12.2.4 Form – Dialog-Fenster

In diesem Kapitel wird in einem Projekt u.a. ein Dialog vorgestellt, der den Beschreibungen für nutzerdefinierte Dialoge folgt, die im Kapitel → 12.4 ausführlich erläutert werden. In diesem Projekt wird für den Aufruf des Dialog-Fensters die spezielle _call(..)-Methode verwendet. Sie erlaubt es, ein Objekt syntaktisch wie eine Funktion zu benutzen. Mit dem Aufruf der _call(..)-Methode kann das selbst entworfene Dialog-Fenster sich selbst modal anzeigen! Gut zu wissen: In den optionalen Argumenten der _call(..)-Methode können Sie Daten an das Dialog-Fenster übergeben und im Rückgabewert der Methode als Funktionswert Daten aus dem Dialog auslesen, speichern und weiter verarbeiten.

Das Besondere am Projekt ist der passwort-geschützte Zugang zu einem (Haupt-)Programm über ein Dialog-Fenster zur Abfrage eines Passwortes. Die Datei *login_password* enthält das benötigte Passwort, das verschlüsselt in der Datei abgespeichert wurde. Das Generieren der Passwort-Datei erfolgt in einem separaten Projekt:

Abbildung 12.2.4.1: Passwort erzeugen

```
sHash = Digest["sha512"](txbPassword.Text)
File.Save(Application.Path &/ "login_password", sHash)
```

Login-Fenster als Dialog

Bereits beim Öffnen des Haupt-Programms wird das Passwort aus der Passwort-Datei ausgelesen und der Variablen *hHash* zugewiesen. Noch bevor das Hauptprogramm angezeigt wird, wird das Login-Fenster als Dialog geöffnet und der Benutzer aufgefordert, das Passwort einzugeben:

```
' Vor dem Haupt-Fenster wird das Login-Fenster (Dialog) *modal* geöffnet:
  fLoginDialog = New FLogin
  aDialogData = fLoginDialog("Geben Sie das Login-Passwort ein:")
```

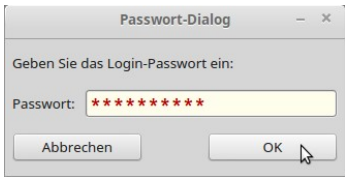

Abbildung 12.2.4.2: Passwort erzeugen

Wird das Dialog-Fenster geschlossen, kann ermittelt werden, ob der Benutzer den Dialog abgebrochen hat oder ein falsches Passwort eingegeben wurde:

```
' Dialog auswerten
  If Not aDialogData Then ' Dialog-Abbruch?
     Me.Close() ' Haupt-Fenster *ohne* Kommentar schließen
     Return
  Endif
' Passwort-Check
  If Digest["sha512"](aDialogData["Password"]) <> sHash Then ' Fehler? Programm mit Kommentar beenden.
     Message.Title = "Passwort-Fehler ..."
     Message.Error("<center>Login fehlgeschlagen!<hr>Das Haupt-Programm wird beendet.</center>")
     Me.Close()
     Return
  Endif
```

Abbildung 12.2.4.3: Passwort erzeugen

In beiden Fällen wird das Haupt-Programm beendet und nur im Fall des fehlerhaften Passworts wird ein Kommentar ausgegeben.

Stimmt das hinterlegte verschlüsselte Passwort in der Passwort-Datei mit dem im Dialog-Fenster eingegebenen Passwort überein, wird das Haupt-Programm angezeigt:

Abbildung 12.2.4.4: Hauptprogramm-Fenster

Der Quelltext für den Login-Dialog wird vollständig angegeben:

```
' Gambas class file

Public Function _call(sMessage As String) As Collection
  lblMessage.Text = sMessage
  ' Kehrt zurück, wenn auf einen der Button geklickt wurde!
  ' Der Rückgabewert wird im Me.Close()-Aufruf angegeben und
  ' zeigt an, ob abgebrochen wurde, oder nicht.
  txbPassword.SetFocus()
  If Me.ShowModal() = 0 Then Return Null
  Return ["Name": User.Name, "Password": txbPassword.Text] ' Me.ShowModal() = 1

End ' Function _call(...)

Public Sub btnOK_Click()
  If Not txbPassword.Text Then
     Message.Title = "Passwort-Fehler ..."
     Message.Warning("Es wurde kein Passwort eingegeben.")
     txbPassword.SetFocus()
     Return
  Endif
  Me.Close(1)
End ' btnOK_Click()

Public Sub btnCancel_Click()
  ' 0 wird ebenfalls beim Schließen durch das Kreuz in der Fenster-Leiste zurückgegeben;
  ' das entspricht einem Abbruch des Dialogs durch den Nutzer.
  Message.Warning("Der Login wurde abgebrochen ....")
  Me.Close(0)
End

Public Sub txbPassword_Activate()
  btnOK_Click()
End ' txbPassword_Activate()
```

Im Download-Bereich finden Sie die beiden o.a. Projekte.