Bernd Held

Know-how ist blau.

Studienausgabe

VBA-Programmierung

für Word, Excel und Access

> Ergänzen Sie Microsoft Office durch selbst programmierte Funktionen

> Erstellen Sie Meldungen und Dialoge, Menü- und Symbolleisten

> Tauschen Sie Daten zwischen den Office-Applikationen aus

Das Praxisbuch für Microsoft-Office-Entwickler

FRANZIS

Bernd Held

VBA-Programmierung für Word, Excel und Access

Bernd Held

Studienausgabe

VBA-Programmierung
für Word, Excel und Access

Mit 323 Abbildungen

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte Daten sind im Internet über http://dnb.ddb.de abrufbar.

Herausgeber: Franz Graser
Satz: DTP-Satz A. Kugge, München
art & design: www.ideehoch2.de
Druck: Bercker, 47623 Kevelaer
Printed in Germany

ISBN 978-3-645-60070-5

Vorwort

Eine der Stärken des Office-Pakets von Microsoft ist, dass Sie es über den Einsatz von Visual Basic for Applications (VBA) erweitern und sogar noch verbessern können. Überall entstehen somit Office-Lösungen in Eigenregie, die den großen, bereits existierenden und oft sehr teuren Standardsoftwareprogrammen trotzen. Diese programmierten Office-Anwendungen können flexibler erstellt und auch jederzeit gepflegt werden. Dabei ist den Anwendern die Office-Oberfläche schon bekannt und die Einführung solcher Lösungen wird dadurch erleichtert.

In diesem Buch werden die wichtigsten Office-Applikationen Excel, Word und Access ausführlich behandelt. Zusätzlich werden Lösungen mit Outlook und PowerPoint präsentiert.

Das Buch ist in drei Teile gegliedert:

- Teil I: Die allgemeinen Office-Themen
- Teil II: Die wichtigsten Office-Komponenten und der Datenaustausch zwischen den einzelnen Office-Programmen
- Teil III: Fehlersuche, Tuning und Office-VBA-FAQ (Frequently Asked Questions, also häufig gestellte Fragen)

Im ersten Teil werden Office umfassende Themen behandelt, die Sie mit kleinen Anpassungen für nahezu alle Office-Komponenten einsetzen können. So lernen Sie im ersten Kapitel, wie Sie die Entwicklungsumgebung in den Office-Komponenten einsetzen können. Im zweiten Kapitel lernen Sie die wichtigsten Sprachelemente wie beispielsweise Schleifen, Verzweigungen und Abfragen kennen. In Kapitel 3 werden VBA-Standardfunktionen des Office-Pakets anhand von Aufgaben aus der täglichen Praxis vorgestellt. Im folgenden Kapitel erfahren Sie, wie Sie Office durch die Programmierung eigener Funktionen weiter bereichern können. In Kapitel 5 erstellen und programmieren Sie benutzerfreundliche UserForms und erfahren, wie Sie bereits integrierte Dialoge im Office-Paket für Ihre Programme einsetzen können. Die Programmierung von Kontextmenüs, Menü- und Symbolleisten ist Thema von Kapitel 6. Dort erfahren Sie, wie Sie diese Objekte ansprechen und programmieren können, um Ihre VBA-Projekte noch benutzerfreundlicher zu machen. In Kapitel 7 dreht sich alles um das Internet. Unter anderem erfahren Sie dort, wie Sie Hyperlinks programmieren, E-Mails verschicken und Web-Abfragen ausführen können. In Kapitel 8 greifen Sie mithilfe von VBA-Makros auf VBA-Projekte zu. Durch diese so genannten VBE-Zugriffe können Sie Quellcode sichern oder entfernen. Des Weiteren besteht die Möglichkeit, Quellcode zu importieren bzw. zu exportieren. Selbstverständlich können Sie über ein Makro auch weitere Makros Zeile für Zeile automatisch erstellen.

Im zweiten Teil wird der Fokus auf die wichtigsten Office-Komponenten gerichtet. In Kapitel 9 lernen Sie die wichtigsten Objekte in Excel wie Zellen, Zeilen, Spalten, Tabellen und Arbeitsmappen anhand zahlreicher Beispiele aus der täglichen Praxis kennen. In Kapitel 10 erstellen Sie Makros in Ihrer Textverarbeitung Word. Unter anderem fügen Sie Bilder in Dokumente ein, erstellen und füllen Tabellen und vieles mehr. In Kapitel 11 lernen Sie die Programmierung der wichtigsten Elemente von Access kennen. Sie erstellen unter anderem

Tabellen und führen Abfragen durch. Eines der wichtigsten Kapitel dieses Buchs stellt das Kapitel 12 dar. Es behandelt den Datenaustausch zwischen den einzelnen Office-Komponenten. In diesem Kapitel steht die Zusammenarbeit der einzelnen Office-Komponenten im Vordergrund. Unter anderem greifen Sie ausgehend von Word auf eine Access-Tabelle zu und übertragen Adressdaten in Ihre Textverarbeitung. Des Weiteren füllen Sie den Kontaktordner von Outlook mit Daten, die Sie aus einer Excel-Tabelle beziehen.

Im dritten Teil des Buchs erfahren Sie in Kapitel 13, wie Sie Fehler in Makros aufspüren und beseitigen können. Allgemeine Ratschläge für das Verhalten im Fehlerfall ergänzen den Inhalt dieses Kapitels. In Kapitel 14 messen Sie die Geschwindigkeit von einzelnen Makros. Schritt für Schritt erfahren Sie, wie Sie Makros noch schneller und benutzerfreundlicher machen können. Im letzten Kapitel dieses Buchs präsentiere ich Ihnen eine Office-FAQ mit Fragen, die in den letzten Jahren an mich gestellt wurden und für die ich eine Lösung zur Verfügung stellen konnte.

In Anhang A des Buchs finden Sie eine Auflistung der Dateien, die Sie auf der Website *www.buch.cd* finden können. Des Weiteren werden im Anhang B alle Listingbeschriftungen in einer Liste zum schnellen Nachschlagen angeboten. Im Index sind alle im Buch verwendeten Methoden, Funktion, Eigenschaften und Anweisungen aufgenommen worden. Sie können somit schnell herausfinden, auf welchen Seiten des Buchs die Befehle verwendet und beschrieben wurden.

In diesem Buch wurden Praxislösugnen in ca. 360 Makros beschrieben. Zögern Sie nicht, mich bei Fragen zum Buch über meine E-Mail-Adresse `held-office@t-online.de` zu kontaktieren.

Ich wünsche Ihnen beim Lesen des Buchs viel Spass und hoffe, dass ich damit einen Beitrag leisten konnte, der Ihnen hilft, erfolgreich Office-Lösungen zu entwickeln.

Bernd Held

Über den Autor:

Bernd Held ist gelernter Informatiker und programmierte drei Jahre lang bei einer Firma der Automobilbranche Warenwirtschafts- und Suchsysteme für den Kfz-Bereich. Danach arbeitete er sechs Jahre beim debis Systemhaus im Controlling. Dort war er verantwortlich für das Berichtswesen, die Leistungsverrechnung, das Erstellen von betrieblichen Auswertungen und Wirtschaftlichkeitsrechnungen sowie für die Entwicklung neuer Controlling-Tools auf der Basis von Microsoft Office. Seit dem 1. Januar 2002 ist Herr Held selbstständig. Er schreibt Fachartikel in renommierten Zeitschriften, verfasst Computerbücher, führt Software-Schulungen durch und programmiert im Auftrag von Kunden. Sein Spezialgebiet ist Microsoft Office. Dort hat er sich auf den Bereich Excel und die Office-VBA-Programmierung spezialisiert. Aber auch über Microsoft Works, FrontPage, Windows und diverse anderer Themen hat er schon viele Bücher geschrieben.

Vor drei Jahren wurde Bernd Held als MVP (Most Valuable Professional) von der Firma Microsoft ausgezeichnet. Dieser Titel wird für besondere fachliche Kompetenz, überdurchschnittlichen Einsatz in den Diskussionsforen und für außergewöhnliches Kommunikationstalent verliehen. Im deutschsprachigen Excel-Forum von Microsoft (news:microsoft.public.de.excel) können Sie Bernd Held des öfternen antreffen, wenn Sie Fragen zu Excel oder zur VBA-Programmierung haben. Dort hilft er Ihnen gerne weiter.

Inhaltsverzeichnis

Vorwort **5**

1 Die Entwicklungsumgebung von VBA **13**
1.1 Makros einfügen 13
1.2 Makros starten 15
1.3 Den Makrorekorder einsetzen 15
1.4 Die Arbeitsumgebung 20
1.5 Wertvolle Helfer bei der Programmierung 28
1.6 Weitere Einstellungen 35

2 Die Sprachelemente von VBA **41**
2.1 Variablen und Konstanten 41
2.2 Operatoren 44
2.3 Verzweigungen 46
2.4 Die Anweisung Select Case 50
2.5 Schleifen 58

3 VBA-Standardfunktionen nutzen **91**
3.1 Textfunktionen einsetzen 91
3.2 Mit Verzeichnissen und Laufwerken arbeiten 108
3.3 Datums- und Zeitfunktionen einsetzen 112
3.4 Prüffunktionen 126
3.5 Sonstige Funktionen 136

4 Eigene Funktionen programmieren **147**
4.1 Farbige Zellen addieren 147
4.2 Dateiprüfung 149
4.3 Daten bereinigen 150
4.4 Aktive Zelle im Zielbereich? 153
4.5 Dokumentschutz aufheben und neu setzen 154
4.6 Römische Zahlen wandeln 156
4.7 Eingefügte Objekte in PowerPoint-Folien ermitteln 158
4.8 Läuft eine Anwendung bereits? 162
4.9 Hyperlinks auf Shape-Objekten identifizieren 167
4.10 Ist Dokument passwortgeschützt? 170
4.11 Ist Arbeitsmappe passwortgeschützt? 171
4.12 Wo bin ich? 173
4.13 Das älteste Dokument in einem Verzeichnis ermitteln 176
4.14 Die Dokumenteigenschaften ermitteln 178
4.15 Wie viele Tage hat ein Monat? 180
4.16 Initialen aus Namen bilden 180

4.17 Automatisch E-Mail-Adressen generieren ... 183
4.18 Ist Add-In bereits eingebunden ... 184
4.19 Wo steckt der größte Wert? ... 185
4.20 Wird Name bereits verwendet? ... 188

5 Meldungen, Eingabemasken, Dialoge und UserForms programmieren.... 191
5.1 Meldungen programmieren ... 191
5.2 Eingabemasken programmieren ... 194
5.3 Integrierte Dialoge verwenden ... 197
5.4 UserForms programmieren ... 203

6 Menü- und Symbolleisten programmieren ... 231
6.1 Allgemeine Anmerkungen zu Leisten ... 231
6.2 Menüleisten programmieren ... 233
6.3 Symbolleisten programmieren ... 251
6.4 Kontextmenüs programmieren ... 260

7 »Internette« Funktionen in Office programmieren ... 265
7.1 Inhaltsverzeichnis einer Arbeitsmappe erstellen ... 265
7.2 Inhaltsverzeichnis eines Verzeichnisses erstellen ... 267
7.3 E-Mail-Links einfügen ... 269
7.4 URL-Links einfügen ... 273
7.5 Aktienkurse abfragen ... 276
7.6 E-Mails per VBA verschicken ... 278

8 VBE-Programmierung in Office ... 291
8.1 Die Voraussetzung ... 291
8.2 Bibliotheken einbinden ... 292
8.3 Die VBE-Komponenten ... 296
8.4 VBE-Komponenten auflisten ... 298
8.5 VBE-Komponenten entfernen ... 300
8.6 VBE-Komponenten exportieren ... 300
8.7 VBE-Komponenten importieren ... 301
8.8 Alle VBE-Komponenten aus Dokument entfernen ... 304
8.9 VBE aufrufen ... 305
8.10 Codezeilen auflisten ... 305

9 Excel-Programmierung ... 309
9.1 Zellen programmieren ... 309
9.2 Zeilen und Spalten programmieren ... 328
9.3 Tabellen programmieren ... 343
9.4 Arbeitsmappen programmieren ... 355

10 Die Programmierung mit Word ... 367
10.1 Dokument(e) identifizieren ... 368
10.2 Dokumentvorlage ermitteln ... 369
10.3 Dokumentvorlage wechseln ... 370
10.4 Einstellungen am Dokument durchführen ... 371

10.5 Schriftarten ermitteln ... 372
10.6 Dokumenteigenschaften auslesen und setzen ... 374
10.7 Kommentare aufspüren und auslesen ... 380
10.8 Texte/Formate suchen und ersetzen ... 383
10.9 Arbeiten mit Hyperlinks ... 397
10.10 Bilder in Dokumenten verarbeiten ... 401
10.11 Arbeiten mit Tabellen ... 406

11 Programmierung mit Access ... 411
11.1 Das Programmieren von Tabellen ... 411
11.2 Das Programmieren von Abfragen ... 436
11.3 Das Programmieren von Formularen ... 450

12 Office im Zusammenspiel ... 459
12.1 Adressen nach Outlook transferieren ... 459
12.2 E-Mail-Verkehr in Word protokollieren ... 462
12.3 Access-DB in Word verfügbar machen ... 464
12.4 Objekte in Word-Dokumente integrieren ... 470
12.5 Excel-Daten nach Word kopieren ... 473
12.6 Der Datenaustausch zwischen Access und Excel ... 477

13 Auf Fehlersuche in Office ... 481
13.1 Typische Fehlerquellen ... 481
13.2 Die Fehlerbehandlung ... 488
13.3 Allgemeine Punkte zur Programmierung ... 490

14 Tuning der VBA-Programme ... 491
14.1 Makros schneller ablaufen lassen ... 491
14.2 VBA-Abläufe sichtbar machen ... 499

15 Die Office-VBA-FAQ ... 503
15.1 Office-Animationen erstellen ... 503
15.2 Termine in den Outlook-Kalender übertragen ... 509
15.3 Excel-Auswertungen nach PowerPoint transportieren ... 511
15.4 Das Steuerelement TreeControl ... 513
15.5 Diagramme als Grafiken speichern ... 515
15.6 Das Kalendersteuerelement einsetzen ... 516
15.7 Zugriff auf Microsoft Graph programmieren ... 518

A Die Dateien zum Buch ... 521

B Anhang ... 523

Stichwortverzeichnis ... 535

1 Die Entwicklungsumgebung von VBA

Egal, in welcher Office-Komponente Sie sich gerade befinden – über die Tastenkombination Alt+F11 gelangen Sie direkt in die Entwicklungsumgebung. Da diese Entwicklungsumgebung in nahezu allen Office-Programmen identisch ist, werde ich Ihnen diese Umgebung am Beispiel von Microsoft Word 2002 beschreiben und auf die Besonderheiten der jeweiligen Office-Komponenten eingehen.

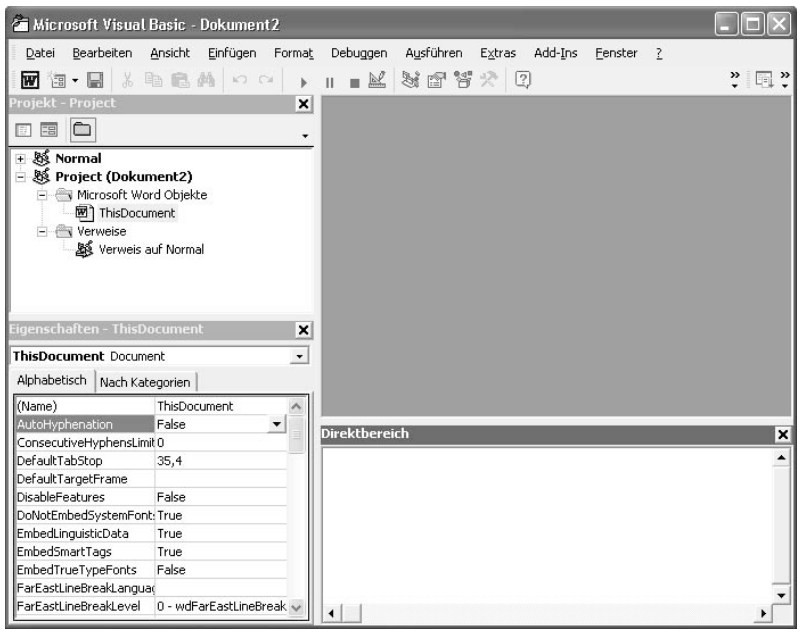

Abb. 1.1 Die Entwicklungsumgebung von Word 2002

1.1 Makros einfügen

Um ein Makro anzulegen, müssen Sie zuerst einmal ein neues Modul anlegen. Dazu wählen Sie in der Entwicklungsumgebung von Word 2002 aus dem Menü EINFÜGEN den Befehl MODUL. Auf der rechten Seite Ihrer Entwicklungsumgebung erscheint nun das so genannte Code-Fenster, in welches Sie Ihre Makros eintippen.

Legen Sie nun Ihr erstes Makro an, indem Sie in das Codefenster das Schlüsselwort `Sub` schreiben. Erfassen Sie danach nach einem Leerschritt einen beliebigen Namen für Ihr Makro, welcher allerdings keine Leer- und Sonderzeichen enthalten darf. Außerdem darf der erste Buchstaben keine Zahl sein. Nach der Benennung des Makros setzen Sie eine öffnenden sowie direkt im Anschluss eine schließende runde Klammer hinter den Namen und bestätigen mit [↵]. Die erste Zeile Ihres Makros wird sogleich mit einer End-Sub-Zeile vervollständigt. Sie haben somit den Rumpf für Ihr erstes Makro.

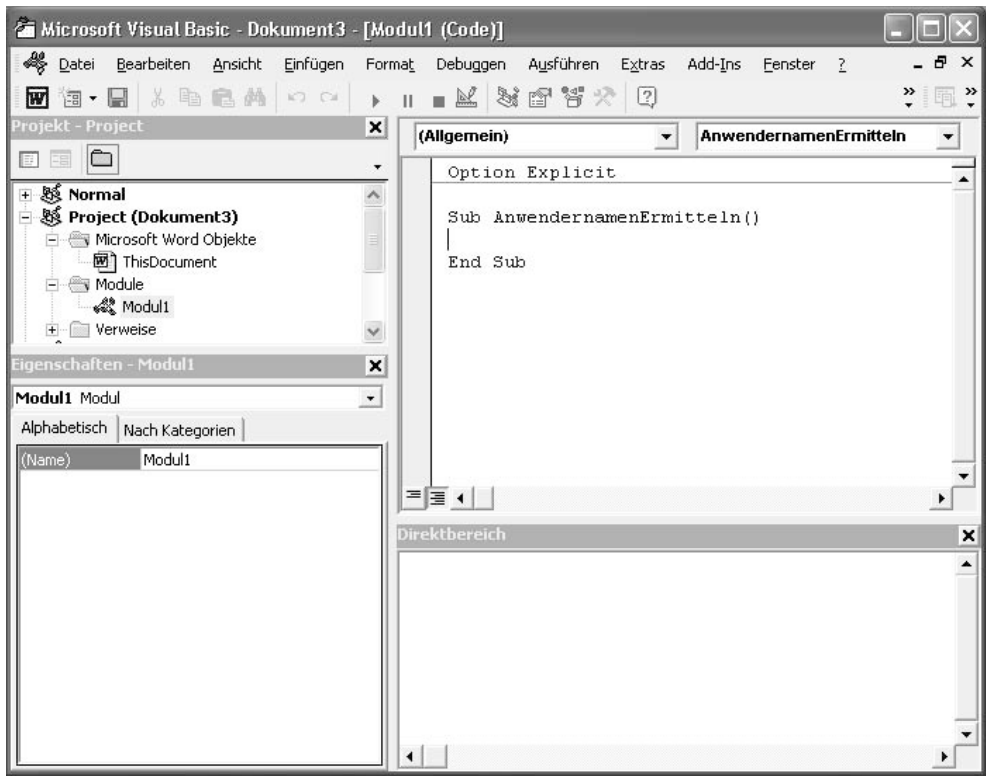

Abb. 1.2 Das erste Makro in Word

Sie können jetzt im ersten Makro ein paar Leerzeilen einfügen, wenn Sie möchten. Dazu setzen Sie den Mauszeiger ans Ende der ersten Zeile des Makros und drücken ein paar Mal die Taste [↵]. Das Makro macht im Moment noch nichts. Als Einstiegsaufgabe werden Sie nun den Namen des Anwenders, den Sie übrigens im Menü EXTRAS unter dem Befehl OPTIONEN auf der Registerkarte BENUTZERINFORMATIONEN wiederfinden, auf dem Bildschirm aus. Der Code für diese Aufgabe lautet:

```
Sub AnwendernamenErmitteln()
```

```
 MsgBox Application.UserName
End Sub
```

Mithilfe der Funktion `Msgbox` können Sie eine Meldung auf dem Bildschirm anzeigen. Die Eigenschaft `Application` gibt Ihnen Zugriff auf Ihre Word-Anwendung. Sie haben somit die Möglichkeit, auf die Eigenschaft `UserName` zurückzugreifen, die Ihnen den Namen des Anwenders ermittelt.

1.2 Makros starten

Zum Starten eines Makros haben Sie mehrere Möglichkeiten:

1. In der Entwicklungsumgebung in der Symbolleiste VOREINSTELLUNG mit einem Klick auf das Symbol SUB/USERFORM AUSFÜHREN.
2. Starten eines Makros vom Dokument aus über das Menü EXTRAS und den Befehl MAKRO/MAKROS und die Auswahl des Makros im Listenfeld mit abschließendem Klick auf die Schaltfläche AUSFÜHREN.
3. Starten eines Makros direkt aus der Entwicklungsumgebung im Codefenster, indem Sie den Mauszeiger auf die erste Zeile des Makros setzen und die Taste F5 drücken.

1.3 Den Makrorekorder einsetzen

Haben Sie bisher noch nicht viel mit VBA gemacht, wird es am Anfang nicht einfach sein, den Aufbau der einzelnen Befehle zu erkennen. Um den Einstieg in die VBA-Programmierung zu erleichtern, stehen Ihnen für die Anwendungsprogramme Word und Excel so genannte Makrorekorder zur Verfügung. Mithilfe dieser Makrorekorder können Sie automatisch Quellcode erzeugen und anhand dieser Umsetzung schnell Fortschritte in der Programmierung mit VBA machen. Dabei führen Sie einige Arbeitsschritte manuell aus und lassen den Makrorekorder im Hintergrund mitlaufen. Nach der Aufzeichnung springen Sie in den Quellcode und passen diesen an.

1.3.1 Makro aufzeichnen

Als kleine Vorarbeit legen Sie zunächst einmal ein neues, noch leeres Dokument an und erfassen einen kleinen Beispieltext. Möchten Sie diesen Beispieltext automatisch von Word selbst erzeugen lassen, dann schreiben Sie die Formel =Rand(3,2) und bestätigen mit ↵.

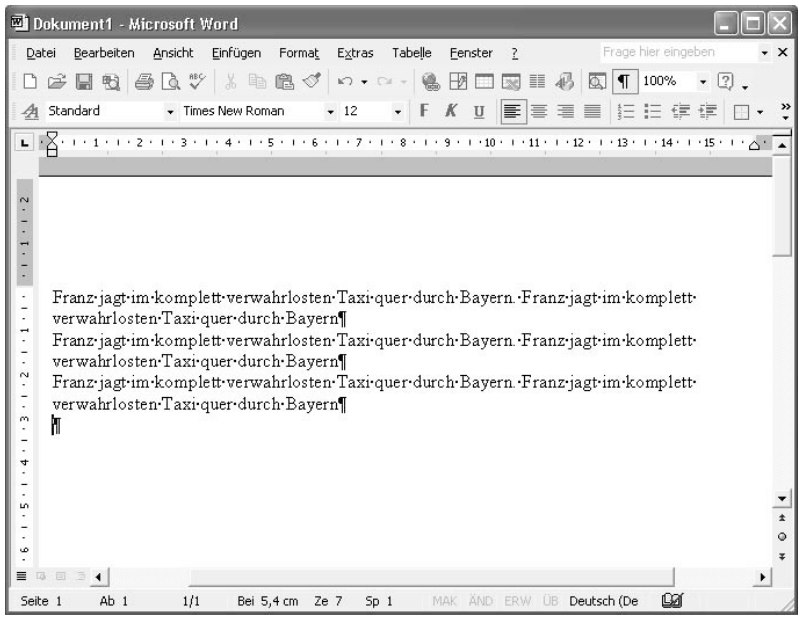

Abb. 1.3 Beispieltext automatisch erzeugen

Wenn Sie sich den Beispielsatz ansehen, dann werden Sie feststellen, dass jeder Buchstabe des Alphabets mindestens einmal darin enthalten ist. Dieser Text ist in diesem Beispiel in der Schriftart TIMES NEW ROMAN eingetragen worden. Ihre Aufgabe ist es nun, die Schriftart des Textes zu ändern. Dabei werden Sie diesen Vorgang mit dem Makrorekorder aufzeichnen lassen. Befolgen Sie dazu die nächsten Arbeitsschritte:

1. Wählen Sie aus dem Menü EXTRAS den Befehl MAKRO/AUFZEICHNEN.

Abb. 1.4 Makro aufzeichnen

2. Geben Sie dem Makro im Feld MAKRONAME einen beschreibenden Namen.
3. Sie haben die Möglichkeit, dieses Makro einem Symbol in einer Symbolleiste bzw. das Makro einer Tastenkombination zuzuweisen. Diese Möglichkeit sollten Sie jedoch beim Testen bzw. bei der Neuanlage des Makros noch nicht einsetzen.
4. Wählen Sie im Dropdownfeld MAKRO SPEICHERN Ihr aktuelles Dokument aus, sofern Sie das Makro nur für dieses Dokument einsetzen möchten. Entscheiden Sie sich hingegen für den Befehl ALLE DOKUMENTE (NORMAL.DOT), dann wird das Makro in der zentralen Dokumentdatei von Word gesichert. Damit kann das Makro für alle Dokumente eingesetzt werden. Sie sollten aber wirklich nur Makros dort hineinnehmen, die Sie wirklich für alle oder zumindest viele Dokumente gebrauchen können.
5. Starten Sie Ihre Aufzeichnung mit OK.
6. Weisen Sie jetzt dem Text im Dokument eine andere Schriftart zu, indem Sie die Tastenkombination Strg+A drücken, um den Text des ganzen Dokuments zu markieren. Wählen Sie danach aus dem Menü FORMAT den Befehl ZEICHEN. Auf der Registerkarte SCHRIFT entscheiden Sie sich für die Schrift ARIAL und bestätigen mit OK. Achtung: Wenn Sie übrigens die Schriftart über die Symbolleiste FORMAT und dem Dropdownfeld SCHRIFTART ändern, dann zeichnet der Makrorekorder diese Aktion nicht auf.
7. Beenden Sie die Aufzeichnung, indem Sie aus dem Menü EXTRAS den Befehl MAKRO/ AUFZEICHNUNG BEENDEN wählen.

Tipp:

Übrigens gibt es in Excel ebenso eine zentrale Arbeitsmappe, in der Sie Makros speichern können, die Sie auch für andere Arbeitsmappen und nicht nur für die aktuelle Mappe verwenden können. Diese Arbeitsmappe heißt PERSONL.XLS und befindet sich im Unterverzeichnis von Office im Ordner XLSTART. Standardmäßig wird diese Arbeitsmappe jedoch nicht automatisch nach der Installation von Office angelegt. Diese Arbeitsmappe können Sie aber recht schnell selbst anlegen, indem Sie ein Makro aufzeichnen und als Speicherort im Dialog MAKRO AUFZEICHNEN aus dem Dropdownfeld MAKRO SPEICHERN den Befehl PERSÖNLICHE MAKROARBEITSMAPPE auswählen.

1.3.2 Resultate des Makrorekorders ansehen

Nun aber zurück zu unserem Word-Beispiel. Sehen Sie sich jetzt den aufgezeichneten Quellcode an, indem Sie über die Tastenkombination Alt+F11 in die Entwicklungsumgebung wechseln.

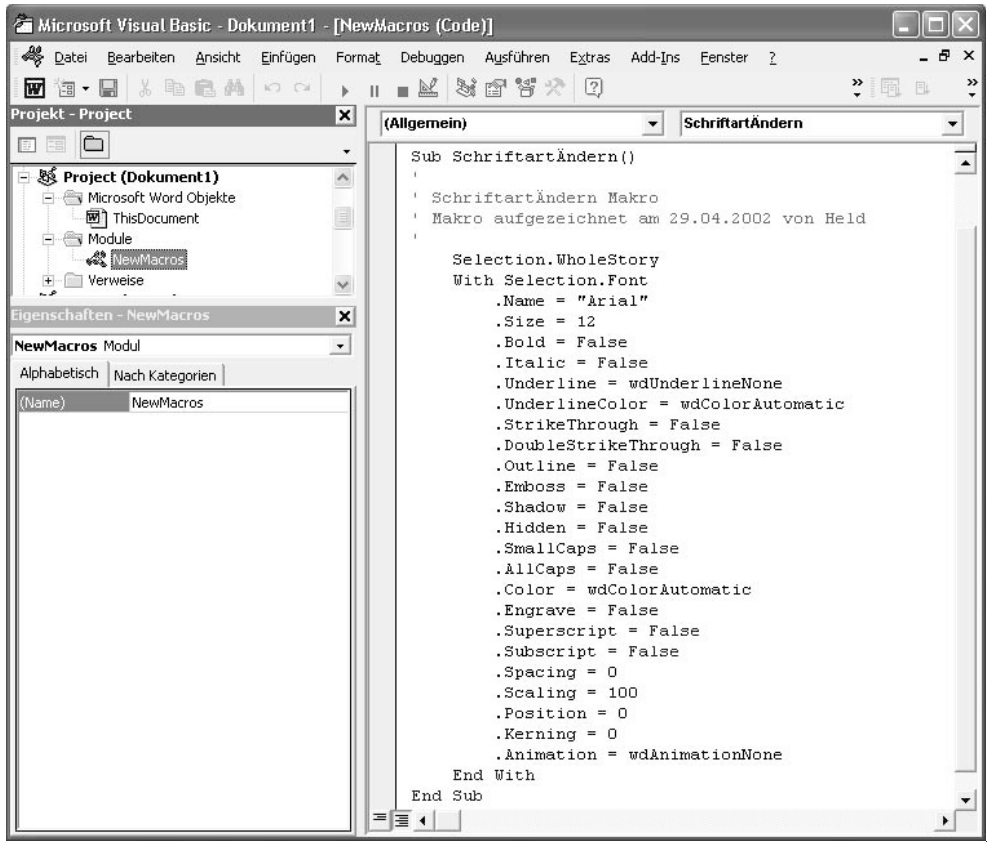

Abb. 1.5 Die Arbeit des Makrorekorders begutachten

Der Makrorekorder hat eine ganze Menge Befehle aufgezeichnet, die eigentlich gar nicht unbedingt gebraucht werden, um die gestellte Aufgabe auszuführen. Standardmäßig werden in den ersten paar Zeilen Informationen über den Anwender eingetragen, der das Makro aufgezeichnet hat sowie der Zeitpunkt der Aufnahme. Diese Informationen beginnen mit einem einfachen Apostroph und werden somit von Word als Kommentar interpretiert. Diese Kommentarzeilen werden beim Makroablauf ignoriert.

Danach wird der Text des gesamten Dokuments mithilfe der Eigenschaft `Selection` markiert, welches Ihnen das Objekt `Selection` (= Markierung) liefert. Auf dieses Objekt werden dann einige Eigenschaften gesetzt, die das Objekt näher beschreiben. So wird beispielsweise die Eigenschaft `Name` abgefragt, um die gewünschte Schriftart des Textes in der Markierung zu bestimmen. Neben der Schriftgröße, die über die Eigenschaft `Size` festgelegt wird, werden Informationen zum Schriftschnitt über die Eigenschaften `Italic` und `Bold` (kursiv und fett) vorgenommen, die für diese Aufgabe nicht benötigt und zu diesem Zweck mit dem Wert `False` ausgestattet werden. Weitere Eigenschaften wie `Underline` und `UnderlineColor` (Unterstreichung und Farbe für die Unterstreichung) werden mit so

genannten Konstanten belegt. Diese Konstanten sind in VBA festgelegt und müssen daher genauso angegeben werden.

Mithilfe der Anweisung `With` können Sie sich eine ganze Menge Schreibarbeit sparen. Über diese Anweisung können Sie eine Reihe von Anweisungen für ein nachfolgendes Objekt ausführen, ohne dieses immer wieder zu benennen. Sie benennen das Objekt `Font` im Beispiel einmal zu Beginn und setzen danach anstatt des Objekts einfach einen Punkt. Schließen Sie die Gültigkeit von `with` mit dem Befehl `End With` ab.

1.3.3 Zusatzinformationen anzeigen

Nachdem Sie den aufgezeichneten Quellcode einmal näher angesehen haben, würde ich Ihnen empfehlen, die Online-Hilfe heranzuziehen, um noch mehr Informationen über die verwendeten Befehle, Objekte und Eigenschaften zu bekommen. Setzen Sie zu diesem Zweck den Mauszeiger auf das Wort, zu dem Sie nähere Informationen wünschen und drücken die Taste F1.

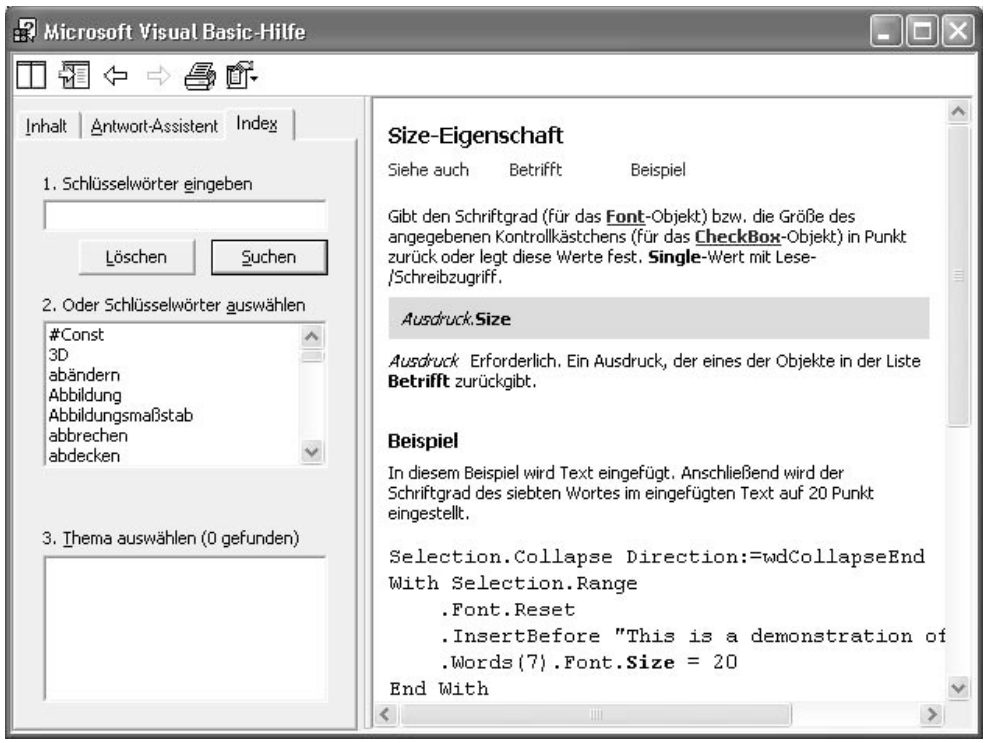

Abb. 1.6 Hilfetexte zur Eigenschaft Font

Die so aufgerufene Hilfe bietet Ihnen eine detaillierte Erklärung des Befehls sowie auch sehr oft ein Beispielmakro an, welches Sie aus der Hilfe kopieren und in Ihr Modul einfügen kön-

nen, indem Sie das Beispiel markieren und dann einen Klick mit der rechten Maustaste auf das markierte Beispiel durchführen. Sie wählen aus dem Kontextmenü den Befehl Kopieren, setzen den Mauszeiger an die gewünschte Einfügeposition in Ihrem Makro und drücken die Tastenkombination Strg+V, um das Beispielmakro einzufügen.

1.4 Die Arbeitsumgebung

Nachdem Sie Ihr erstes Makro ins Code-Fenster geschrieben und den Makrorekorder eingesetzt haben, lernen Sie jetzt die einzelnen Komponenten der Entwicklungsumgebung kennen.

1.4.1 Der Projekt-Explorer

Den Projekt-Explorer finden Sie standardmäßig am linken oberen Rand Ihrer Entwicklungsumgebung. Dort sind alle momentan geöffneten Dokumente verzeichnet.

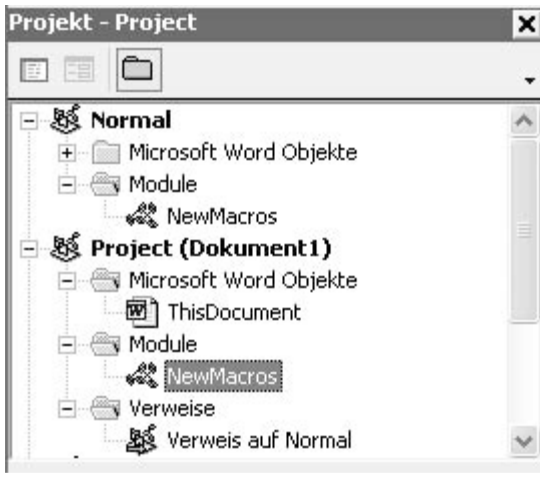

Abb. 1.7 Der Projekt-Explorer

Ganz zu Beginn sehen Sie die globale Vorlagendatei NORMAL.DOT, in der Sie Makros ablegen können, die Sie für alle Dokumente einsetzen können. Mit einem Klick auf die Plus-Symbole öffnen Sie weitere Unterordner in diesem Explorer. Hinter den Rubriken THISDOCUMENT und NEWMACROS können Sie Quellcode hinterlegen. Klicken Sie dazu einfach die entsprechende Rubrik doppelt an.

In der Rubrik `ThisDocument` haben Sie die Möglichkeit, Ereignisse einzustellen. Unter einem Ereignis versteht man einen Vorgang wie beispielsweise das Öffnen oder Schließen eines Dokuments. Dies Ereignisse können Sie abfangen und zusätzlichen Quellcode hinzufü-

gen. So haben Sie beispielsweise die Möglichkeit, beim Öffnen eines Dokuments bestimmte Vorgänge automatisch ablaufen zu lassen. So könnten Sie sich beispielsweise ein Beispiel aus der Praxis vorstellen, bei dem beim Öffnen eines Dokuments automatisch eine E-Mail an eine bestimmte Person versendet wird.

Um die Ereignisse, auch als Events bezeichnet, in Word zu aktivieren, befolgen Sie die nächsten Arbeitsschritte:

1. Wechseln Sie in die Entwicklungsumgebung von Word.
2. Führen Sie einen Doppelklick auf den Eintrag THISDOCUMENT im Projekt-Explorer durch.
3. Wählen Sie aus dem ersten Dropdownfeld den Befehl DOCUMENT.
4. Wählen Sie aus dem zweiten Dropdownfeld das gewünschte Ereignis bzw. auch mehrere nacheinander aus.

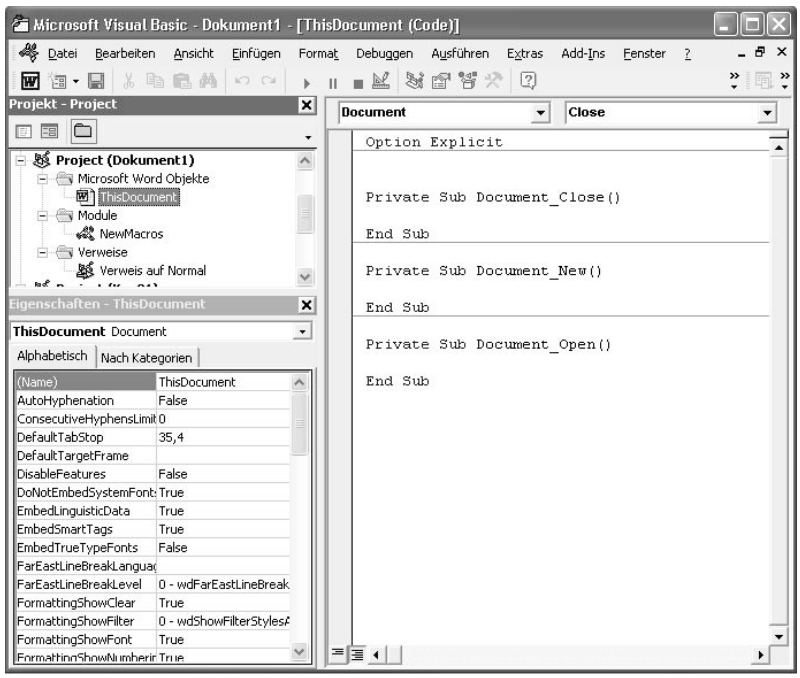

Abb. 1.8 Word-Ereignisse einstellen

Das Ereignis `Document_Close` wird dann ausgeführt, wenn Sie Ihr Dokument schließen. Das Ereignis `Document_New` wird gestartet, wenn Sie ein neues Dokument einfügen. Das Ereignis `Document_Open` wird automatisch ausgeführt, wenn Sie Ihr Dokument öffnen.

Innerhalb dieser Ereignisse können Sie zusätzliche VBA-Anweisungen angeben, die beim entsprechenden Ereignis mit ausgeführt werden sollen.

Tipp:

In Excel gibt es eine ganze Reihe weitere Ereignisse, die Sie entweder für die komplette Arbeitsmappe oder auch nur für spezielle Tabellen einstellen können. Sie finden die Ereignisse in Excel, die für die ganze Arbeitsmappe Gültigkeit haben, hinter der Rubrik DIESEARBEITSMAPPE. Möchten Sie hingegen ein Ereignis nur für eine einzelne Tabelle einstellen, dann klicken Sie die gewünschte Tabelle im Projekt-Explorer doppelt an und wählen das Ereignis aus dem zweiten Dropdownfeld aus.

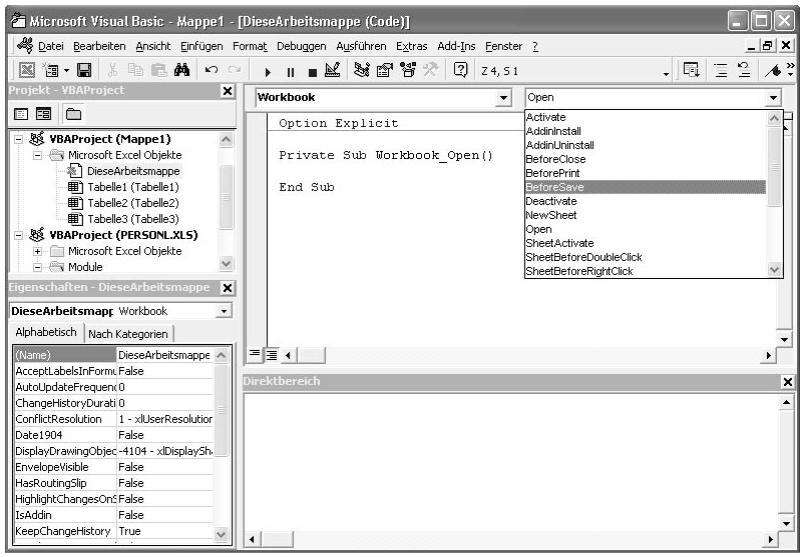

Abb. 1.9 Die Arbeitsmappenereignisse von Excel

1.4.2 Das Eigenschaftsfenster

Das Eigenschaftsfenster finden Sie standardmäßig unterhalb des Projekt-Explorers. Es enthält nähere Festlegungen zum Dokument. Diese Eigenschaften können Sie entweder in diesem Fenster einstellen oder auch per VBA-Code im Code-Fenster programmieren.

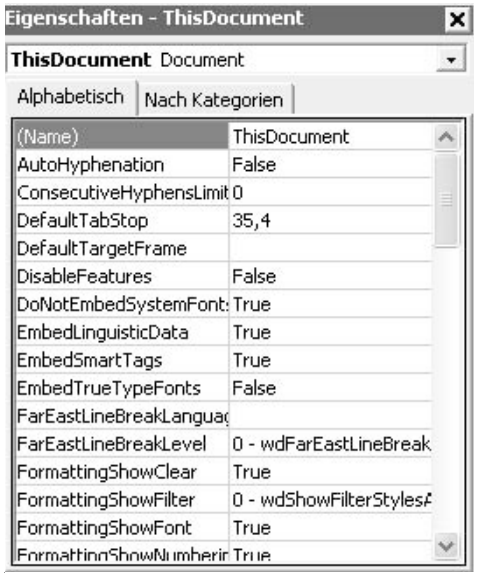

Abb. 1.10 Das Eigenschaftsfenster

1.4.3 Das Code-Fenster

Im Code-Fenster erfassen und bearbeiten Sie Ihre VBA-Makros. Sollte dieses Fenster noch nicht eingeblendet sein, dann wählen Sie aus dem Menü ANSICHT den Befehl CODE.

Das Code-Fenster enthält folgende Objekte:

- Dropdownfeld OBJEKT: Hier werden die Namen der markierten Objekte angezeigt. Klikken Sie auf den Pfeil rechts neben dem Listenfeld, um eine Liste aller mit dem Formular verknüpften Objekte anzuzeigen.
- Dropdownfeld PROZEDUR: Hier werden alle Ereignisse und Prozeduren aufgelistet, die dem im Objekt-Feld angezeigten Objekt zugeordnet sind. Bei der Auswahl eines Ereignisses wird die mit diesem Ereignisnamen verknüpfte Ereignisprozedur im Code-Fenster angezeigt.

Alle Prozeduren werden übrigens in alphabetischer Reihenfolge im Dropdownfeld angezeigt. Dies erleichtert die Suche nach bestimmten Modulen.

Hinweis:

Wenn Sie auf den rechten oberen Rand des Codefensters sehen, erkennen Sie einen Fensterteiler. Damit können Sie das Code-Fenster in zwei horizontale Bereiche unterteilen, in denen separate Bildläufe durchgeführt werden können. So können Sie unterschiedliche Teile des Codes gleichzeitig angezeigen. Die Informationen, die im Feld OBJEKT und Prozedur angezeigt werden, beziehen sich auf den Code in dem Fenster, das gerade aktiv ist. Mit einem Doppelklick auf den Fensterteiler wird ein Fenster geschlossen.

1.4.4 Das Direktfenster

Das Direktfenster setzen Sie ein, um beispielsweise bestimmte Inhalte von Variablen im Direktfenster ausgeben zu lassen, um beispielsweise das Ergebnis eines Makros zu testen. Erfassen Sie zunächst einmal in einer beliebigen Office-Anwendung das folgende Beispiel:

```
Sub Schleifendurchläufe()
Dim i_Zähler As Integer
For i_Zähler = 1 To 15
  Debug.Print "Schleifendurchlauf: " & i_Zähler
Next i_Zähler
End Sub
```

Listing 1.1: Schleifendurchläufe aufzeichnen

Das Makro wird eine Schleife genau fünfzehnmal durchlaufen. Bei jedem Schleifendurchlauf wird mithilfe des Befehls `Debug.Print` ein Eintrag ins Direktfenster geschrieben, der den aktuellen Schleifendurchlauf festhält. Setzen Sie die Einfügemarke in die erste Zeile des Makros und drücken Sie die Taste F5, um das Makro zu starten. Kontrollieren Sie jetzt einmal das Ergebnis im Direktfenster. Dazu wählen Sie den Menübefehl ANSICHT/DIREKTFENSTER oder drücken die Tastenkombination Strg + G.

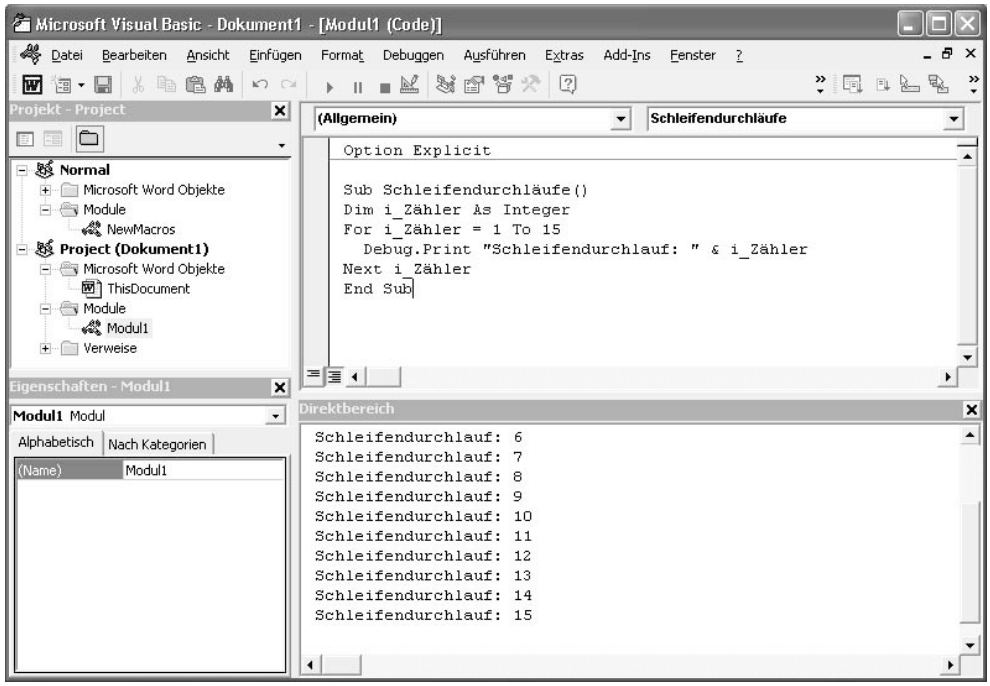

Abb. 1.11 Bildunterschrift eintragen

Möchten Sie Ihren Quellcode schrittweise durchlaufen, dann setzen Sie beispielsweise die Einfügemarke in die Zeile, bis zu der Sie den Code durchlaufen möchten, und drücken Sie die Taste F9. Auf diese Weise haben Sie einen Haltepunkt im Quellcode gesetzt. Der Code wird jetzt bis zu dieser braun gekennzeichneten Zeile durchlaufen und stoppt genau an dieser Position. Nun können Sie prüfen, ob das Makro auch korrekt funktioniert hat. Wenn alles soweit stimmt, können Sie durch die Taste F5 dafür sorgen, dass das Makro bis zum Ende durchläuft. Sollte etwas bei dem Makro nicht stimmen, brechen Sie es ab, indem Sie aus dem Menü AUSFÜHREN den Befehl ZURÜCKSETZEN wählen.

1.4.5 Das Überwachungsfenster

Eine nützliche Funktion können Sie einsetzen, wenn Sie das Überwachungsfenster einblenden. Wählen Sie dazu aus dem Menü ANSICHT den Befehl ÜBERWACHUNGSFENSTER. Sie haben jetzt beispielsweise die Möglichkeit zu überprüfen, wann sich eine bestimmte Variable ändert. Genau dann soll der Makroablauf unterbrochen werden. Im folgenden Beispiel wird eine Schleife genau fünfzehnmal durchlaufen. Bei jedem Schleifendurchlauf wird die Variable

i_zähler verändert. Nach dem fünften Schleifendurchlauf soll das Makro stoppen. Erfassen Sie zunächst das folgende Makro:

```
Sub Schleifendurchläufe()
Dim i_Zähler As Integer
For i_Zähler = 1 To 15
  Debug.Print "Schleifendurchlauf: " & i_Zähler
Next i_Zähler
End Sub
```

Stellen Sie jetzt die Überwachung ein, indem Sie wie folgt vorgehen:

1. Blenden Sie das Überwachungsfenster ein, indem Sie aus dem Menü ANSICHT den Befehl ÜBERWACHUNGSFENSTER auswählen.
2. Wählen Sie aus dem Menü DEBUGGEN den Befehl ÜBERWACHUNG HINZUFÜGEN.

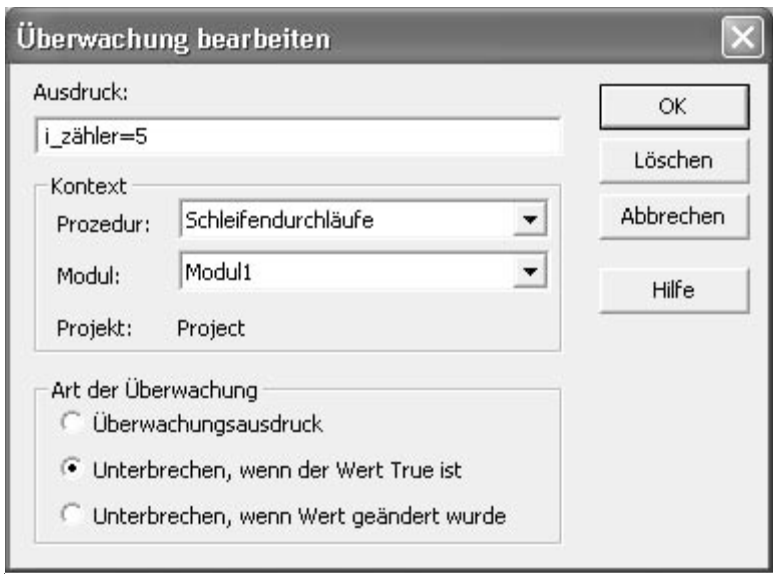

Abb. 1.12 Überwachung hinzufügen

3. Geben Sie im Feld AUSDRUCK den Überwachungsausdruck I_ZÄHLER=5 ein.
4. Stellen Sie sicher, dass im Dropdownfeld PROZEDUR das richtige Makro eingestellt ist.
5. Aktivieren Sie die Option UNTERBRECHEN, WENN DER WERT TRUE IST.
6. Bestätigen Sie Ihre Einstellung mit OK.
7. Starten Sie nun das Makro.

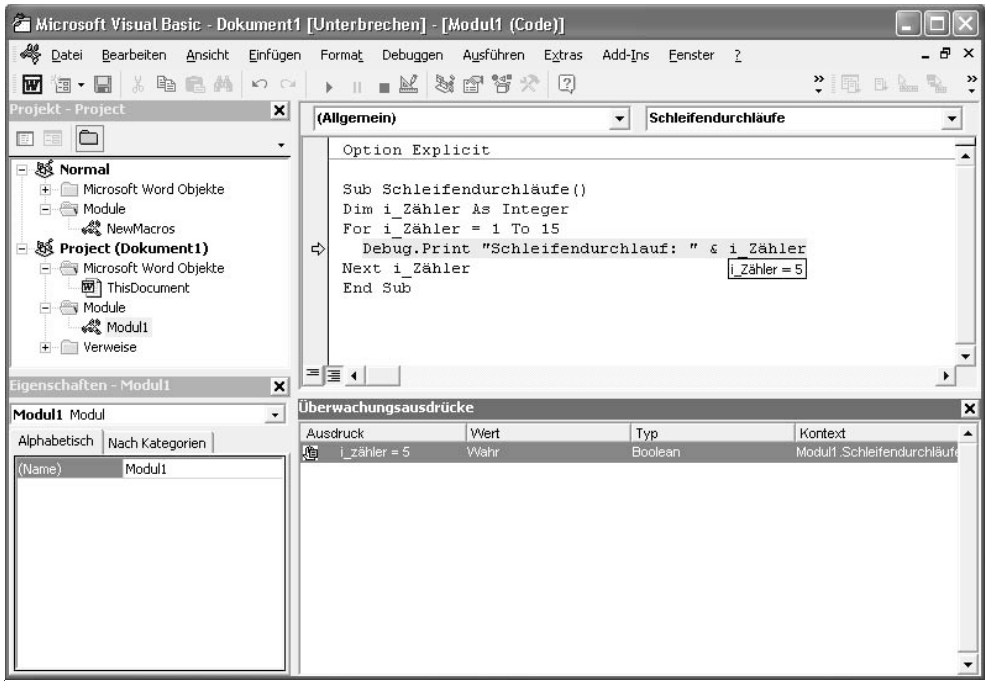

Abb. 1.13 Makro bei bestimmter Bedingung stoppen

Das Makro läuft genau so lange, bis es fünfmal die Schleife durchlaufen hat. Dann entscheiden Sie, wie weiter verfahren wird. Sie können entweder das Makro fortsetzen, indem Sie die Taste F5 drücken oder das Makro abbrechen, indem Sie aus dem Menü AUSFÜHREN den Befehl ZURÜCKSETZEN wählen.

1.4.6 Das Lokal-Fenster

Das Lokal-Fenster wird in der Entwicklungsumgebung standardmäßig nicht angezeigt. Über das Menü ANSICHT können Sie dieses Fenster jedoch einblenden. Das Lokal-Fenster zeigt alle deklarierten Variablen in der aktuellen Prozedur und deren Werte an. Sie haben damit die Möglichkeit, die Werte von Variablen übersichtich zu prüfen.

Wenden Sie nun das Lokal-Fenster anhand unseres letzten Beispiels an. Dabei gehen Sie wie folgt vor:

1. Blenden Sie das Lokal-Fenster ein, indem Sie aus dem Menü ANSICHT den Befehl LOKAL-FENSTER wählen.
2. Setzen Sie den Mauszeiger auf die Zeile `next i_zähler` und drücken Sie die Taste F9, um einen Haltepunkt zu setzen.

3. Setzen Sie den Mauszeiger auf die erste Zeile Ihres Makros und drücken Sie die Taste F5, um das Makro zu starten.

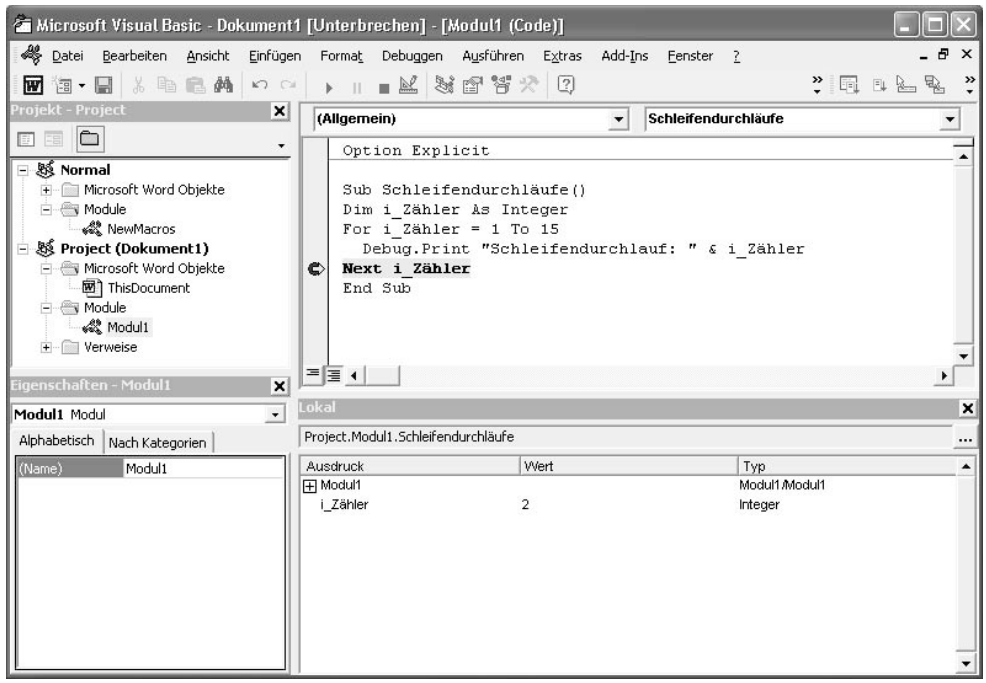

Abb. 1.14 Variablen mit dem Lokalfenster überwachen

Das Makro stoppt nun genau nach dem ersten Schleifendurchlauf. Sie können sich den augenblicklichen Wert der Variablen nun im Lokal-Fenster ansehen.

1.5 Wertvolle Helfer bei der Programmierung

Um schnellstmöglich programmieren zu können, stehen Ihnen bestimmte Hilfsmittel in der Entwicklungsumgebung zur Verfügung. So können Sie beispielsweise mithilfe von Symbolleisten noch schneller Standardbefehle anwenden.

1.5.1 Die Symbolleiste Bearbeiten

Diese Symbolleiste ist standardmäßig in der Entwicklungsumgebung nicht eingeblendet. Holen Sie das jetzt nach, indem Sie im Menü ANSICHT den Befehl SYMBOLLEISTEN/BEARBEITEN aufrufen.

Abb. 1.15 Die Symbolleiste Bearbeiten

Die wichtigsten Funktionen werden nun erklärt. Wenn Sie den Mauszeiger auf ein beliebiges Symbol setzen, wird der Name der Funktion in einem kleinen QuickInfo-Fenster angezeigt.

Einzug vergrößern bzw. verkleinern

Mit der Funktion EINZUG VERGRÖSSERN können Sie einzelne Zeilen oder auch mehrere Zeilen blockweise nach rechts einrücken. Dies macht den Programmcode leichter lesbar. Analog zu dieser Funktion können Sie mit der Funktion EINZUG VERKLEINERN eingerückte Programmteile wieder nach links rücken und pro Klick jeweils den markierten Text um einen Tabstopp nach links versetzen.

Haltepunkt ein/aus

Setzen Sie den Mauszeiger auf die Zeile im Code, bis zu der das Makro laufen soll. Wenn Sie nun das Makro starten, stoppt es genau an diesem Haltepunkt. Hiermit können Sie Programm-Zwischenstände überprüfen. Sie werden somit auch erstklassig bei der Fehlersuche unterstützt. Einen Haltepunkt können Sie auch setzen, wenn Sie die Taste F9 drücken.

Block auskommentieren

Hinterlegen Sie ausreichend Kommentare in Ihren Makros. Es fällt Ihnen dadurch später leichter, die einzelnen Befehle nachzuvollziehen. Auch Änderungen am Makro selbst können auf diese Art und Weise festgehalten werden.

Um einen Kommentar zu hinterlegen, haben Sie mehrere Möglichkeiten:

- Geben Sie ein einfaches Anführungszeichen (') vor dem eigentlichen Befehl oder Text ein oder
- erfassen Sie etwas altertümlicher die Anweisung Rem, gefolgt von einem Leerzeichen und dem Befehl oder Text.

Der Kommentar nimmt in beiden Fällen standardmäßig die Schriftfarbe GRÜN an. Diese so kommentierten Zeilen werden beim Ablauf des Makros nicht ausgewertet. Sie können ganze Kommentarzeilen anlegen oder auch innerhalb einer Zeile am Ende einen Kommentar anfügen. Möchten Sie innerhalb einer Zeile einen Kommentar im Anschluss eines Befehls erfas-

sen, fügen Sie nach dem eigentlichen Befehl ein einfaches Anführungszeichen (') ein und schreiben Ihren Kommentar dazu.

Hinweis:

Möchten Sie nicht nur einzelne Zeilen, sondern ganze Textblöcke in Kommentar setzen, dann markieren Sie den Bereich, den Sie auskommentieren möchten, und klicken auf das Symbol BLOCK AUSKOMMENTIEREN. Um einzelne Zeilen oder auch einen ganzen Block wieder zu aktivieren, markieren Sie die entsprechende(n) Zeile(n) und klicken auf das Symbol AUSKOMMENTIERUNG DES BLOCKS AUFHEBEN.

Eigenschaften/Methoden anzeigen

Möchten Sie ganz schnell die zur Verfügung stehenden Eigenschaften und Methoden einer Anweisung einsehen, markieren Sie den gewünschten Befehl im Code und klicken auf das Symbol EIGENSCHAFTEN/METHODEN ANZEIGEN.

In der folgenden Abbildung werden für das Objekt `Selection` die zur Verfügung stehenden Methoden und Eigenschaften in einem Kontextmenü angezeigt.

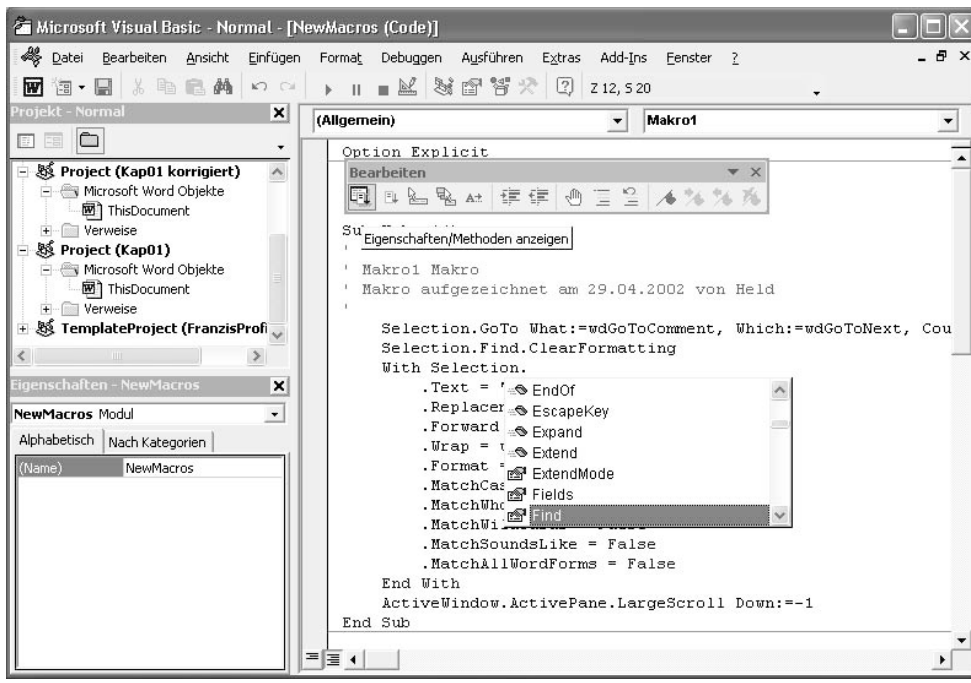

Abb. 1.16 Verfügbare Eigenschaften und Methoden anzeigen

Übrigens können Sie die zur Verfügung stehenden Methoden und Eigenschaften auch über die Tastenkombination Strg+J abrufen. Geben Sie davor das Objekt ein, zudem Sie die verfügbaren Methoden und Eigenschaften abrufen möchten.

Parameterinfo

Wenn Sie einen Befehl im Code-Bereich eingeben und danach die Taste Leer drücken, wird dynamisch ein QuickInfo-Fenster angezeigt, in dem Sie die zur Verfügung stehenden Argumente ablesen können. Die QuickInfo hilft Ihnen schnell weiter, die Befehle in der richtigen Syntax und mit den zur Verfügung stehenden Argumenten einzugeben. Möchten Sie bereits erfasste Befehle mit weiteren Argumenten bestücken, setzen Sie die Einfügemarke in die Anweisung (nicht auf den Befehl) und klicken in der Symbolleiste BEARBEITEN auf das Symbol PARAMETERINFO.

In der folgenden Abbildung wird für die Methode GoTo des Objekts Selection die erforderliche Syntax ermittelt und in einem QuickInfo-Fenster angezeigt.

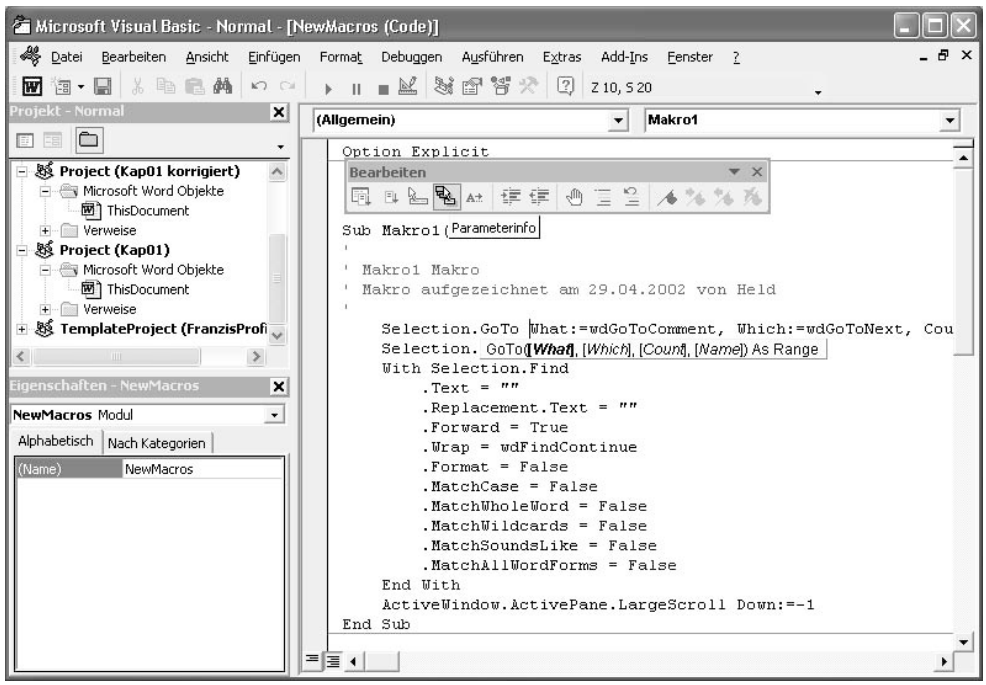

Abb. 1.17 Parameterinfos anzeigen

Konstanten anzeigen

Einige VBA-Befehle bieten Ihnen so genannte Konstanten an. Solche Konstanten finden Sie beispielsweise beim Befehl MsgBox. Dieser Befehl zeigt ein normales Meldungsfenster mit