11 VBA ausprobieren

»Chi lascia la strada vecchia per la nuova, sa quel che lascia ma non quel che trova. – Wer die alte Straße wegen der neuen verlässt, weiß, was er verlässt, aber nicht, was er findet.« In diesem Kapitel werden Sie neue Wege beschreiten und Ihre ersten Schritte in der VBA-Programmierung machen. Dabei steht das Ausprobieren im Mittelpunkt. Ab Kapitel 12 folgt dann etwas mehr Theorie.

11.1 Warum überhaupt VBA?

Wenn Sie jetzt mit Makros gut zurechtkommen, stellt sich natürlich die Frage, warum Sie sich überhaupt mit VBA beschäftigen sollen. Die Antwort ist einfach: Wenn Sie Ihre *Access*-Datenbanken weiter verbessern und professioneller gestalten möchten, gelangen Sie irgendwann an einen Punkt, an dem Sie mit Makros nicht weiterkommen. Deshalb sollten Sie sich frühzeitig an den Umgang mit VBA gewöhnen.

Sie werden sehen, dass das Erlernen von VBA gar nicht so schwer ist, zumal Sie einiges aus den Erfahrungen ableiten können, die Sie mit Makros gesammelt haben. Und ich kann Ihnen sagen, dass auch die englischen Schreibweisen (die mir als Italienerin am Anfang so in's Auge sprangen) schnell ihren Schrecken verlieren.

11.2 Eingetragenen VBA-Code ändern

Am Ende des letzten Kapitels hatte ich Sie gebeten, dem Startformular *frmStart* eine Befehlsschaltfläche hinzuzufügen, mit der die Anwendung beendet, d.h. *Access* geschlossen wird. Der Assistent hatte dabei automatisch einige Programmzeilen in VBA erstellt. Diesen VBA-Code wollen wir gleich einmal anschauen und ändern.

Als Arbeitserleichterung *frmStart* zur eigenen Symbolleiste hinzufügen

Da Sie in diesem Kapitel viele Dinge anhand des Startformulars *frmStart* ausprobieren werden, lohnt es sich, für dieses Formular zu Ihrer eigenen Symbolleiste eine Schaltfläche hinzuzufügen. Gehen Sie dabei vor wie im letzten Kapitel beschrieben.

☑ Öffnen Sie bitte das Formular *frmStart* in der Entwurfsansicht, markieren Sie die Befehlsschaltfläche *cmdSchließen* und lassen sich in den Eigenschaften das Registerblatt EREIGNIS anzeigen.

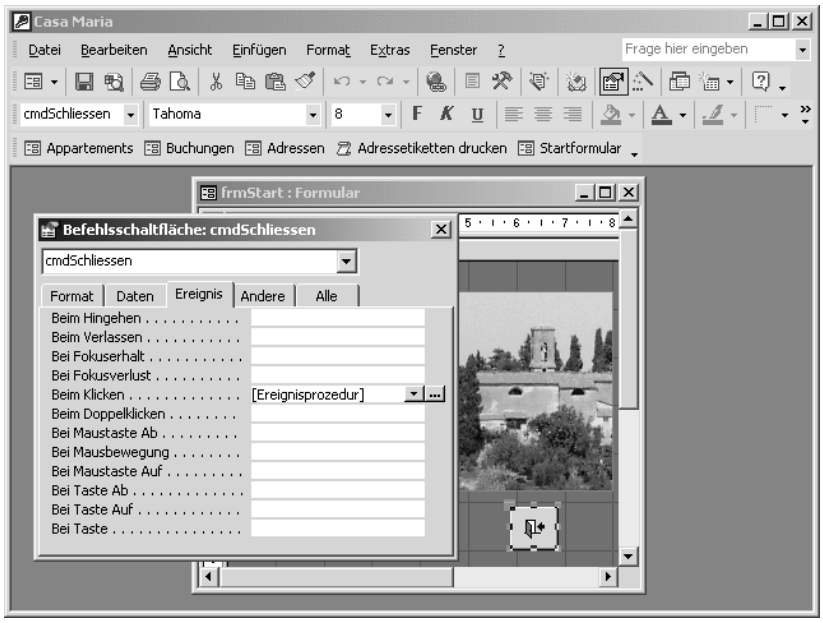
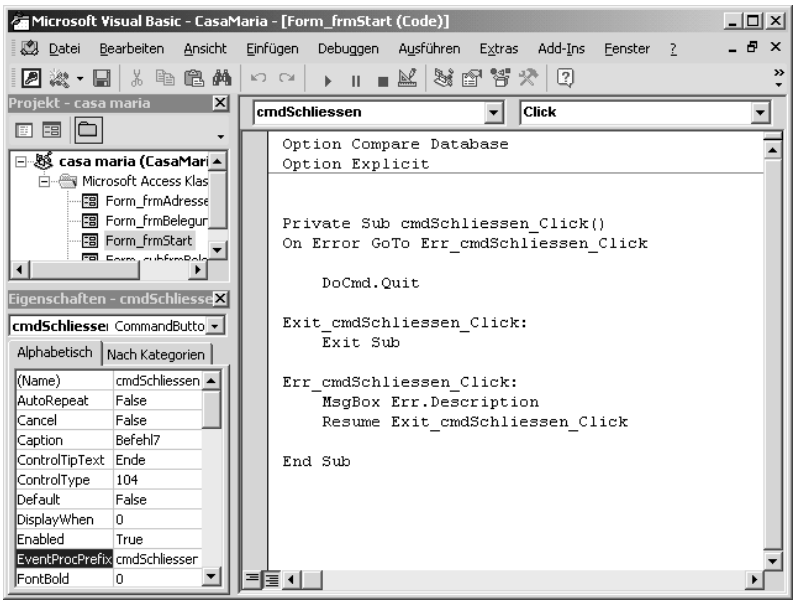


Bild 11.1: Die eingetragene Ereignisprozedur wird geöffnet

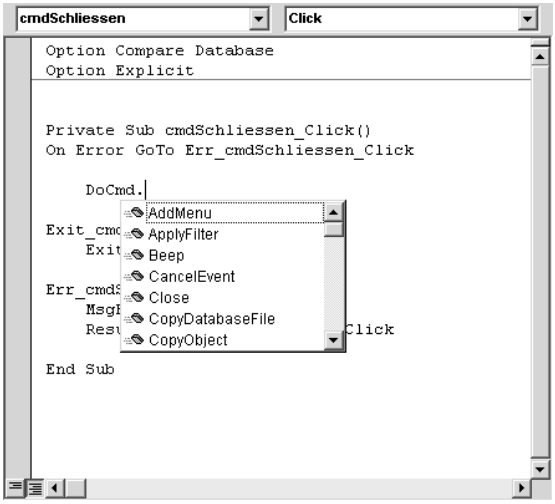
☑ Klicken Sie in die Zeile BEIM KLICKEN und dann auf die drei Punkte am Ende. Damit wechseln Sie zum VBA-Entwurfsfenster. (Die einzelnen Komponenten des VBA-Entwurfsfensters beschreibe ich Ihnen im nächsten Kapitel.)

Bild 11.2:
Der VBA-Code soll geändert werden



Die entscheidende VBA-Zeile lautet: `DoCmd.Quit`. Mit `Quit` wird festgelegt, dass das Programm geschlossen wird. Entfernen Sie `Quit` und den Punkt davor. Dann fügen Sie den Punkt wieder hinzu. In diesem Augenblick bekommen Sie eine Liste mit möglichen Einträgen angezeigt.

Bild 11.3:
Wählen Sie `Close` statt `Quit`



☑ Wenn Sie in der Liste auf `Close` doppelklicken, wird damit eingetragen, dass nur noch das Formular geschlossen und nicht mehr das ganze Programm beendet werden soll.

Wird bei Ihnen keine Liste mit Vorschlägen angezeigt?

Falls Sie wie beschrieben vorgegangen sind, aber trotzdem keine Vorschlagsliste angezeigt bekommen, überprüfen Sie bitte unter EXTRAS • OPTIONEN, ob dort alle Häkchen gesetzt sind.

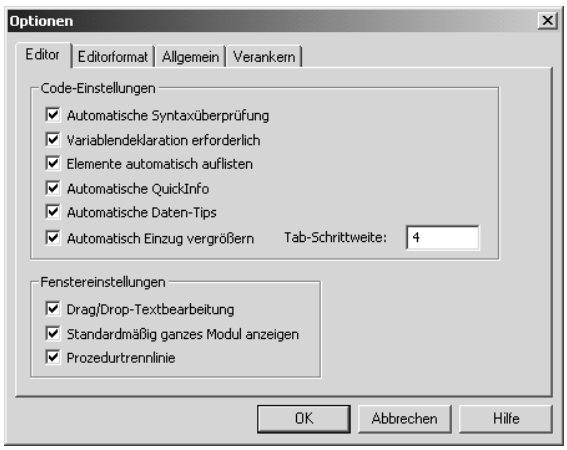


Bild 11.4: Zur Eingabeunterstützung alle Häkchen setzen

☑ Schließen Sie das VBA-Entwurfsfenster, speichern Sie das Formular *frmStart*. Der eingetragene VBA-Code wird dabei automatisch mitgespeichert.

Wenn Sie nun in der Formularansicht auf die Schaltfläche mit der Tür klicken, wird nur noch das Formular geschlossen und nicht mehr *Access* beendet.

11.3 VBA-Code neu erstellen für InputBox

In Kapitel 2 haben Sie bereits mit VBA ein Meldungsfenster aufrufen lassen. Auch im Makroteil des Buches haben Sie ausgiebig von der Möglichkeit Gebrauch gemacht, Angaben in Form von Meldungen auszugeben. Im letzten Kapitel habe ich Ihnen auch gezeigt, wie Sie die Meldungsbedingung in Makros nutzen können, um vom Anwender z.B. Ja-Nein-Entscheidungen abzufragen.

Mit der InputBox, die ich Ihnen jetzt vorstellen möchte, ist es auf einfache Weise möglich, Eingaben aufzunehmen, die dann weiter verarbeitet werden können.

11.3.1 InputBox nimmt Eintragungen auf

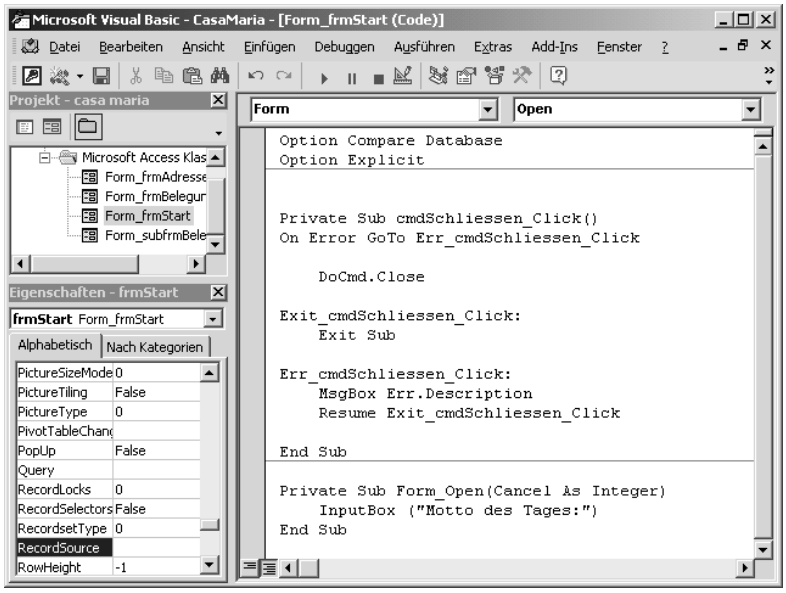
Beim Öffnen des Startformulars *frmStart* soll in einer InputBox das Motto des Tages abgefragt werden. Das Motto soll dann im Formular erscheinen. Statt eines Makros binden Sie nun einfach eine Ereignisprozedur an das BEIM ÖFFNEN-Ereignis des Formulars.

☑ Dazu öffnen Sie bitte das Formular *frmStart* in der Entwurfsansicht, klicken im Eigenschaftenfenster auf dem Registerblatt EREIGNIS in die Zeile BEIM ÖFFNEN und wählen dann nach Klick auf die drei Punkte am Ende den CODE-GENERATOR aus.

☑ Sie sehen im oberen Teil den Code, den Sie aus dem letzten Abschnitt kennen. Unten finden Sie die blinkende Einfügemarke zwischen `Private Sub` und `End Sub`. Hier schreiben Sie den folgenden Text: `inputbox ("Motto des Tages:")` und rücken den Text für die bessere Lesbarkeit mit der ↵-Taste ein.

☑ Klicken Sie in die nächste Zeile. Wenn Sie sich nicht vertippt haben, erscheint `InputBox` jetzt groß geschrieben.

Bild 11.5:
InputBox ist eingetragen



Wenn Sie nun erneut das Formular *frmStart* öffnen, wird die InputBox angezeigt. Übrigens müssen Sie das VBA-Entwurfsfenster dazu nicht schließen – wechseln Sie einfach über den Eintrag in der Taskleiste oder klicken Sie ganz links in der Symbolleiste auf das Symbol ANSICHT MICROSOFT ACCESS.

Bild 11.6:
InputBox fordert zur Eingabe auf

11.3.2 Eingabe aus InputBox weiterverarbeiten

Bisher werden die Angaben, die Sie in der InputBox machen, noch nirgends angezeigt. Das soll sich jetzt ändern.

Angaben in MessageBox anzeigen

Am einfachsten bekommen Sie die Eintragung mit einer Meldung (bzw. MessageBox) auf den Bildschirm. In Kapitel 2 hatten Sie die MessageBox mit `MsgBox` aufgerufen. Jetzt probieren Sie einmal Folgendes aus:

- ☑ Öffnen Sie das Formular *frmStart* wieder in der Entwurfsansicht, klicken Sie im Eigenschaftenfenster auf dem Registerblatt EREIGNIS in die Zeile BEIM ÖFFNEN und dort auf die drei Punkte am Ende. (Wenn Sie das VBA-Entwurfsfenster vorhin geöffnet gelassen hatten, reicht es, wieder dorthin zu wechseln!)
- ☑ Schreiben Sie einfach `msgbox` an den Anfang der Zeile, in der Sie die InputBox definiert hatten – fertig! Wenn Sie möchten, können Sie den Teil `(InputBox ("Motto des Tages:"))` noch in Klammern fassen, damit deutlicher sichtbar ist, dass der in der InputBox erfasste Text in der MessageBox angezeigt wird.

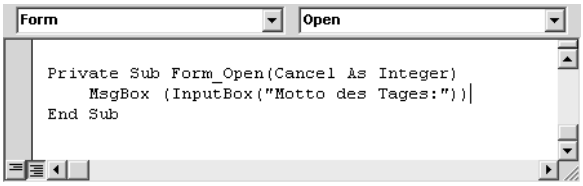


Bild 11.7:
Nicht die beste Lösung, aber es funktioniert

Wenn Sie nun erneut das Formular *frmStart* öffnen, erscheint zuerst die InputBox. Nachdem Sie das Motto des Tages eingetragen und auf OK geklickt haben, erscheint das Motto des Tages gleich danach in einer MessageBox.

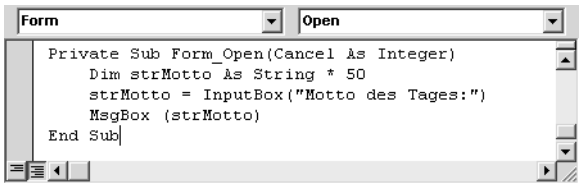
Bild 11.8:
Hier rein ...

Bild 11.9:
... da raus!

Schon mal eine Variable deklarieren

Dasselbe passiert, wenn Sie den in der InputBox eingegebenen Wert zwischenspeichern:

Bild 11.10:
So wird es besser



Mit der Schreibweise im obigen Bild haben Sie festgelegt, dass der Eintrag aus der InputBox in `strMotto` gespeichert wird. In der MessageBox wird dann der Inhalt von `strMotto` ausgegeben. Diese Schreibweise hat unter anderem den Vorteil, dass es nun ganz leicht ist, den in `strMotto` gespeicherten Wert auch in anderer Form auszugeben.

Ein solcher Zwischenspeicher wird übrigens als Variable bezeichnet. Die mit `Dim` beginnende Zeile dient zur Deklaration der Variablen `strMotto`. Das Deklarieren von Variablen ist ein Thema, auf das ich im nächsten Kapitel noch genauer eingehen werde.

Jetzt geht es erst einmal um ein praktisches Beispiel: Um zu verhindern, dass mehr Text gespeichert wird als später angezeigt werden kann, soll für die Variable `strMotto` festgelegt werden, dass

sie Text von maximal 50 Zeichen aufnehmen soll. Für diese Festlegung wird die Variable mit dem Ausdruck `Dim strMotto As String * 50` deklariert.

Ausgabe in Bezeichnungsfeld

Fügen Sie zum Formular *frmStart* ein neues Bezeichnungsfeld hinzu, in das Sie als Platzhalter ein beliebiges Zeichen, z.B. ein Leerzeichen eintragen und dem Sie z.B. den Namen `lblMotto` geben. Mit der Vorsilbe `lbl` für »label« werden Bezeichnungsfelder gekennzeichnet. Jetzt können Sie mit `lblMotto.Caption = strMotto` das in der InputBox eingetragene Motto im Bezeichnungsfeld anzeigen lassen. Dazu müssen Sie den obigen Ausdruck nur in einer weiteren Zeile zur oben erstellten Ereignisprozedur hinzufügen (s. Bild 11.11).

Sobald Sie bei der Eingabe den Punkt vor `Caption` gesetzt haben, wird wieder eine Liste mit Auswahlmöglichkeiten aufgeführt, aus der Sie per Doppelklick `Caption` auswählen können. (`Caption` steht übrigens für Überschrift bzw. Titel.)

Ausgabe als neue Beschriftung einer Befehlsschaltfläche

Nach demselben Muster können Sie auch die Beschriftung einer Befehlsschaltfläche ändern. Wenn Sie eine Befehlsschaltfläche mit der Bezeichnung `cmdMotto` hinzugefügt haben, müssen Sie den Ausdruck `cmdMotto.Caption = strMotto` in die Ereignisprozedur einfügen (s. Bild 11.11).

Ausgabe als neuer Eintrag in der Titelleiste

Auch der Eintrag in der Titelleiste des Formulars lässt sich auf diese Weise mit dem Ausdruck `Form_frmStart.Caption = strMotto` ändern (s. Bild 11.11).

Bild 11.11:
Vier Ausgabebeispiele für den InputBox-Eintrag

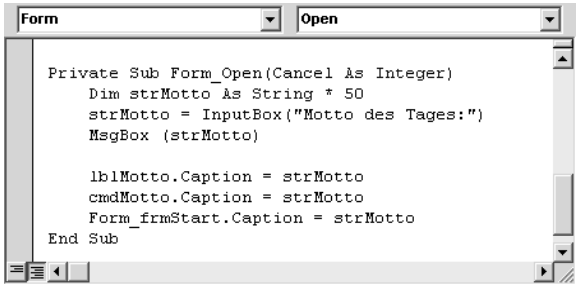
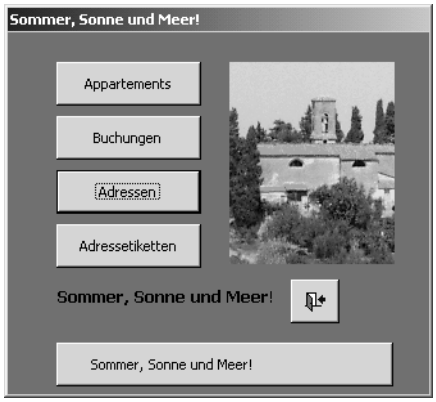


Bild 11.12:
So würde das Motto des Tages im Formular erscheinen



Mir persönlich gefällt das Motto des Tages am besten im Titel des Startformulars. Deshalb lösche ich die Zeilen für das Bezeichnungsfeld (`lblMotto.Caption = strMotto`), für die Befehlsschaltfläche (`cmdMotto.Caption = strMotto`) und für die MessageBox (`MsgBox (strMotto)`) aus der Prozedur. Die Befehlsschaltfläche `cmdMotto` entferne ich wieder aus dem Formular und das Bezeichnungsfeld `lblMotto` benenne ich in `lblZeit` um, denn hier soll die aktuelle Zeit angezeigt werden. Wie das geht, zeige ich Ihnen jetzt.

11.4 Aktuelle Uhrzeit im Formular anzeigen

Um im Formular *frmStart* im Bezeichnungsfeld LBLZEIT die aktuelle Zeit einzublenden,

☑ tragen Sie in den Eigenschaften des Formulars auf dem Registerblatt ALLE als ZEITGEBERINTERVALL den Zeittakt `1000` ein. Damit wird die sekundengenaue Anzeige eingestellt. (Der Wert wird in

Millisekunden angegeben – mit `60000` stellen Sie einen Zeittakt von einer Minute ein.)

☑ Klicken Sie in die Zeile BEI ZEITGEBER und dann auf die drei Punkte am Ende, um den Code-Generator aufzurufen.

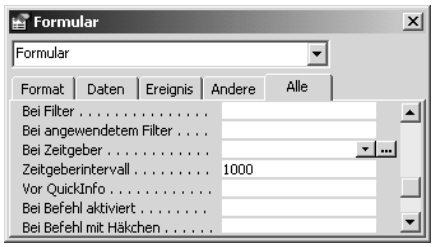


Bild 11.13: Für *frmStart* Zeitgeberintervall eintragen

☑ In der neuen Ereignisprozedur fügen Sie die Zeile `lblZeit.Caption = Now` ein.

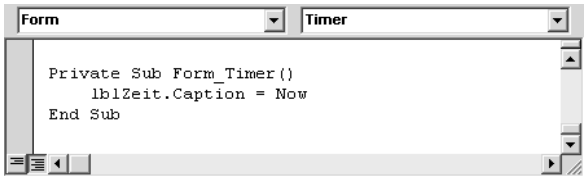


Bild 11.14: So wird die Zeit im Standardformat angezeigt

Wenn Sie jetzt das Formular *frmStart* in der Formularansicht anzeigen, wird die Zeit im Bezeichnungsfeld LBLZEIT im Format angezeigt, das in den Systemeinstellungen festgelegt ist.

Um Datum und Uhrzeit in einem eigenen Format anzuzeigen, nutzen Sie die `Format`-Funktion, die Sie ja schon aus den Makro-Kapiteln kennen. Allerdings müssen Sie jetzt die englische Schreibweise wählen (mit `d` für day als Tag und `y` für year als Jahr). In den folgenden beiden Bildern sehen Sie, wie ich das Format angegeben habe und wie das Ergebnis im Formular aussieht.

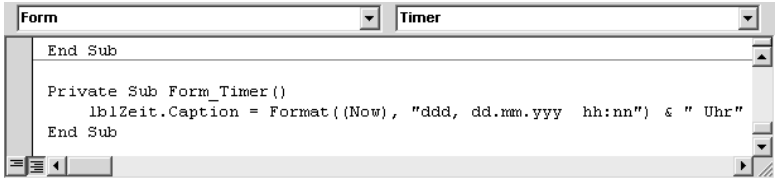
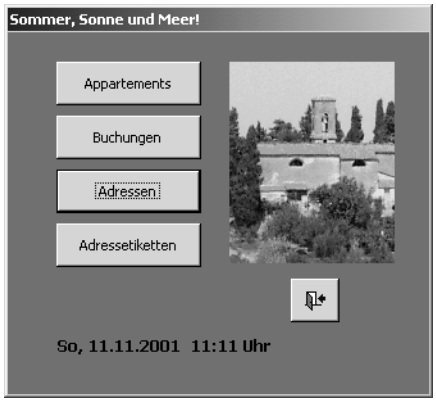


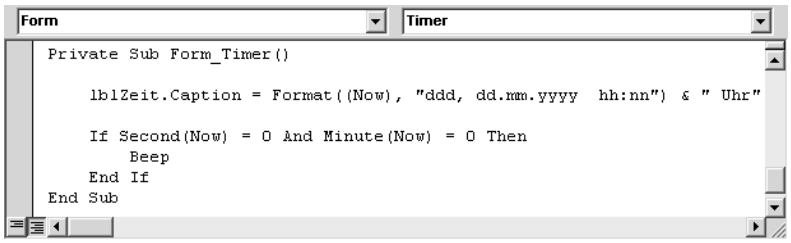
Bild 11.15: Auch ein eigenes Format kann definiert werden

Bild 11.16: Datums- und Uhrzeitanzeige im eigenen Format



Außerdem habe ich noch eingestellt, dass es zur vollen Stunde piepst – oder genauer gesagt: Es ertönt ein Beep. Wie im nächsten Bild zu sehen, habe ich dazu einen `If...Then`-Ausdruck hinzugefügt. Dieser Wenn...Dann-Ausdruck besagt, dass es piepsen soll, wenn sowohl die Sekunden als auch die Minuten der aktuellen Zeit gleich Null sind, denn dann handelt es sich um eine volle Stunde.

Bild 11.17: Zur vollen Stunde piepst es



Zum Ausprobieren können Sie den Teil `And Minute(Now) = 0` einfach entfernen, dann piepst es jede Minute! Wenn es bei Ihnen überhaupt nicht piepsen soll und Sie auch keine Zeit im Startformular angezeigt bekommen möchten, dann löschen Sie den gesamten Code von `Private Sub Form_Timer ()` bis `End Sub` und entfernen das Bezeichnungsfeld LBLZEIT aus dem Startformular.

11.5 Makro in VBA umwandeln

Selbst erstellte Makros lassen sich einfachin Visual Basic umwandeln. Um das Makro *mcrAdressenDrucken* zu konvertieren,

☑ markieren Sie es im Datenbankfenster und

☑ rufen Sie den Menübefehl EXTRAS • MAKROS • MAKROS ZU VISUAL BASIC KONVERTIEREN auf.

Bild 11.18:
Makro zu *Visual Basic* konvertieren

Ein Dialogfeld erscheint, in dem Sie wählen können, ob die Kommentare, die Sie im Makro gemacht haben, mit übernommen werden sollen und ob automatisch eine Fehlerbehandlung integriert werden soll. Wenn Sie beide Häkchen setzen, wird der erstellte VBA-Code etwas länger und für Sie möglicherweise etwas unübersichtlicher. Dafür ist dann auch alles Nötige enthalten.

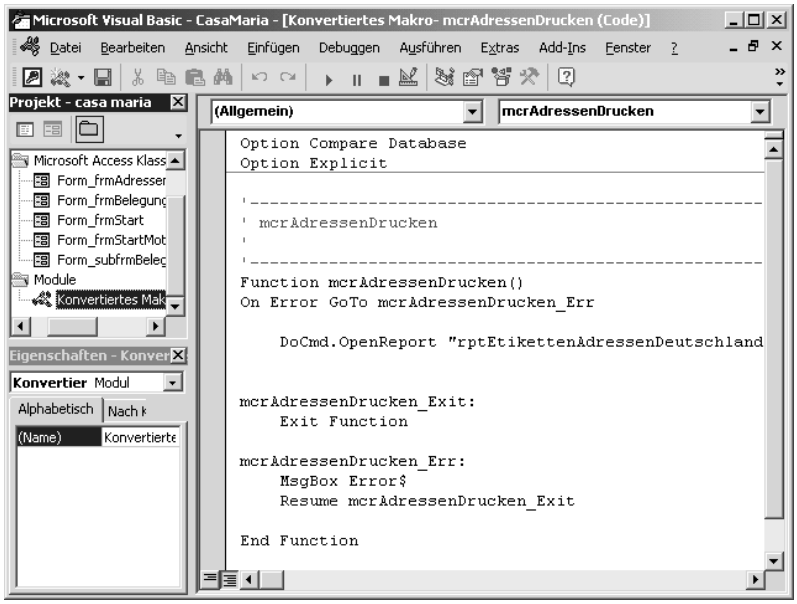
☑ Klicken Sie auf OK. Die Meldung, dass das Makro fertig konvertiert wurde, bestätigen Sie ebenfalls mit OK.

Das konvertierte Makro finden Sie, wenn Sie im Datenbankfenster auf der Objektleiste den Objekttyp MODULE wählen. Konvertierten Makros wird vor dem Namen des Makros der Text KONVERTIERTES MAKRO vorangestellt. Das eben erstellte Makro heißt also *Konvertiertes Makro- mcrAdressenDrucken*.

Bild 11.19:
Konvertiertes Makro im Datenbankfenster

Wenn Sie auf den Eintrag im Datenbankfenster doppelklicken, wird der erstellte Code im VBA-Entwurfsfenster angezeigt. Die zentrale Zeile beginnt mit `DoCmd.OpenReport`, diese VBA-Funktion entspricht der Makroaktion ÖFFNEN BERICHT. Die meisten der restlichen Zeilen dienen zur Fehlerbehandlung und regeln, was passieren soll, wenn ein Fehler auftritt.

Bild 11.20:
So sieht der
erstellte Code aus



Von Makros konvertierter Code ist nicht unbedingt der beste

Der beim Konvertieren von Makros zu VBA erstellte Code gibt nicht unbedingt die beste aller Lösungen wieder. Wenn Sie etwas mehr Erfahrung mit VBA haben, können Sie den konvertierten Code als Grundlage nehmen und ihn dann selber überarbeiten.

Konvertiertes Makro aufrufen

Wollen Sie ein konvertiertes Makro aus einem Formular heraus aufrufen, brauchen Sie nur den Namen der Funktion (also in unserem Beispiel *mcrAdressenDrucken*) in eine an das gewünschte Ereignis gebundene Prozedur einzufügen. Um mit Klick auf die ADRESSETIKETTEN-Schaltfläche nicht mehr das Ursprungsmakro, sondern die neue Prozedur auszuführen,

☑ öffnen Sie das Formular *frmStart* in der Entwurfsansicht und löschen Sie in den Eigenschaften auf dem Registerblatt EREIGNIS den Eintrag in der Zeile BEIM KLICKEN.

☑ Klicken Sie auf die drei Punkte am Ende der Zeile und wählen Sie den Code-Generator.

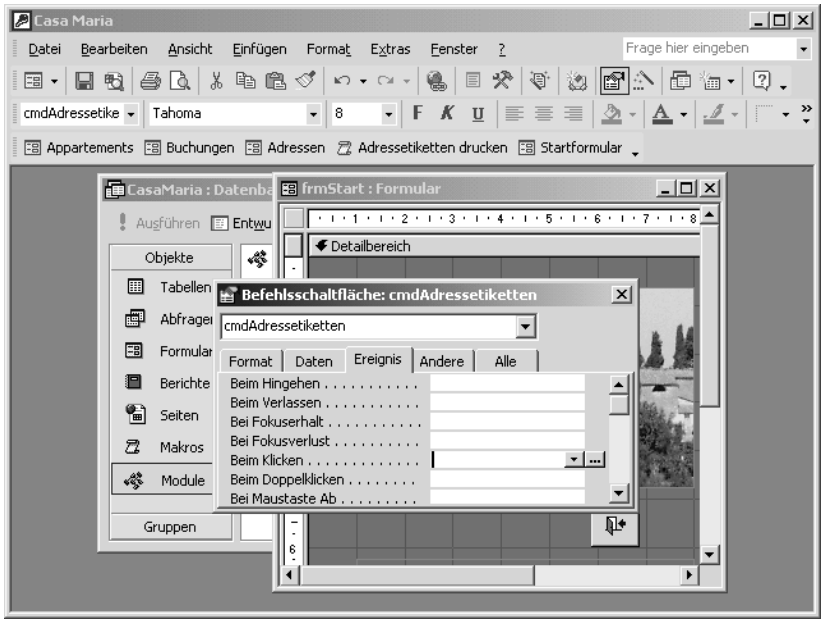


Bild 11.21:
BEIM KLICKEN-Eintrag löschen und Code-Generator wählen

☑ In der neuen Prozedur tragen Sie jetzt den Namen der Funktion ein, die bei konvertierten Makros dem Namen des Ursprungsmakros entspricht.

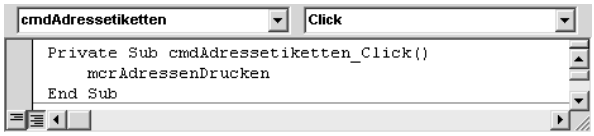


Bild 11.22:
Namen der Funktion eintragen

Wenn Sie nun im Formular *frmStart* auf die Schaltfläche ADRESSETIKETTEN klicken, wird zum Ausdrucken des Berichts die neu konvertierte Prozedur ausgeführt.

Ausblick auf die nächsten Kapitel

Nachdem Sie in diesem Kapitel schon einige praktische Erfahrungen im Umgang mit VBA gesammelt haben, folgt in den nächsten Kapiteln nun etwas Theorie. Damit erhalten Sie das nötige Hintergrundwissen, um sich sicherer auf den neuen VBA-Pfaden zu bewegen.