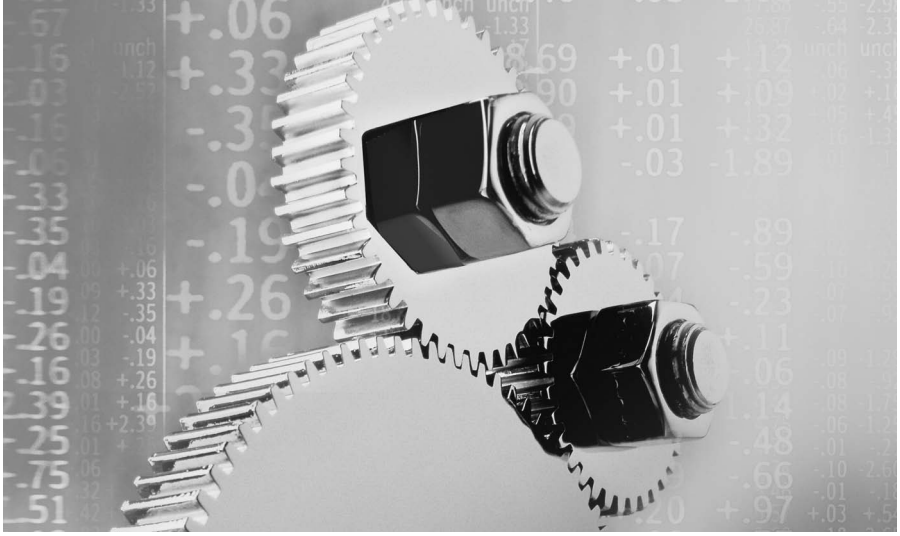

Automatisation
Performance & Benutzerfreundlichkeit
Entwickler-Know-how
Tools & Add-Ins
E-Mail-Support
Praxis pur
Anwendungen mit Excel entwickeln
Professionelle Excel-VBA-Programmierung für die Versionen 2000 bis 2010
BERND HELD
Markt+Technik
CD
KOMPENDIUM



Kapitel 3
So nützen Sie Access als Datenlieferant für Excel

Eine Stärke der Programmiersprache VBA (Visual Basic for Applications) ist die universelle Verwendbarkeit im ganzen Office-Paket. So können Sie Daten zwischen den einzelnen Anwendungen problemlos importieren sowie exportieren und dadurch die Stärke einer jeden Office-Anwendung optimal ausschöpfen. Im folgenden Kapitel erfolgen Zugriffe von Excel auf eine Access-Datenbank. Dabei werden Daten sowohl importiert als auch exportiert.

Die Beispieldateien ABFRAGE.XLS und ADRESSEN.MDB finden Sie auf der CD-ROM in Verzeichnis KAP03.

CD

3.1 So bereiten Sie die Programmierung vor

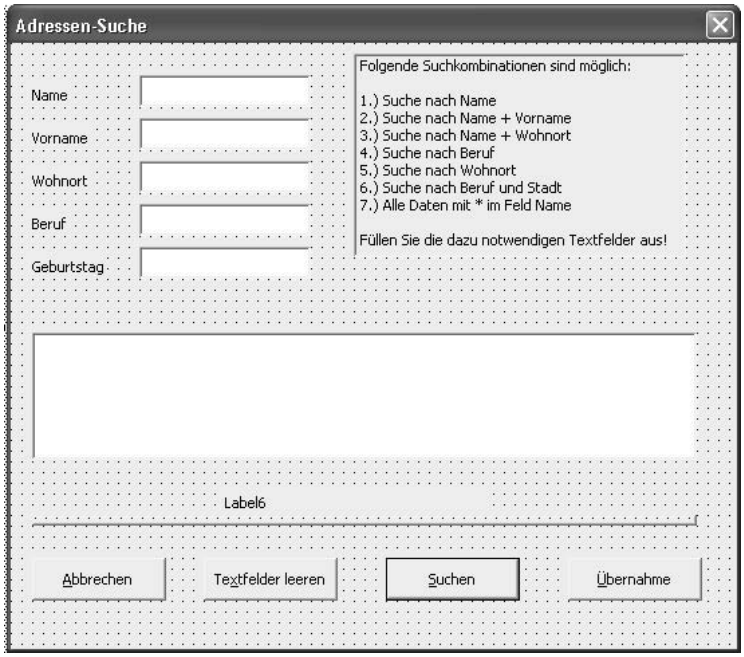
Um den Zugriff von Excel auf Access zu realisieren, wird eine zusätzliche Objektbibliothek benötigt, die in der Entwicklungsumgebung eingebunden werden muss. Dabei wechseln Sie in die Entwicklungsumgebung von Excel und wählen aus dem Menü EXTRAS den Befehl VERWEISE. Im Listenfeld VERFÜGBARE VERWEISE wird die Bibliothek MICROSOFT ACTIVEX DATA OBJECTS 2.5 ausgewählt und mit OK bestätigt. Damit haben Sie Zugriff auf die Datenzugriffsmethode ADO (*ActiveX Data Object*). Unter anderem zeichnet sich diese Methode durch eine sehr hohe Geschwindigkeit, eine benutzerfreundliche Bedienung sowie einen geringen Verbrauch an Arbeitsspeicher und Festplattenspeicher aus. Über den Einsatz von ADO können Sie auf Access-Tabellen zugreifen, um diese auszulesen bzw. um diese um neue Datensätze zu erweitern.

3.2 So lauten die Anforderungen

Das Ziel dieses Kapitels ist es, mithilfe von Excel auf eine Access-Datentabelle, die Adressdaten enthält, zuzugreifen. Des Weiteren sollen bestimmte Datensätze wie beispielsweise die drei ältesten Personen sowie der »Junior« aus dieser Datenbank ermittelt werden.

Zu Beginn wurde in Excel eine UserForm für die Datensuche erstellt, die in Abbildung 3.1 einsehbar ist.

Abbildung 3.1: Die UserForm für die Datensuche



Diese Datenmaske enthält vier Texteingabefelder, mit denen diverse Suchen durchgeführt werden können. Möglich sind folgende Suchoptionen:

- Die Eingabe des Stern-Symbols im Feld NAME listet im Listenfeld alle Datensätze auf.
- **Suche nach dem Namen:** Dabei wird lediglich das Feld NAME ausgefüllt.
- **Suche nach Name und Vorname:** Beide Felder müssen hierzu ausgefüllt werden.
- **Suche über Name und Wohnort:** Hier erfolgt eine Eingabe von Suchbegriffen in den Feldern NAME und WOHNORT.
- **Suche nach dem Beruf:** Hier darf nur im Feld BERUF ein Suchbegriff eingegeben werden.
- **Suche nach Beruf und Wohnort:** Suchen Sie beispielsweise einen Metzger in Hamburg, und füllen Sie dazu beide Felder aus.
- **Suche nach Wohnort:** Hier werden alle Personen aus einer Stadt gesucht.

3.2.1 So führen Sie Standardaufgaben beim Starten der UserForm durch

Im Listenfeld werden alle gefundenen Personen aufgelistet. Dabei wird das Listenfeld von Beginn an mehrspaltig eingerichtet, und die Spaltenbreiten werden definiert. Diese Aufgabe können Sie gleich beim Öffnen der UserForm durchführen lassen. Dazu setzen Sie das UserForm-Ereignis `Initialize` ein, wie in Listing 3.1 dargestellt.

Listing 3.1: Bestimmte Aufgaben beim Starten der UserForm ausführen

```
Private Sub UserForm_Initialize()
'UserForm initialisieren
  With UserForm1
    .ListBox1.ColumnCount = 5
    .ListBox1.ColumnWidths = "80;80;60;70;70"
    .CommandButton1.Default = True
    .TextBox1.SetFocus
    .Label6.Visible = False
    .Label6.Caption = ""
    .Label6.ForeColor = RGB(0, 0, 255)
    .Label6.Font.Bold = True
  End With

End Sub
```

Über die Eigenschaft `ColumnCount` wird angegeben, wie viele Spalten im Listenfeld angelegt werden sollen. Mithilfe der Eigenschaft `ColumnWidth` werden dann die Spaltenbreiten festgelegt. Damit die Schaltfläche SUCHEN beim Drücken der Taste ↵ automatisch gedrückt wird, setzen Sie die Eigenschaft `Default` dieser Schaltfläche auf den Wert `True`.

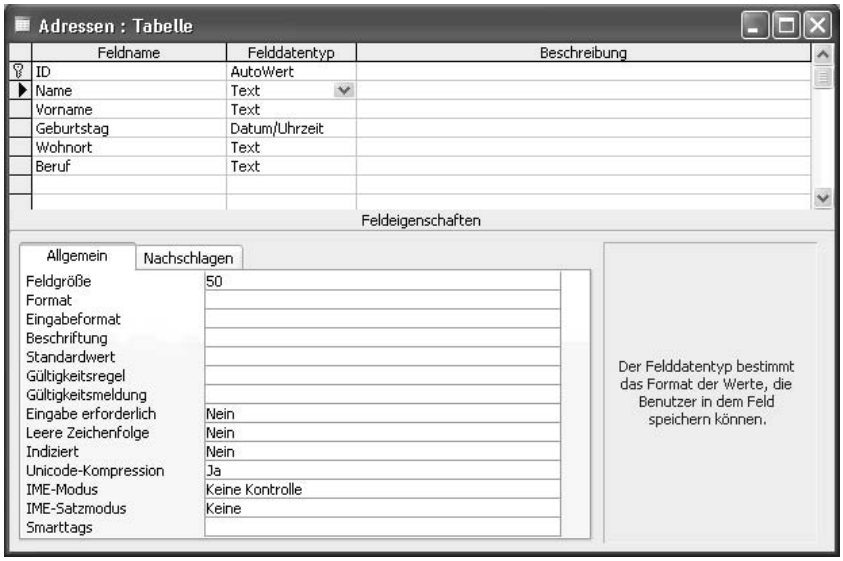
3.2.2 So definieren Sie Bezeichnungsfelder

Standardmäßig wird der Mauszeiger gleich beim Starten der UserForm in dem ersten Textfeld (NAME) positioniert. Über ein Bezeichnungsfeld wird später dargestellt, wie viele Datensätze gefunden wurden. Indem der Eigenschaft `Visible` der Wert `False` zugewiesen wird, bleibt das Feld zu Beginn verborgen. Eine Beschriftung des Bezeichnungsfeldes kann über die Eigenschaft `Caption` erfolgen. Die Farbe des Textes wird über die Eigenschaft `ForeColor` und die Funktion `RGB` definiert. Die Funktion `RGB` mischt sich die Farbe aus den drei Komponenten Rot, Grün und Blau zusammen. Mit der Eigenschaft `Bold` und dem Wert `True` wird eine Schrift mit dem Schriftschnitt FETT dargestellt.

3.2.3 So legen Sie die Access-Datenbank an

Bevor die eigentliche Programmierung beginnt, sollte eine Access-Datenbank (ADRESSEN.MDB) angelegt werden. Gestalten Sie die Access-Datentabelle nach dem Vorbild aus Abbildung 3.2 und speichern diese unter dem Namen ADRESSEN im gleichen Verzeichnis wie Ihre Excel-Arbeitsmappe.

Abbildung 3.2: Die Access-Tabelle zur Ablage der Daten



3.2.4 So führen Sie die Datenbanksuche durch

Die diversen Such-Funktionen werden über die Schaltfläche SUCHEN ausgeführt. Alternativ kann auch die Tastenkombination Alt + S hierfür hinterlegt werden. Über die Eigenschaft `Accelerator` lässt sich der Buchstabe hinterlegen, der auch Bestandteil des Schaltflächentextes sein sollte. In diesem Fall wird dieser dann automatisch unterstrichen. Markieren Sie die Schaltfläche und hinterlegen im Eigenschaften-Fenster unter `Accelerator` den Buchstaben, über den in Verbindung mit der Taste Alt die Schaltfläche per Shortcut ausgeführt werden soll.

Übernehmen Sie nun den Code aus Listung 3.2 direkt hinter die Schaltfläche SUCHEN.

Listing 3.2: Die unterschiedlichen Suchen-Funktionen

```
Private Sub CommandButton1_Click()
'Daten in Access-Datenbank suchen und übergeben
Dim AccApp As Object
Dim conn As New ADODB.Connection
Dim RST As ADODB.Recordset
Dim strBegriff As String
Dim strBegriff2 As String
Dim intZ As Integer
Dim BSchalter As Boolean

  intZ = 0
  BSchalter = False

  With UserForm1
    'Listenfeld löschen
    .ListBox1.Clear
```

```
'Alle Datensätze
If .TextBox1.Value = "*" Then
  strBegriff = ""
  strBegriff = "Name='" & strBegriff & "'"
  BSchalter = True
  GoTo weiter
End If

'Suche nach Beruf
If .TextBox4.Value <> "" And .TextBox1.Value = "" _
And .TextBox2.Value = "" And .TextBox3.Value = "" Then
  strBegriff = UserForm1.TextBox4.Value
  strBegriff = "Beruf='" & strBegriff & "'"
End If

'Suche nach Beruf und Stadt
If .TextBox4.Value <> "" And .TextBox1.Value = "" _
 And .TextBox2.Value = "" And .TextBox3.Value <> "" Then
  strBegriff = UserForm1.TextBox4.Value
  strBegriff2 = UserForm1.TextBox3.Value
  strBegriff = "Beruf='" & strBegriff & "' and Wohnort='" & _
   strBegriff2 & "'"
End If

'Suche nach Name
If .TextBox1.Value <> "" And .TextBox2.Value = "" _
 And .TextBox3.Value = "" And .TextBox4.Value = "" Then
  strBegriff = UserForm1.TextBox1.Value
  strBegriff = "Name='" & strBegriff & "'"
End If

'Suche nach Name und Vorname
If .TextBox1.Value <> "" And .TextBox2.Value <> "" _
 And .TextBox3.Value = "" And .TextBox4.Value = "" Then
  strBegriff = UserForm1.TextBox1.Value
  strBegriff2 = UserForm1.TextBox2.Value
  strBegriff = "Name='" & strBegriff & _
   "' and Vorname='" & strBegriff2 & "'"
End If

'Suche nach Stadt
If .TextBox4.Value = "" And .TextBox1.Value = "" _
 And .TextBox2.Value = "" And .TextBox3.Value <> "" Then
  strBegriff = UserForm1.TextBox3.Value
  strBegriff = "Wohnort='" & strBegriff & "'"
End If

'Suche nach Name und Wohnort
If .TextBox1.Value <> "" And .TextBox2.Value = "" _
 And .TextBox3.Value <> "" And .TextBox4.Value = "" Then
  strBegriff = UserForm1.TextBox1.Value
  strBegriff2 = UserForm1.TextBox3.Value
  strBegriff = "Name='" & strBegriff & _
   "' and Wohnort='" & strBegriff2 & "'"
End If
```

```
    'Felder nicht ausgefüllt
    If .TextBox1.Value = "" And .TextBox2.Value = "" _
     And .TextBox3.Value = "" And .TextBox4.Value = "" Then
       MsgBox "Sie müssen die Suchkriterien noch angeben!", _
        vbExclamation
      .TextBox1.SetFocus
      Exit Sub
      End If
  End With


weiter:
  Set AccApp = CreateObject("Access.Application")
  'AccApp.Visible = True
  Set RST = New ADODB.Recordset
  conn.Open "Provider=Microsoft.Jet.OLEDB.4.0; " & _
  "Data source=" & ThisWorkbook.Path & "\Adressen.mdb;"

  'Tabelle öffnen und abfragen
  If BSchalter = False Then
   RST.Open _
  "Select * FROM Adressen where (" & strBegriff & ")", _
   conn, adOpenKeyset, adLockOptimistic
Else
  RST.Open _
  "Select * FROM Adressen", conn, adOpenKeyset, _
   adLockOptimistic
End If

'Gefundene Sätze in mehrspaltiges Listenfeld einfügen
 With UserForm1
  Do Until RST.EOF
    .ListBox1.AddItem RST!Name
    .ListBox1.Column(1, intZ) = RST!Vorname
    .ListBox1.Column(2, intZ) = RST!Geburtstag
    .ListBox1.Column(3, intZ) = RST!Wohnort
    .ListBox1.Column(4, intZ) = RST!Beruf
    intZ = intZ + 1
     RST.MoveNext
  Loop
  If .ListBox1.ListCount = 0 Then
    .Label6.Caption = "Keine Datensätze gefunden!"
  Else
    .Label6.Caption = .ListBox1.ListCount &  _
     " Datensätze gefunden!"
  End If
    .Label6.Visible = True
End With
RST.Close
AccApp.Quit
Set RST = Nothing
Set conn = Nothing

End Sub
```

Zu Beginn werden einige Variablen sowie die Objektvariablen für den Zugriff auf die Access-Datentabelle über ADO definiert. Vor jeder neuen Suche wird das Listenfeld über die Methode `Clear` gelöscht und die Zählvariable `intZ` auf den Wert `0` gesetzt. Mit dieser Zählvariablen werden später die einzelnen Spalten des mehrspaltigen Listenfeldes angesprochen. Die boolesche Variable `BSchalter` wird standardmäßig auf den Wert `False` gesetzt. Der Wert `True` ergibt sich, wenn Sie einen Stern in das Feld NAME eingeben und danach auf die Schaltfläche SUCHEN klicken. In diesem Fall findet keine Suche statt, sondern es werden alle Datensätze der Access-Datentabelle in das Listenfeld eingelesen.

In den nachfolgenden Zeilen wird ermittelt, welche Suche vom Anwender auf Basis seiner Eingabe gewünscht ist. Je nachdem, welche Felder mit Suchkriterien ausgefüllt wurden, wird entweder ein Suchbegriff aus einem oder aus zwei einzelnen Feldern zusammengesetzt und später dann an die SELECT-Anweisung übergeben. Innerhalb dieses zusammengesetzten Suchbegriffs muss der Feldname des jeweiligen Access-Datentabellenfeldes mit angegeben werden.

Mithilfe der Funktion `CreateObject` wird ein neues Access-Objekt erstellt. Die Zeile `AccApp.Visible = True` ist im Listing als Kommentar hinterlegt worden. Wenn Sie diese Zeile aktivieren, dann können Sie erkennen, dass ein Access-Objekt erstellt wird, d.h., Microsoft Access wird dabei geöffnet und ist sichtbar (`Visible = true`). Standardmäßig bleibt Microsoft Access aber im Hintergrund (`Visible = False`), damit der Anwender von dem Zugriff auf die Access-Datentabelle nichts mitbekommt.

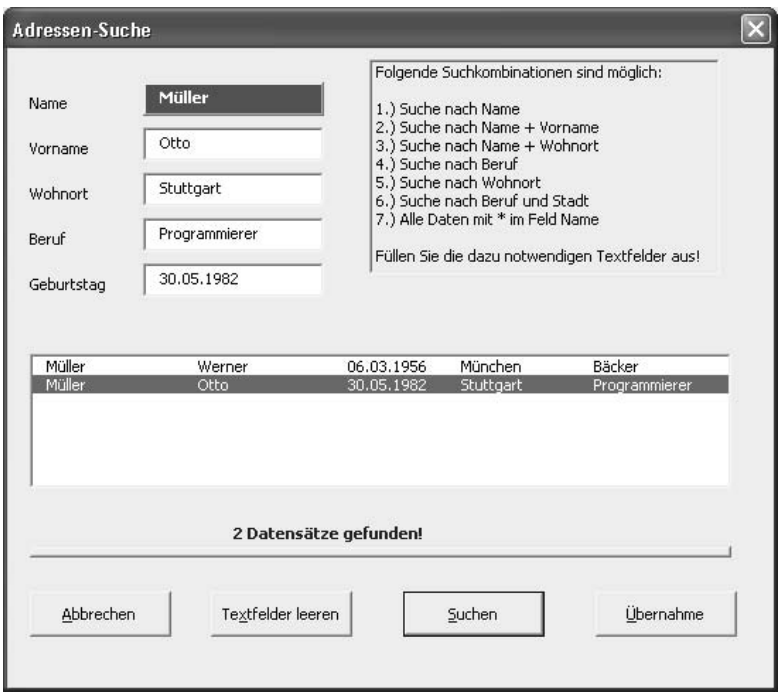


Abbildung 3.3: Die gefundenen Personen werden im Listenfeld angezeigt.

Über die Methode `Open` öffnen Sie die Datenbank ADRESSEN.MDB. Gleichzeitig wird eine SQL-Abfrage durchgeführt, die auf Basis des vorher zusammengesetzten Suchbegriffs stattfindet. Für den Fall, dass vorher in dem Feld NAME ein Stern eingegeben wurde, genügt es, den SELECT-Befehl ohne weitere Argumente einzusetzen.

Alle gefundenen Adressen werden nun im `RecordSet`-Objekt `rst` zur Verfügung gestellt und können dort direkt abgefragt werden. Übertragen werden diese Informationen in das Listenfeld mithilfe der Methode `AddItem`. Über die Eigenschaft `Column` kann nun jede einzelne Spalte des mehrspaltigen Listenfeldes gefüllt werden. Über die Methode `MoveNext` wird das jeweils nächste gefundene Element im `RecordSet` eingestellt. So werden alle Elemente nach und nach ins Listenfeld übertragen, bis die Bedingung `EOF` (End of File) zutrifft. Konnten über die Suche keine Datensätze ermittelt werden, dann bleibt das Listenfeld leer. Dabei liefert die Eigenschaft `ListCount` den Wert `0`, was eine Beschriftung des Bezeichnungsfeldes mit KEINE DATENSÄTZE GEFUNDEN bewirkt. Ansonsten wird in dem Bezeichnungsfeld die Anzahl der gefundenen Datensätze angegeben. Über die Eigenschaft `Visible = True` wird das Bezeichnungsfeld eingeblendet.

Im Anschluss daran wird die Access-Datentabelle über die Methode `Close` geschlossen und die Access-Anwendung mit der Methode `Quit` beendet. Die Objektverweise werden über das Schlüsselwort `Nothing` aufgehoben, um reservierten Arbeitsspeicher wieder freizugeben.

3.2.5 So werten Sie den Klick aufs Listenfeld aus

Wenn Sie auf einen der gefundenen Datensätze im Listenfeld klicken, dann sollen diese Daten automatisch in die oberen Textfelder übernommen werden (siehe Abbildung 3.3). Dazu können die Einträge direkt aus dem mehrspaltigen Listenfeld der UserForm entnommen und in die Textfelder eingelesen werden.

Listing 3.3: ListBox-Einträge in Textfelder übertragen

```
Private Sub ListBox1_Click()
'Daten aus Listenfeld in Textfelder übertragen
Dim intEintrag As Integer

 With UserForm1
   intEintrag = .ListBox1.ListIndex
  .TextBox1.Value = .ListBox1.Column(0, intEintrag)
  .TextBox2.Value = .ListBox1.Column(1, intEintrag)
  .TextBox4.Value = .ListBox1.Column(4, intEintrag)
  .TextBox3.Value = .ListBox1.Column(3, intEintrag)
  .TextBox5.Value = .ListBox1.Column(2, intEintrag)
 End With

End Sub
```

Mit jedem Klick auf das Listenfeld wird automatisch das Ereignis `Click` ausgelöst. Diesem Ereignis wurden in Listing 3.3 weitere Befehle zugewiesen, die dann beim Ereigniseintritt zusätzlich ausgeführt werden. Über die Eigenschaft `ListIndex`

kann festgestellt werden, welcher Eintrag im Listenfeld gerade angeklickt wurde. Dabei liefert der erste Eintrag im Listenfeld den Index `0`! Mithilfe der Eigenschaft `Column` greifen Sie direkt auf die einzelnen Spalten zu und übertragen die Inhalte über die Eigenschaft `Value` in die dazugehörigen Textfelder.

3.2.6 So wenden Sie Spezialeffekte an

Beim Verlassen eines Textfeldes bzw. beim Aktivieren eines Textfeldes werden die beteiligten Textfelder dynamisch formatiert. So bekommt ein Textfeld beim Aktivieren eine rote Hintergrundfarbe, die Schriftfarbe wird auf Weiß gesetzt und das Feld somit optisch hervorgehoben. Dazu werden zwei Makros benötigt, die Sie Listing 3.4 entnehmen können.

Listing 3.4: Dynamisches Hervorheben von Textfeldern

```
Sub FeldAktiv()

 With UserForm1
  .ActiveControl.BackColor = RGB(255, 0, 0)
  .ActiveControl.ForeColor = RGB(255, 255, 255)
  .ActiveControl.Font.Bold = True
  .ActiveControl.SpecialEffect = fmSpecialEffectRaised
 End With

End Sub


Sub FeldDeaktiv()

With UserForm1
 .ActiveControl.BackColor = RGB(255, 255, 255)
 .ActiveControl.ForeColor = RGB(0, 0, 0)
 .ActiveControl.Font.Bold = False
 .ActiveControl.SpecialEffect = fmSpecialEffectSunken
End With

End Sub
```

Über die Eigenschaft `BackColor` lässt sich der Hintergrund eines Textfeldes ansprechen. Mithilfe der Funktion `RGB` können Sie die gewünschte Farbe zusammenmischen. Die Farbe Rot hat dabei das Rot-Grün-Blau-Verhältnis `RGB(255, 0, 0)`. Da standardmäßig die schwarze Schriftfarbe verwendet wird und diese auf rotem Grund etwas schwer zu lesen ist, stellen Sie die Schriftfarbe Weiß per Farbenmischung `RGB(255, 255, 255)` ein. Zusätzlich soll der Text des Feldes im Schriftschnitt FETT dargestellt werden, indem der Eigenschaft `Bold` der Wert `True` zugewiesen wird. Einen räumlichen Effekt kann man über die Eigenschaft `SpecialEffect` erzielen, indem diesem die Konstante `fmSpecialEffectRaised` zugewiesen wird.

Beim Verlassen eines Textfeldes müssen diese Einstellungen wieder zurückgesetzt werden (siehe Listing 3.4, Makro `FeldDeaktiv`).

Die beiden Makros aus Listing 3.4 müssen jetzt den Textfeld-Ereignissen (`Enter =` beim Aktivieren bzw. `Exit =` beim Verlassen) zugewiesen werden. Exemplarisch sehen Sie diese Zuweisung in Listing 3.5.

Listing 3.5: Beim Betreten bzw. beim Verlassen von Feldern die entsprechenden Aktionen auslösen

```
Private Sub TextBox1_Enter()
  FeldAktiv
End Sub


Private Sub TextBox1_Exit _
(ByVal Cancel As MSForms.ReturnBoolean)
  FeldDeaktiv
End Sub
```

3.2.7 So leeren Sie Textfelder

Zusätzlich wird die UserForm mit einer weiteren Funktion bestückt. Über eine Schaltfläche soll es möglich sein, alle Textfelder sowie das Listenfeld von Einträgen zu säubern. Den Code für diese Aufgabe sehen Sie in Listing 3.6.

Listing 3.6: Textfelder und Listenfeld initialisieren

```
Private Sub CommandButton4_Click()
'Textfelder + Listenfeld leeren
Dim tb As Control

  With UserForm1
   For Each tb In .Controls
       If TypeName(tb) = "TextBox" Then
          tb.Value = ""
       End If
   Next tb
.ListBox1.Clear
.Label6.Caption = ""
.Label6.Visible = False
End With

End Sub
```

In einer Schleife werden alle einzelnen Steuerelemente der UserForm nacheinander angesprochen. Innerhalb der Schleife wird überprüft, ob es sich bei dem jeweiligen Element um ein Textfeld handelt. Bei dieser Prüfung wird die Funktion `TypeName` verwendet, wobei Sie bei der Prüfung auf die korrekte Schreibweise von `TextBox` achten müssen. Diese Funktion unterscheidet zwischen Groß- und Kleinschreibung.

Um das Listenfeld zurückzusetzen, müssen nicht die einzelnen Einträge gelöscht werden, sondern über die Methode `Clear` wird der komplette Inhalt des Listenfeldes gelöscht.

3.3 So übernehmen Sie Daten in das Excel-Formular

Wurden Daten aus Access gefunden und im Listenfeld eingelesen, dann können die Daten des markierten Datensatzes im Listenfeld in ein Tabellenblatt FORMULAR übertragen werden. Dazu starten Sie das Makro aus Listing 3.7.

Listing 3.7: Daten in Excel-Tabelle übernehmen

```
Private Sub CommandButton3_Click()
'Übernahme der Daten in das Formular
Dim intEintrag As Integer

  With UserForm1
   If .ListBox1.ListIndex = -1 Then
    MsgBox "Bitte einen Eintrag im Listenfeld markieren!"
   Else
   intEintrag = .ListBox1.ListIndex
   Sheets("Formular").Range("C6").Value = _
   .ListBox1.Column(0, intEintrag)
   Sheets("Formular").Range("C8").Value = _
   .ListBox1.Column(1, intEintrag)
   Sheets("Formular").Range("C10").Value = _
   .ListBox1.Column(2, intEintrag)
   Sheets("Formular").Range("C12").Value = _
  .ListBox1.Column(3, intEintrag)
   Sheets("Formular").Range("C14").Value = _
  .ListBox1.Column(4, intEintrag)
 End If
End With

End Sub
```

Da die Daten in der Excel-Tabelle immer an derselben Position eingefügt werden, können Sie den markierten ListBox-Eintrag über eine Zuweisung direkt in diese Zellen »entleeren«.

Abbildung 3.4: Übernahme der Daten in eine Excel-Tabelle

3.4 So führen Sie den Datenexport aus Excel durch

Als nächste Aufgabe führen Sie einen Datenexport aus Excel bzw. einen Datenimport in die Access-Datentabelle durch. Hierfür werden neue Daten über die UserForm2 eingegeben und per Klick direkt in die im Hintergrund liegende Access-Tabelle geschrieben. Dazu wird die UserForm aus Abbildung 3.5 eingesetzt und folgendes Makro aus Listing 3.8 ausgeführt.

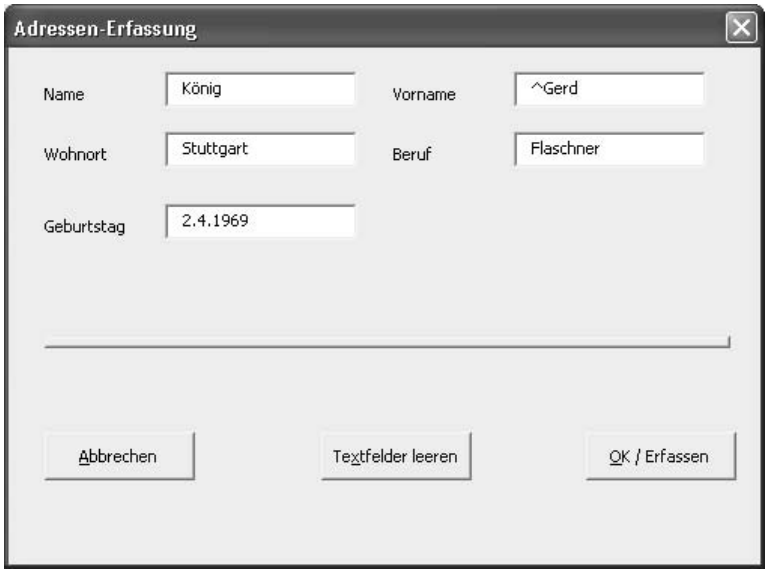


Abbildung 3.5: Datenerfassung in Excel – Speicherung in Access

Listing 3.8: Daten aus Excel direkt in Access speichern

```
Private Sub CommandButton2_Click()
'Daten zurückschreiben
Dim AccApp As Object
Dim conn As New ADODB.Connection
Dim rst As ADODB.Recordset

  Set AccApp = CreateObject("Access.Application")
  Set rst = New ADODB.Recordset
  conn.Open "Provider=Microsoft.Jet.OLEDB.4.0; " & _
  "Data source=" & ThisWorkbook.Path & "\Adressen.mdb;"

  rst.Open "Adressen", conn, adOpenKeyset, adLockOptimistic

  rst.AddNew

  With UserForm2
   rst!Name = .TextBox1.Value
   rst!Vorname = .TextBox2.Value
   rst!Wohnort = .TextBox3.Value
   rst!Beruf = .TextBox4.Value
```

```
    rst!Geburtstag = .TextBox5.Value
    rst.Update
    UserForm2.Label5.Visible = True
    UserForm2.Label5.Caption = "Satz erfasst!"
    rst.Close
  End With
  AccApp.Quit
  Set rst = Nothing
  Set conn = Nothing

End Sub
```


Auch in diesem Beispiel muss zuerst ein Access-Objekt über die Funktion `CreateObject` erstellt sowie die Datenbank und die Datentabelle müssen über die Methode `Open` geöffnet werden. Mithilfe der Methode `AddNew` wird ein neuer, noch leerer Satz am Ende der Access-Datentabelle angehängt. Dieser Satz wird danach gefüllt. Dabei werden die Inhalte der Textfelder direkt in die Datenbankfelder geschrieben. Allerdings wird erst mit dem Einsatz der Methode `Update` der neue Satz endgültig angelegt. Zum Abschluss wird, wie bereits vorher beschrieben, die Datentabelle geschlossen, die Access-Applikation beendet sowie die Objektverweise werden aufgehoben.

3.5 So führen Sie eine Datenbankabfrage durch

Am Ende des Kapitels bleiben noch zwei Aufgaben übrig:

- Wie können die drei ältesten Personen ermittelt werden?
- Wie kann der Name des »Juniors« abgefragt werden?

Beide Fragen werden durch das Listing 3.9 beantwortet. Die Makros werden direkt hinter die Befehlsschaltflächen gelegt, die vorher über die Symbolleiste STEUERELEMENT-TOOLBOX ausgewählt wurden.

Listing 3.9: Eine Datenbankabfrage starten

```
Sub CommandButton1_Click()
'Den Ältesten finden
Dim AccApp As Object
Dim conn As New ADODB.Connection
Dim rst As ADODB.Recordset
Dim intZ As Integer
Dim ArrPersonName(2) As Variant
Dim ArrPersonVorname(2) As Variant
Dim ArrDatum(2) As Variant

  Set AccApp = CreateObject("Access.Application")
  Set rst = New ADODB.Recordset
  conn.Open "Provider=Microsoft.Jet.OLEDB.4.0; " & _
  "Data source=" & ThisWorkbook.Path & "\Adressen.mdb;"
```

```
  rst.CursorLocation = adUseClient
  rst.Open "Adressen", conn, adOpenKeyset, adLockOptimistic
  rst.Sort = "Geburtstag Asc"

  For intZ = 0 To 2
    ArrPersonName(intZ) = rst!Name
    ArrPersonVorname(intZ) = rst!Vorname
    ArrDatum(intZ) = rst!Geburtstag
    rst.MoveNext
  Next intZ

  MsgBox "Die älteste Person heißt " & ArrPersonVorname(0) _
  &  " " & ArrPersonName(0) & " und ist am " & _
  ArrDatum(0) & " geboren!" & vbLf & vbLf & _
  "Die zweit-älteste Person heißt " & ArrPersonVorname(1) _
   & " " & ArrPersonName(1) & " und ist am " & _
   ArrDatum(1) & " geboren!" & vbLf & vbLf & _
   "Die dritt-älteste Person heißt " & _
   ArrPersonVorname(2) & " " & ArrPersonName(2) & _
   " und ist am " & ArrDatum(2) & " geboren!", _
    vbInformation, "Alters-Check"
  rst.Close
  AccApp.Quit
  Set rst = Nothing
  Set conn = Nothing

End Sub


Private Sub CommandButton4_Click()
'Die jüngste Person finden
Dim AccApp As Object
Dim conn As New ADODB.Connection
Dim DBS As ADODB.Recordset
Dim StrPersonName As String
Dim StrPersonVorname As String
Dim DatPerson As Date

  Set AccApp = CreateObject("Access.Application")
  Set rst = New ADODB.Recordset
  conn.Open "Provider=Microsoft.Jet.OLEDB.4.0; " & _
  "Data source=" & ThisWorkbook.Path & "\Adressen.mdb;"

  rst.CursorLocation = adUseClient
  rst.Open "Adressen", conn, adOpenKeyset, adLockOptimistic
  rst.Sort = "Geburtstag Desc"

  StrPersonName = rst!Name
  StrPersonVorname = rst!Vorname
  DatPerson = rst!Geburtstag

  MsgBox "Die jüngste Person heißt " & StrPersonVorname & _
" " & StrPersonName & " und ist am " & DatPerson & _
 " geboren!", vbInformation, "Junior-Check"
```

```
  rst.Close
  AccApp.Quit
  Set rst = Nothing
  Set conn = Nothing

End Sub
```

Beim jeweiligen Öffnen der Datentabelle wird eine Sortierung eingestellt, und dabei werden im ersten Makro die ersten drei Datensätze, im zweiten Makro der erste Datensatz abgegriffen, in einem Datenfeld gespeichert und dann am Bildschirm dargestellt.