im vorgestellten Projekt beim Suchen und ersetzen von Text vorwiegend mit Methoden und Eigenschaften der Klasse TextArea zu arbeiten.

- Den Suchtext können Sie in der TextArea markieren und mit einem Klick auf die drei Pünktchen in der gelben ButtonBox in die Suchtext-Zeile übernehmen. Sie können den Suchtext auch direkt in die Suchtext-Zeile eingeben. Das Suchen von Text erfolgt schrittweise, wobei jede Fundstelle *temporär* markiert wird → Abbildung 16.14.1.1. Wenn keine oder keine weitere Fundstelle existiert, dann werden Sie darüber mit einer MessageBox informiert.

- Für das Ersetzen von Text sind die Angabe des Suchtextes und des Ersetzungstextes in den entsprechenden TextBoxen notwendig. Es werden alle Ersetzungen vorgenommen und abschließend nur die erste Ersetzung markiert. Kann der Suchtext im Text nicht gefunden werden, so werden Sie informiert, dass nichts ersetzt wurde.

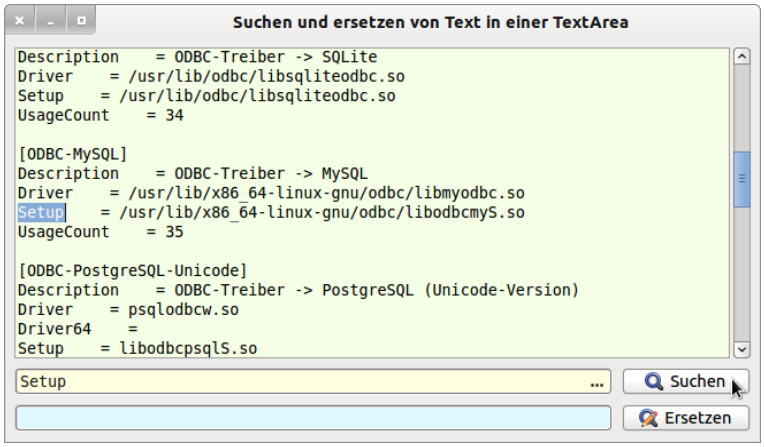

Abbildung 16.14.1.1: Suchen von Text in einer TextArea

16.14.1.1 Projekt – Quelltext

Der Quelltext wird vollständig angegeben und anschließend kommentiert:

```
[1]  ' Gambas class file
[2]
[3]  Private $iPos As Integer
[4]
[5]  Public Sub Form_Open()
[6]    Dim sMimeTypeCharSet, sCharSet, sMime As String
[7]    Dim sFilePath As String
[8]
[9]    FMain.Center
[10]   btnNextSearch.Tooltip = ("Suchen")
[11]   btnReplace.Tooltip = ("Ersetzen")
[12]
[13]   txaArea.Background = &HF5FFE6& ' Text-Hintergrund-Farbe: hellgrün
[14]   txaArea.Foreground = Color.RGB(0, 0, 0) ' Schrift-Farbe: schwarz
[15]   txaArea.Font = Font["Monospace, 10"] ' dicktengleiche Schrift
[16] ' Print txaArea.Font.ToString()
[17]
[18]   ' Dialog.Title = "Wählen Sie eine Text-Datei aus!"
[19]   ' Dialog.Filter = ["*.ini", "INI-Dateien", "*.txt", "Text-Dateien", "*", "Alle Dateien"]
[20]   ' Dialog.Path = "/etc/odbcinst.ini"
[21]   ' If Dialog.OpenFile() Then Return
[22]   ' sFilePath = Dialog.Path
[23]
[24]   sFilePath = Application.Path &/ "odbcinst.ini" ' Beispiel-Textdatei im Projekt-Verzeichnis
[25]
[26]   Exec ["file", "-bi", sFilePath] To sMimeTypeCharSet ' Ermittlung MimeTyp und Zeichensatz
[27]   Wait
```

```
[28] ' Print sMimeTypeCharSet
[29] ' sMime = Split(sMimeTypeCharSet, ";")[0]
[30] ' Print "Mime-Typ = "; Trim(sMime)
[31]   sCharSet = Trim(Split(Split(sMimeTypeCharSet, ";")[1], "=")[1])
[32] ' Print "CharSet  = "; Trim(sCharSet)
[33]
[34]   If sCharSet <> "utf-8" Then
[35]      Try txaArea.Text = Conv(File.Load(sFilePath), sCharSet, "UTF-8")
[36]   Else
[37]      txaArea.Text = File.Load(sFilePath)
[38]   Endif
[39]
[40]   txaArea.Pos = 0
[41]
[42] End ' Form_Open()
[43]
[44] Public Sub txbSearch_Click()
[45]   txbSearch.Text = txaArea.Selection.Text ' Markierter Text = Suchtext
[46] End
[47]
[48] Public Sub txbSearch_Change()
[49]   $iPos = -1 ' Ab Text-Anfang suchen
[50] End
[51]
[52] Public Sub btnReplace_Click()
[53]   ReplaceText()
[54] End
[55]
[56] Public Sub btnNextSearch_Click()
[57]   If Not txbSearch.Text Then
[58]      Message.Warning(("Es fehlt der Suchtext!"))
[59]      txbSearch.SetFocus
[60]      Return
[61]   Endif
[62]   txaArea.Unselect
[63]   SearchText()
[64] End
[65]
[66] '-------------------------------------------------------------------------------------
[67]
[68] Private Sub SearchText()
[69]   Dim iPos As Integer
[70]
[71]   iPos = String.InStr(txaArea.Text, txbSearch.Text, $iPos + 2)
[72]   $iPos = iPos - 1
[73]   If Not iPos Then
[74]     Message(("Suchtext nicht gefunden!"))
[75]     txaArea.SetFocus
[76]     txaArea.Pos = 0
[77]     Return
[78]   Endif
[79]   txaArea.Select($iPos, String.Len(txbSearch.Text)) ' Markierung einer Fundstelle im Text
[80]
[81] End ' SearchText(...)
[82]
[83] Private Sub ReplaceText()
[84]   Dim iPos As Integer
[85]   Dim bNotReplaced As Boolean = True
[86]
[87]   If Not txbSearch.Text Then
[88]      Message.Warning(("Es fehlt der Suchtext!"))
[89]      Return
[90]   Endif
[91]   If Not txbReplace.Text Then
[92]      Message.Warning(("Es fehlt der Ersetzungstext!"))
[93]      txbReplace.SetFocus
[94]      Return
[95]   Endif
[96]   While String.InStr(txaArea.Text, txbSearch.Text)
[97]     txaArea.Text = Replace(txaArea.Text, txbSearch.Text, txbReplace.Text)
[98]     bNotReplaced = Not bNotReplaced
[99]   Wend
[100]
[101]    iPos = String.InStr(txaArea.Text, txbReplace.Text, 0)
[102]    If Not iPos And bNotReplaced Then
[103]       Message(("Es wurde NICHTS ersetzt!"))
[104]       Return
[105]    Endif
[106]
[107]    txaArea.Select(iPos - 1, String.Len(txbReplace.Text))
[108]
[109]  End ' ReplaceText()
```

Kommentar:

- Die Hintergrund-Farbe, die Schrift-Farbe und der Schrift (Font) werden in den Zeilen 13-15 für die TextArea global festgelegt.
- Mit den den Zeilen 18-22 können Sie einen Datei-Öffnen-Dialog realisieren. Dazu müssten Sie die Zeilen 18-22 aktivieren und die Zeile 24 auskommentieren. Darauf wurde hier verzichtet, um mit einer speziellen Datei zu arbeiten, die im Projekt-Pfad liegt.
- Damit garantiert ein UTF-8-String in die TextArea eingefügt wird, erfolgt die Ermittlung des Zeichensatzes der zu importierenden (Text-)Datei in den Zeilen 26 und 31. Eine Konvertierung erfolgt nur dann, wenn das notwendig ist (Zeilen 34-38).
- In der Zeile 79 wird die temporäre Markierung einer Fundstelle erzeugt.
- Die Hauptlast im Projekt tragen die Prozedur *SearchText()* in den Zeilen 68-81 sowie die Prozedur *ReplaceText()* in den Zeilen 83-109.