VBA kennen lernen

1

In diesem Kapitel

- Anwendungsmöglichkeiten für VBA-Programme (Visual Basic für Applikationen) finden
- Herausfinden, wo außer in Microsoft Office VBA noch drinsteckt
- Die integrierte Entwicklungsumgebung (IDE, Integrated Development Environment) VBA
- Ein einzeiliges Programm schreiben

Haben Sie schon einmal in einem Gespräch über eine von Ihnen verwendete Anwendung jemandem gesagt, dass Sie den Eindruck haben, der Anbieter, der die Anwendung erstellt hat, habe keine Ahnung? Die Anwendung sei einfach zu schwierig oder zu zeitintensiv in der Handhabung, da auf die Features nur schwer zugegriffen werden kann. Ich wette, in einigen Fällen haben Sie Features gesehen, die *beinahe* das getan haben, was Sie wollten …, aber eben nur beinahe. Etwas, das nur beinahe funktioniert, ist frustrierend, und viele von uns haben sich in solch einer Situation schon eine Lösung des Problems gewünscht.

Irgendwann einmal erwachte einer der klugen Köpfe bei Microsoft aus seinem vom vielen Koffein verursachten Koma und brachte etwas zustande, das all diese Probleme und noch einiges mehr löst: Visual Basic für Applikationen (VBA). VBA ist eine einfache Programmiersprache. Wenn Sie VBA verwenden, funktionieren die Dinge, wie Sie es möchten: Sie können Ihre Anwendungen an Ihre Bedürfnisse und Erwartungen anpassen. Sie sind kein Sklave der Anbieterwünsche mehr. Wenn Sie eine Anwendung verwenden, die VBA unterstützt, können Sie neue Features hinzufügen, zum Beispiel eine Funktion zum automatischen Schreiben von Briefen oder eine Funktion zur Berechnung spezieller Formeln, um die Dinge an Ihre Vorstellungen und Wünsche anzupassen. Somit werden aus Anwendungen Ihre benutzerdefinierten Anwendungen, und Sie müssen sich nicht mehr mit dem begnügen, von dem ein Anbieter meint, das Sie es brauchen.

VBA ist in vielen Anwendungen integriert, so auch in den Microsoft Office-Programmen. Mit VBA schreiben Sie Programme, mit denen Aufgaben automatisch ausgeführt werden, oder Sie ändern die Anwendungsumgebung. Viele Leute denken, sie könnten keine Programme schreiben, auch wenn sie noch so einfach sind. Wenn Sie dieses Buch gelesen haben, werden Sie wissen, dass jeder ein Programm schreiben kann. Sie werden übrigens Ihr erstes Programm in diesem Kapitel schreiben. Zuerst werden Sie natürlich das »Sesam öffne dich« zum Aufrufen des VBA-Editors kennen lernen. Der VBA-Editor unterscheidet sich nur geringfügig von den Textverarbeitungsprogrammen, die Sie bereits kennen. Im Laufe dieses Kapitels werden Sie einige interessante Anwendungsmöglichkeiten für VBA kennen lernen und feststellen, wie viele Anwendungen Sie mit VBA anpassen können.

VBA wird mit Office ausgeliefert

Viele Menschen haben mich gefragt, ob VBA wirklich zusammen mit Office ausgeliefert wird. Die Antwort lautet natürlich ja. Alle Office-Produkte unterstützen VBA und Sie können mit VBA sehr viele Aufgaben durchführen. Viele dieser Aufgaben erscheinen Ihnen vielleicht momentan noch unmöglich. Ältere Office-Versionen boten eine sehr einfache Methode, um auf VBA zuzugreifen. Bisher mussten Sie im Menü nur Extras|Makro|Visual Basic-Editor auswählen, um den VBA-Editor aufzurufen, in den Sie VBA-Befehle eingeben und zur späteren Wiederverwendung speichern konnten.

Einer der Gründe für diesen Abschnitt ist, dass Microsoft nicht mehr der Meinung ist, dass der durchschnittliche Anwender schlau genug ist, um mit VBA zu arbeiten. Ich finde es bemerkenswert, dass Microsoft die Produkte immer häufiger auf den dümmsten anzunehmenden Benutzer zuschneidert, dabei aber die Bedienung gleichzeitig verkompliziert. Aber sie schaffen es. Neuere Office-Versionen verstecken VBA. Wenn Sie ein Produkt wie Word 2007 verwenden, müssen Sie zuerst nach VBA suchen, bevor Sie VBA verwenden können. Verschwenden Sie keine Zeit damit, VBA in der neuen Multifunktionsleiste zu suchen – Sie werden es dort nicht finden. Die folgenden Schritte helfen Ihnen dabei, das versteckte VBA in Ihrem Word, Excel und PowerPoint wiederzufinden.

1. **Klicken Sie auf die Schaltfläche Office in Word, Excel oder PowerPoint und klicken Sie im Menü Word-Optionen, Excel-Optionen oder PowerPoint-Optionen an.**

   Der Word-, Excel- oder PowerPoint-Optionsdialog wird angezeigt (vergleiche Abbildung 1.1). Alle drei Optionsdialoge sind ähnlich und haben die VBA-Option an derselben Stelle.

2. **Wählen Sie das Kontrollkästchen Entwicklerregisterkarte in der Multifunktionsleiste anzeigen aus.**

3. **Klicken Sie auf OK.**

   Bei Word, Excel oder PowerPoint öffnet sich die Registerkarte Entwicklertools (vergleiche Abbildung 1.2), die die VBA-Optionen enthält, die in diesem Buch beschrieben werden.

Abhängig davon, welches Office 2007-Produkt Sie verwenden, finden Sie die VBA-Optionen an unterschiedlichen Stellen. Sie wissen bereits, dass Word, Excel und PowerPoint diese Optionen auf der Registerkarte Entwicklertools in der Multifunktionsleiste darstellen. Wenn Sie mit Access arbeiten, finden Sie die VBA-Schaltflächen auf der Registerkarte Datenbanktools. Die Schaltflächen sehen genauso aus, wie die Schaltflächen in Abbildung 1.2. Outlook verwendet die neue Multifunktionsleiste nicht, daher finden Sie VBA wie gewohnt im Menü Extras|Makros.

Eine andere Sache, die sich mit Einführung der Multifunktionsleiste ändert, ist, dass Sie Werkzeugleisten (weil es ja gar keine Werkzeugleisten mehr gibt!) nicht mehr mit dem Rechtsklick anklicken und aus dem Menü Anpassen auswählen können, um neue Menübefehle hinzuzufügen. Die Multifunktionsleiste kann man nicht ohne eigene Programmierung ändern. In Kapitel 12 ist beschrieben, wie man eine neue Schaltfläche in die Multifunktionsleiste einbaut. Alle Werkzeugleisten, die Sie unter VBA selbst erstellt haben, erscheinen nun auf der Registerkarte

Add-Ins der Multifunktionsleiste, das heißt, selbst programmierte Werkzeugleisten haben etwas von ihrer Effektivität unter Office 2007 verloren.

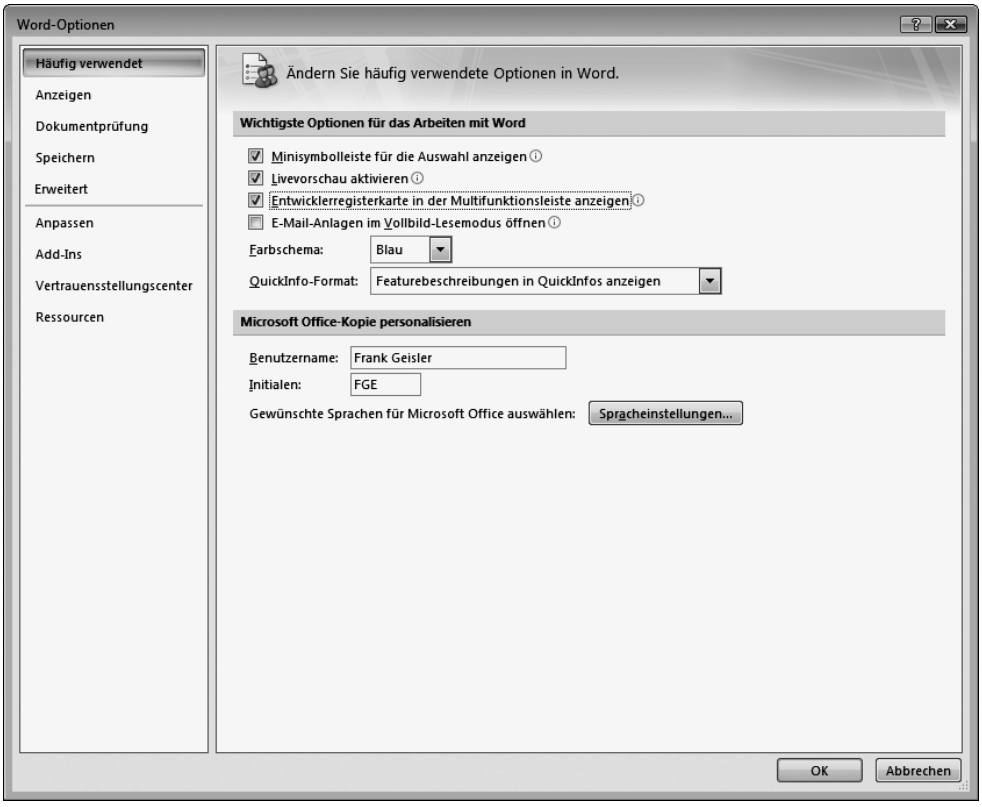

Abbildung 1.1: Das Dialogfeld Word-Optionen hilft Ihnen, Word an Ihre persönlichen Anforderungen anzupassen.

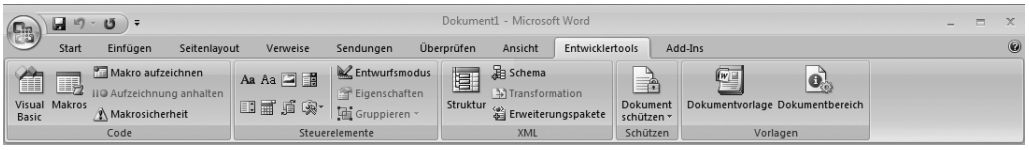

Abbildung 1.2: Die Registerkarte Entwicklertools enthält die Funktionen, die Sie früher im Menü Extras gefunden haben.

Interessanterweise hat Microsoft weder OneNote 2007, Publisher 2007, Visio 2007 noch Project 2007 auf der neuen Multifunktionsleiste aktualisiert. Daher verwenden Sie in diesen Programmen die Methode, um auf VBA zuzugreifen, die Sie auch bisher verwendet haben: das Menü Extras|Makro. Des Weiteren fehlen bei diesen Programmen viele der neuen Funktionen, die Microsoft in die Kernprodukte eingebaut hat.

VBA: Nicht nur für Programmierer

Eines der Dinge, die Sie sich überlegen sollten, ist, warum Sie VBA verwenden möchten. Ich weiß, dass einige von Ihnen wahrscheinlich einfach nur so an VBA interessiert sind. Die meisten werden jedoch einen guten Grund dafür haben, warum sie sich die Zeit dafür nehmen. Sie müssen wissen, für welche Aufgaben Sie VBA verwenden können. VBA wird weder den Müll runtertragen, noch Ihre Wäsche bügeln, aber Sie können mit VBA das Schreiben bestimmter Briefe automatisieren. In den nachfolgenden Abschnitten werden Sie einige der Dinge kennen lernen, die ich mit VBA getan habe. Und so wie ich Sie einschätze, werden Ihnen noch viel mehr Dinge einfallen.

Dokumente automatisieren

Ich hasse Briefeschreiben. Besonders hasse ich es, wenn ein Brief mehr oder weniger dieselben Informationen wie ein bereits geschriebener Brief enthält. Manchmal lassen sich Briefe mithilfe von Seriendruck automatisieren. Bei individuellen Briefen hilft das normalerweise jedoch nicht wirklich weiter. In Fällen wie diesen richte ich ein Formular mit Informationen ein, die in einigen Briefen enthalten sein sollen, in anderen jedoch nicht. Ich aktiviere die Elemente, die ich für den aktuellen Brief brauche, und VBA schreibt den Brief automatisch für mich. Geheimtipps zu meinem automatisierten Brief finden Sie in Kapitel 13.

Dokumente lassen sich nicht nur in Textverarbeitungsprogrammen automatisch erstellen. Sie können auch eine Tabellenkalkulation automatisieren. Ich habe mehrere Programme für Excel geschrieben. Wenn beispielsweise ein neuer Kunde auf mich zukommt, klicke ich auf eine Schaltfläche, und VBA erstellt für mich alle erforderlichen Kundeneinträge in Excel. Da Excel die Aufgabe jedes Mal gleich ausführt, kann ich nichts vergessen, und jedem Kunden sind dieselben Serviceleistungen mit derselben Qualität gewiss. Techniken zum Erstellen automatisierter Excel-Arbeitsblätter finden Sie in Kapitel 14.

Und wenn Sie im Textverarbeitungsprogramm oder im Tabellenkalkulationsprogramm erstellte Daten ins Internet stellen möchten, kann VBA diesen Prozess nahezu vollständig automatisieren. Kapitel 16 enthält alle wichtigen Informationen, die Sie kennen müssen, um Daten von einem Microsoft Office-Produkt in ein anderes zu verschieben, ohne dabei Änderungen oder Neuformatierungen vornehmen zu müssen.

Die Benutzeroberfläche einer Anwendung anpassen

Es gibt manche Anwendungsfunktionen, die einfach nur nerven. Diese Funktionen ließen sich sicherlich deaktivieren, wenn sie zu nervig werden. Aber das ist oft keine wirkliche Lösung, wenn die jeweilige Funktion für die Arbeit benötigt wird. Erstellen Sie mit VBA eine neue Version der Funktion mit allem, was Sie brauchen, und ohne das, was Sie nervt. Mir hat beispielsweise die Wörterzählfunktion von Word nie so richtig gefallen. Also habe ich für diese Aufgabe mein eigenes Programm geschrieben. In Kapitel 12 gebe ich einige meiner Geheimnisse zum Zähmen widerspenstiger Benutzeroberflächen preis.

Es ist ganz einfach, die Benutzeroberfläche einer Anwendung an eigene Wünsche anzupassen. Sie können benutzerdefinierte Menüsysteme oder Symbolleisten erstellen. Sie können einige Elemente der Benutzeroberfläche in ein Formular ausgliedern oder völlig entfernen. Und jede Änderung, die Sie an einer Benutzeroberfläche vornehmen möchten, lässt sich mit ziemlicher Sicherheit mit VBA vornehmen. Darüber hinaus sind Sie nicht an nur eine Benutzeroberfläche gebunden. Sie können Programme erstellen, mit deren Hilfe Sie die Benutzeroberfläche je nach der Aufgabe wechseln können. Ich habe beispielsweise ein Programm zum Wechseln zwischen den Schreibansichten für Buch-, Artikel- und Kundendokumente. In Kapitel 7 finden Sie eine Menge interessanter Möglichkeiten für die Verwendung von Formularen.

Berechnungen durchführen

Sehr häufig werden spezielle Anwendungen für komplexe Berechnungen verwendet. Mit den Produkten von Microsoft Office können Sie viele Arten von Formeln erstellen. Manchmal müssen Sie jedoch die Daten ändern, bevor Sie sie verwenden können, oder die Berechnung je nach dem Wert einer oder mehrerer Eingaben unterschiedlich durchführen. Immer dann, wenn eine Berechnung für eine einfache Formel zu kompliziert wird, können Sie sie mit VBA vereinfachen. VBA löst Berechnungsprobleme nicht in einem großen Schritt, sondern in vielen kleinen. In Kapitel 4 und 14 zeige ich Ihnen zahlreiche Möglichkeiten zum Arbeiten mit Berechnungen.

Manchmal bedeutet die Zahl, die sich aus einer Berechnung ergibt, nicht viel. Es ist einfach nur eine Zahl – bis jemand eine Entscheidung trifft. Einige Entscheidungen sind einfach, wiederholen sich jedoch ständig. In Kapitel 5 lernen Sie Methoden kennen, die Ihre Anwendung nutzen kann, um mit VBA Entscheidungen automatisch zu treffen. Mit intelligenten Anwendungen sparen Sie viel Zeit, die Sie zum Spielen von Solitär nutzen können.

Daten aus Datenbanken abrufen

Viele Informationen, angefangen bei meiner Filmesammlung bis hin zu einer Liste mit meinen Kunden, für die ich regelmäßig arbeite, speichere ich in Access. Datenbanken werden dazu verwendet, Daten zu speichern, obwohl das nicht weiterhilft, wenn Sie die Daten nicht abrufen können. Mit VBA können Sie aus Ihrer Datenbank Daten in der gewünschten Form abrufen. So können Sie diese Daten beispielsweise zum Überprüfen in einem Formular anzeigen oder mithilfe derselben Daten einen Bericht erstellen.

Ich liebe Datenbanken, weil sie die flexibelste Möglichkeit zum Speichern von sich wiederholenden Daten wie Kundenlisten oder jede andere Art von Liste, die Sie sich vorstellen können, bieten. Denken Sie nicht, dass Datenbanken so kompliziert sind, dass Sie nie verstehen werden, wie sie funktionieren. Die meisten Datenbanken lassen sich recht einfach nutzen. Für den Zugriff auf diese Datenbanken ist lediglich ein bisschen einfach zu verstehender VBA-Code erforderlich. In Kapitel 15 erfahren Sie alles, was Sie zum Arbeiten mit Datenbanken wissen müssen.

VBA stellt sogar Möglichkeiten zum Erstellen temporärer Datenbanken für Listen, die Sie nur heute brauchen, zur Verfügung. Damit können Sie eine Menge Zeit sparen und den Computer trotzdem dazu bringen, Ihnen die Arbeit abzunehmen. Diese Möglichkeiten lernen Sie in Kapitel 9 kennen.

Anwendungen mit neuen Features erweitern

Angesichts all der Features in Anwendungen müsste man eigentlich annehmen, dass keine Wünsche mehr offen bleiben. Ich habe jedoch den Eindruck, dass viele Anbieter von Anwendungen ihre eigenen Anwendungen nie wirklich nutzen. (Unter einem notwendigen Feature stelle ich mir nämlich nicht unbedingt einen tollen neuen Bildschirmschoner für Windows vor.) Das Programm zum Anpassen der Fenstergröße, das ich wirklich brauchte, stammt von einem Drittanbieter.

In diesem Buch geht es meist um das Hinzufügen neuer Anwendungsfeatures. Lesen Sie die entsprechenden Kapitel, wenn Sie herausfinden möchten, wie bestimmte Features hinzugefügt werden. (In den Abschnitten weiter vorne in diesem Kapitel finden Sie Informationen dazu, welche Kapitel das sind.) Wenn Sie das Buch von Anfang bis zum Ende lesen, können Sie mit VBA zu jedem beliebigen Produkt, das VBA unterstützt, so ziemlich jedes Feature hinzufügen. Ihre Freunde werden beeindruckt sein und Sie für ein Genie halten. Vielleicht sieht Ihr Chef endlich ein, dass Sie der wertvollste Mitarbeiter überhaupt sind. Und wer weiß, vielleicht springt dabei sogar eine ordentliche Gehaltserhöhung für Sie raus. Wenn Sie dieses Buch lesen, werden Sie vielleicht berühmt. Aber was viel wichtiger ist: Sie werden nicht mehr so frustriert sein.

Spezialtools erstellen

Wenn Sie einer anderen Person, die nicht mit Microsoft Office arbeitet, formatierte Daten schicken wollen, werden Sie lange nach einer Lösung suchen müssen. In Kapitel 10 und 11 lernen Sie zwei Methoden zum Speichern von Daten in unterschiedlichen Formaten kennen. In Kapitel 10 wird die altbekannte Textdatei verwendet, während in Kapitel 11 mit den XML-Dateien (eXtensible Markup Language) das Neueste vom Neuen zum Einsatz kommt.

So geht es nach Ihrem Kopf

Manchmal möchte ich einfach nur noch schreien. Microsoft bildet sich ein, genau zu wissen, was ich möchte, und das aufgrund von Umfragen. Wer da gefragt wird, bleibt ein Geheimnis. Aber ich würde dem Typen im dunklen Anzug neben Ihnen nicht über den Weg trauen.

Aber zum Glück gibt es VBA. Damit können Sie die gut gemeinten Anwendungsfeatures von Microsoft anpassen. Wenn Word unbedingt darauf besteht, Informationen anzuzeigen, die Sie beim Programmstart nicht sehen möchten, speichern Sie Ihre Einstellungen auf Festplatte und stellen Sie sie bei jedem Start von Word wieder her. Der Einsatz von automatisch ausführenden Programmen (siehe Kapitel 2) trägt dazu bei, dass die Dinge Ihren Wünschen entsprechen. In

Kapitel 10 erfahren Sie, wie Sie Ihre Einstellungen im Textformat, und in Kapitel 11, wie Sie diese im XML-Format speichern können.

Auch andere Produkte nutzen VBA

VBA können Sie übrigens nicht nur verwenden, wenn Sie mit Microsoft Office oder ein paar anderen Microsoft-Produkten arbeiten. Mit VBA haben Sie auch eine ganze Reihe anderer Anwendungen im Griff. Auf der Microsoft-Website finden Sie unter `http://msdn.microsoft.com/vba/companies/company.asp` eine Liste mit Unternehmen, die über eine Lizenz für VBA verfügen. Sie werden staunen, in wie vielen Anwendungen VBA integriert ist. Hier ein paar meiner Lieblingsprodukte:

- **Corel-Produkte (`www.corel.de`):** Corel stellt WordPerfect und CorelDRAW her. *WordPerfect* ist ein Textverarbeitungsprogramm. Für einen meiner ersten professionellen Aufträge musste ich mit WordPerfect arbeiten. *CorelDRAW* ist ein Zeichenprogramm, mit dem viele Profis gerne arbeiten und das zahlreiche Features unterstützt. Alle Diagramme in diesem Buch wurden ursprünglich mit CorelDRAW erstellt. Und all meine Zeichnungseinstellungen werden mit VBA-Programmen automatisch angewendet.
- **Micrografx iGrafx-Reihe (`www.micrografx.de`):** Mit diesem Produkt können Sie Flussdiagramme und Organigramme erstellen. Im Gegensatz zu vielen Zeichenaufgaben wiederholt sich sowohl bei Flussdiagrammen als auch bei Organigrammen vieles, weshalb diese Diagramme für VBA geradezu prädestiniert sind.
- **IMSI TurboCad (`www.turbocad.de`):** Ich arbeite gerne mit Holz. Das bedeutet, dass ich immer wieder Pläne für neue Projekte zeichnen muss. Dazu verwende ich am liebsten TurboCad. Hierbei handelt es sich um ein relativ kostengünstiges Programm. Und die VBA-Programme, die ich dafür geschrieben habe, erledigen viele der Zeichenaufgaben automatisch.

VBA gab es nicht immer schon. Sollten Sie aus den Tiefen Ihres Rechners WordPerfect für DOS ausgraben, werden Sie enttäuscht sein, weil diese Version VBA nicht unterstützt. In der Teilnehmerliste mit Microsoft-Anbietern steht nicht, welche Version eines Produkts VBA unterstützt. Das müssen Sie also entweder über die Produktverpackung herausfinden oder den Anbieter fragen.

Zimmer mit Aussicht

Kann es sein, dass Sie an VBA mit derselben Begeisterung und mit denselben klaren Gedanken herangehen wie ein Verurteilter an den Galgen? Wenn das der Fall ist, bekommen Sie bei der Arbeit mit einer Anwendung nicht viel mehr als das zu sehen, was der Entwickler Sie sehen lassen möchte. Sie befinden sich im Benutzerzimmer – dem ohne Aussicht. Gehen Sie an VBA so heran, als würden Sie ein neues Zimmer betreten. So haben Sie ein Zimmer *mit* Aussicht. Sie sind derjenige, der sieht, wann was geschehen wird.

Die integrierte Entwicklungsumgebung (IDE, Integrated Development Environment)

VBA ist eine visuelle Programmierumgebung. Das bedeutet, dass Sie vor dem Ausführen Ihres Programms sehen können, wie dieses aussieht. Der Editor dieser Umgebung besteht aus verschiedenen Fenstern, wodurch das Programmieren erleichtert wird und besser bedienbar ist. Ihnen werden leichte Veränderungen dieses Editors auffallen, wenn Sie ihn mit Vista anstelle einer älteren Windows-Version verwenden. Zusätzlich werden Sie Unterschiede feststellen, wenn Sie den Editor einer Office-Kernanwendung (das sind die mit der neuen Multifunktionsleiste) verwenden. In Abbildung 1.3 ist diese integrierte Entwicklungsumgebung (IDE, Integrated Development Environment) von Excel dargestellt, wenn sie unter Vista geöffnet wird. Unabhängig davon, welches Office-Produkt oder welche Windows-Version Sie verwenden, hat der Editor immer ein ähnliches Erscheinungsbild (und ein paar kleine Unterschiede), dieselben Menübefehle und dieselbe Funktionalität.

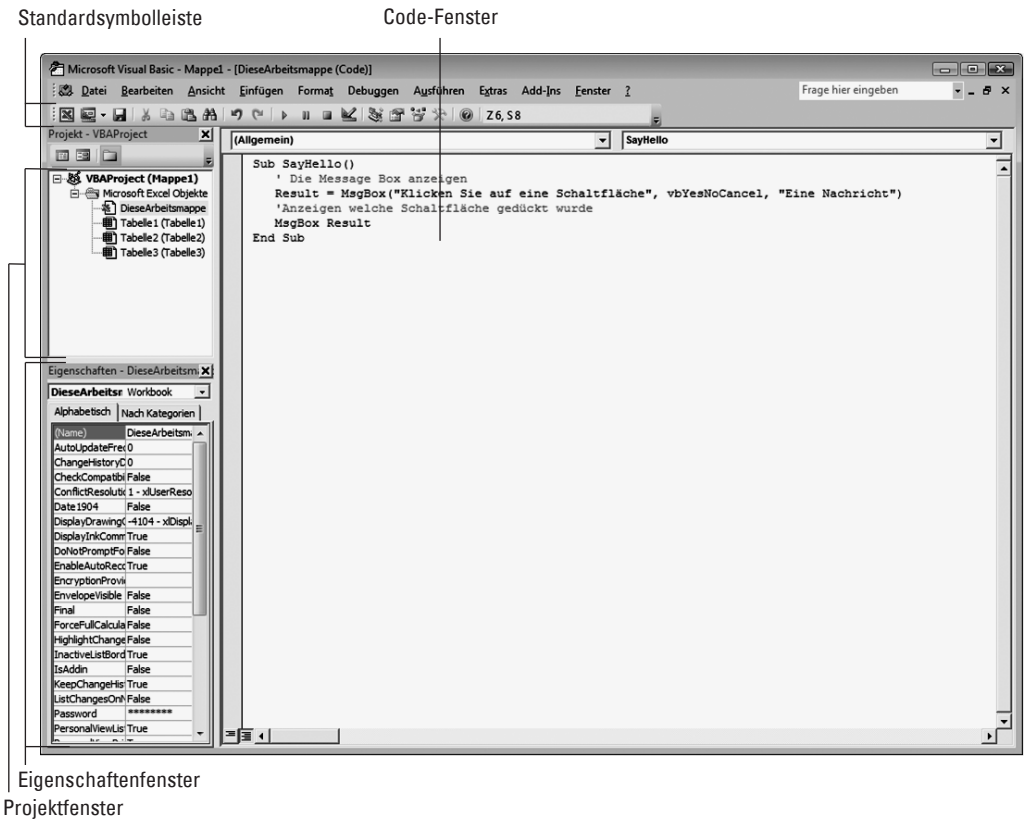

Abbildung 1.3: Die VBA-IDE ist ein Editor zum Schreiben von VBA-Anwendungen.

Eine *IDE* ist ein Editor so wie Ihr Textverarbeitungsprogramm, Ihr Tabellenkalkulationsprogramm oder Ihr Datenbankformular. Genauso wie Anwendungseditoren über spezielle Features verfügen, durch die diese Editoren besonders hilfreich zum Bearbeiten von Daten werden, so ist eine IDE ein Programmeditor mit speziellen Features, durch die dieser besonders hilfreich zum Schreiben von Anweisungen für die Anwendung wird. Bei diesen Anweisungen handelt es sich um *prozeduralen Code*, das heißt um eine Folge von Schritten.

Wie in Abbildung 1.3 zu sehen ist, besteht die VBA-IDE zunächst einmal aus einem Menüsystem, Symbolleisten, einem Projektfenster, einem Eigenschaftenfenster und einem Code-Fenster. In der IDE können bei Bedarf weitere Fenster angezeigt werden. Aber diese drei Fenster werden beim Start von VBA angezeigt. Im Folgenden finden Sie eine kurze Übersicht über die Aufgaben der einzelnen Fenster. (Weiter hinten in diesem Kapitel erfahren Sie, wie Sie diese Fenster nutzen können.)

- ✔ **Projekt:** Dieses Fenster zeigt eine Liste mit den Elementen in Ihrem Projekt an. Sämtliche Dokumentelemente des Projekts sind in einer Datei gespeichert. Ihre Anwendung befindet sich in einer Datei, die im Projektfenster angezeigt wird.
- ✔ **Eigenschaften:** Wenn Sie ein Objekt auswählen, wird dies im Eigenschaftenfenster angezeigt. So kann dieses Fenster beispielsweise anzeigen, ob das Objekt blau ist oder ob es Wörter enthält.
- ✔ **Code:** Irgendwann ist es dann einmal so weit, Code zu schreiben, damit Ihre Anwendung funktioniert. In diesem Fenster werden die Befehle angezeigt, die Ihrer Anwendung sagen, wie sie sich zu verhalten hat. Stellen Sie sich dieses Fenster als einen Ort vor, an dem Sie eine besondere To-do-Liste schreiben.

Die VBA-Werkzeugsammlung

Sie müssen nicht für jede Aufgabe in VBA-Code schreiben. Die IDE unterstützt Formulare, so wie die Formulare, die Sie zum Durchführen anderer Aufgaben verwenden. Hier entscheiden Sie, was auf dem Formular angezeigt wird und wie sich das Formular verhält, wenn der Benutzer damit arbeitet. Um das Erstellen von Formularen zu erleichtern, stellt VBA die Werkzeugsammlung bereit (siehe Abbildung 1.4), die Steuerelemente zum Erstellen von Formularen enthält.

Abbildung 1.4: Die VBA-Werkzeugsammlung

Jede Schaltfläche in der Werkzeugsammlung ist für eine bestimmte Aufgabe zuständig. So wird mit dem Klick auf eine Schaltfläche beispielsweise ein Textfeld zum Formular hinzugefügt, mit dem Klick auf eine andere Schaltfläche dagegen eine Befehlsschaltfläche. Die Formularfeatures, die mit diesen Schaltflächen erstellt werden, werden als *Steuerelemente* bezeichnet. In Kapitel 7 erfahren Sie, wie Sie all diese Steuerelemente verwenden und wie Sie weitere Steuerelemente hinzufügen können, wenn die in der Werkzeugsammlung enthaltenen nicht ausreichen.

Und was sind Objekte?

Der Ausdruck *Objekt* wird Ihnen beim Lesen dieses Buches und beim Erstellen einer eigenen Anwendung mit VBA immer wieder begegnen. Ein Objekt in einem Programm ist einem Objekt aus dem echten Leben sehr ähnlich. Programmierer haben sich diesen Ausdruck ausgedacht, um Programme leichter verständlich zu machen. Im Folgenden werde ich anhand des Apfels – einem Beispiel aus dem wirklichen Leben – erklären, was ein Objekt in VBA ist, warum Objekte so ein wichtiger Bestandteil von VBA sind und warum sie vieles vereinfachen.

Eigenschaftenwerte sind alles

Wenn Sie einen Apfel betrachten, sehen Sie einen Teil seiner Eigenschaften. Der Apfel ist rot, grün oder gelb. VBA-Objekte verfügen ebenfalls über Eigenschaften: So kann eine Schaltfläche beispielsweise eine *Beschriftung* aufweisen (einen Text, den der Benutzer sieht, wenn die Schaltfläche angezeigt wird). Einige der Apfeleigenschaften liegen aber im Verborgenen. Sie werden erst wissen, wie der Apfel schmeckt, wenn Sie in ihn hineinbeißen. Ähnlich verhält es sich bei VBA-Objekten. Auch VBA-Objekte besitzen versteckte Eigenschaften.

Meine Verrücktheit hat Methode

Mit einem Apfel können Sie eine Menge tun. So ist beispielsweise das Pflücken des Apfels vom Baum eine Methode der Interaktion mit dem Apfel. Auf ähnliche Art und Weise verfügen VBA-Objekte über Methoden. So können Sie beispielsweise mit der `Move`-Methode eine Schaltfläche von einer Stelle an eine andere verschieben. Mithilfe von *Methoden* kann der Entwickler mit einem Objekt etwas tun.

Und jetzt noch ein besonderes Ereignis

Ein Apfel verändert in der Regel während des Reifeprozesses seine Farbe. Niemand hat dem Apfel etwas angetan. Er wurde reif, weil er älter wurde. Das ist ein *Ereignis*. Auf ähnliche Art und Weise können VBA-Objekten Ereignisse widerfahren. Ein Benutzer klickt auf eine Befehlsschaltfläche, und die Befehlsschaltfläche generiert ein `Click`-Ereignis. Als Entwickler haben Sie der Befehlsschaltfläche nichts angetan. Die Befehlsschaltfläche entscheidet, wann das Ereignis generiert wird. Mit *Ereignissen* kann der Entwickler also auf sich ändernde Objektbedingungen reagieren.

Den Visual Basic-Editor starten

Wie Sie den Visual Basic-Editor starten, hängt von der Anwendung ab, die Sie verwenden. Neuere Office-Versionen verwenden einen anderen Ansatz als die älteren. In allen Fällen wird eine Anzeige ähnlich der in Abbildung 1.3 angezeigt. In diesem Abschnitt werden alle Möglichkeiten beschrieben.

Word 2007, Excel 2007 und PowerPoint 2007

Vergewissern Sie sich, dass Sie VBA aktiviert haben, so wie es im Abschnitt »VBA kommt mit Office« weiter vorne in diesem Kapitel beschrieben wurde. Nachdem Sie eingestellt haben, dass die Registerkarte ENTWICKLERTOOLS angezeigt wird, wählen Sie diese aus. Wenn Sie auf die Schaltfläche VISUAL BASIC auf der linken Seite der Multifunktionsleiste klicken, wird der Visual Basic-Editor angezeigt.

Access 2007

Access 2007 zeigt die Registerkarte DATENBANKTOOLS in der Multifunktionsleiste immer dann an, wenn es möglich ist den Visual Basic-Editor zu verwenden. Da Sie hierzu eine Datenbank geöffnet haben und bestimmte andere Bedingungen erfüllt sein müssen, werden Sie diese Registerkarte nicht immer sehen. Wenn Sie die Registerkarte sehen, wählen Sie sie aus und klicken Sie die Schaltfläche VISUAL BASIC an, dann wird der Visual Basic-Editor angezeigt.

OneNote 2007, Publisher 2007, Visio 2007, Project 2007 und alle älteren Office-Versionen

Wenn Sie eine der in der Überschrift dieses Abschnittes aufgelisteten Office-Anwendungen verwenden, starten Sie den Visual Basic-Editor, indem Sie aus dem Menü EXTRA|MAKRO|VISUAL BAISC EDITOR auswählen.

Sicherheit unter Vista

Vista stellt zusätzliche Sicherheitseinschränkungen für Office-Produkte bereit. Die *User Access Control* (UAC) macht es unmöglich, dass bestimmte Makros ausgeführt werden, die früher unter den alten Windows-Versionen tadellos funktioniert haben. In manchen Fällen hilft es noch nicht einmal, die Makrosicherheit zu verändern, je nach den Sicherheitsrichtlinien, die vom Administrator festgelegt wurden, Ihren persönlichen Sicherheitseinstellungen und der Aufgabe, die das Makro ausführt. Im Allgemeinen sollten Sie Ihre Makros digital signieren, bevor Sie sie unter Vista verwenden. Schauen Sie in den Abschnitt »Ihrer Kreation eine digitale Signatur hinzufügen« im Kapitel 5, um mehr zu diesem Thema zu erfahren.

Die Makrosicherheit in Word 2007, Excel 2007, PowerPoint 2007 und Access 2007 einstellen

Office 2007 legt die Latte für Sicherheit ziemlich hoch an. Es ist unwahrscheinlich, dass Sie die meisten in diesem Buch beschriebenen Makros ausführen können, ohne Ihre Sicherheitseinstellungen zu verändern. Die folgenden Schritte zeigen Ihnen, wie man die erforderlichen Änderungen durchführt:

1. **Wählen Sie die Registerkarte Entwicklertools in der Multifunktionsleiste aus.**
2. **Klicken Sie auf Makrosicherheit.**

   Das Dialogfeld Vertrauensstellungscenter wird angezeigt, so wie es in Abbildung 1.5 zu sehen ist.
3. **Wählen Sie Alle Makros aktivieren aus, außer Sie planen alle Makros in diesem Buch digital zu signieren.**

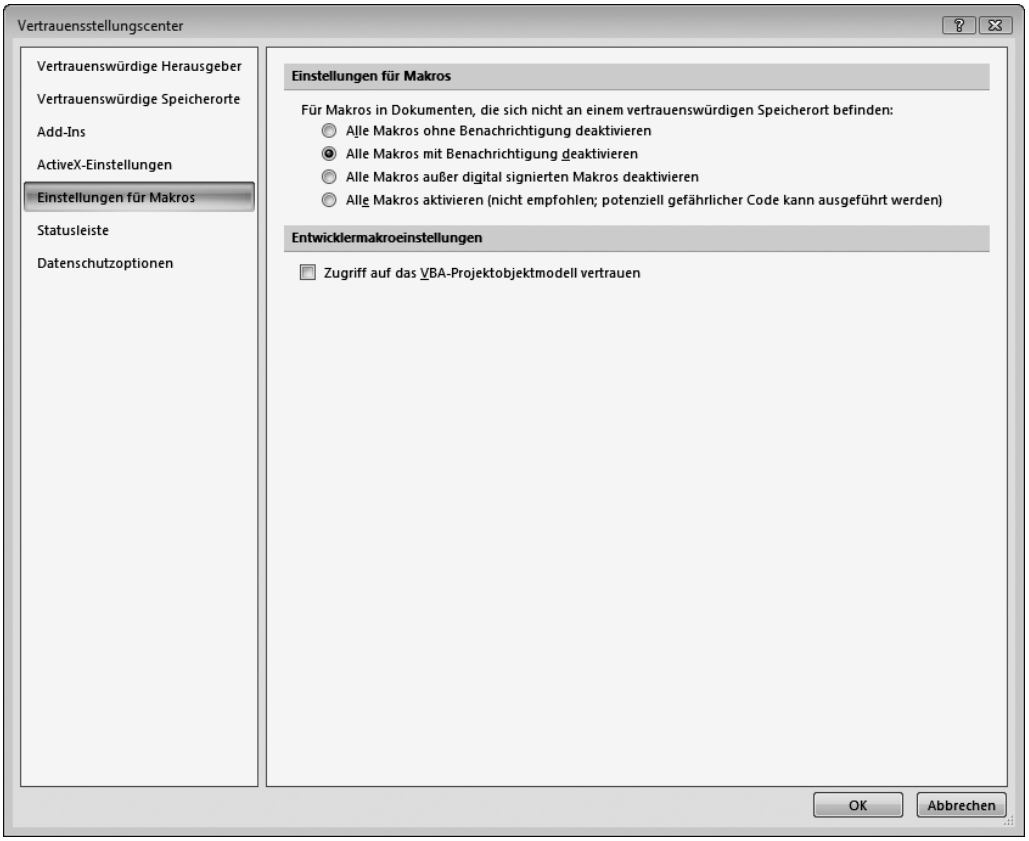

Abbildung 1.5: Im Dialogfeld Vertrauensstellungscenter stellen Sie die Sicherheitseinstellungen Ihrer Office-Anwendung ein.

4. **Wählen Sie Zugriff auf das VBA-Objektmodell vertrauen an.**
5. **Klicken Sie auf OK.**

   Sie können nun Makros ausführen, aber mit stark heruntergesetzter Sicherheit. Denken Sie daran, die Sicherheitseinstellungen so schnell wie möglich wieder hoch einzustellen.

Die Makrosicherheit in OneNote 2007, Publisher 2007, Visio 2007, Project 2007 und allen älteren Office-Versionen setzen

Je nachdem, welche Version von Microsoft Office Sie verwenden und wie Sie diese konfiguriert haben, kann es sein, dass das Makrosicherheitsfeature zu hoch eingestellt ist, um die Beispiele in diesem Buch zu verwenden. Sie können die Sicherheitsstufe für Makros ändern. Das kann jedoch dazu führen, dass auch Makroviren ausgeführt werden. Daher sollten Sie die Sicherheitsstufe für Makros nur bei vertrauenswürdigen Dateien ändern, von denen Sie wissen, dass sie keine Viren enthalten. Um die Makrosicherheitsstufe zu ändern, gehen Sie wie folgt vor:

1. **Wählen Sie Extras|Optionen.**

   Die Microsoft Office-Anwendung zeigt das Dialogfeld Optionen an.
2. **Wählen Sie die Registerkarte Sicherheit aus.**
3. **Klicken Sie auf Makrosicherheit.**

   Die Microsoft Office-Anwendung zeigt das Dialogfeld Sicherheit an.
4. **Wählen Sie auf der Registerkarte Sicherheitsstufe die Option Niedrig aus.**
5. **Klicken Sie zweimal auf OK, um die Dialogfelder Sicherheit und Optionen zu schließen.**

Der Projekt-Explorer

Der Projekt-Explorer wird im Projektfenster angezeigt. Mithilfe des Projekt-Explorers können Sie mit den Objekten eines Projekts arbeiten. Bei einem Projekt handelt es sich um eine einzelne Datei zum Speichern des Programms oder zumindest von Teilen davon. Das Projekt befindet sich in dem Office-Dokument, mit dem Sie arbeiten. Wenn Sie also das Office-Dokument öffnen, öffnen Sie damit auch das Projekt. In Kapitel 3 erfahren Sie mehr über den Zusammenhang zwischen Projekten und Programmen. Der Projekt-Explorer funktioniert mehr oder weniger genauso wie die linke Fensterseite des Windows-Explorers. Normalerweise werden nur die Objekte der obersten Ebene angezeigt wie das Excel-Objekt in Abbildung 1.6.

Welche Objekte im Projekt-Explorer angezeigt werden, hängt von der Anwendung ab, mit der Sie arbeiten. Wenn Sie beispielsweise mit Word arbeiten, werden Dokumente und Dokumentvorlagen angezeigt. Entsprechend werden bei der Verwendung von Excel Arbeitsmappen und Tabellen angezeigt. Es spielt jedoch keine Rolle, mit welcher Anwendung Sie arbeiten. Die Art und Weise der Verwendung des Projekt-Explorers ist immer dieselbe.

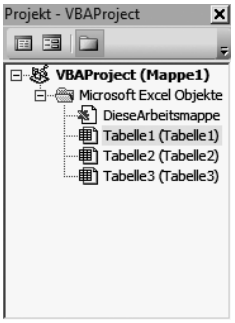

Abbildung 1.6: Verwenden Sie den Projekt-Explorer für die Arbeit mit Projektobjekten.

In Abbildung 1.6 sind außerdem noch einige besondere Objekte dargestellt. Ein Projekt kann Formulare, Module und Klassenmodule enthalten. Diese besonderen Objekte werden im Folgenden kurz beschrieben:

- ✔ **Formulare:** Enthalten Elemente der Benutzeroberfläche und helfen Ihnen, mit dem Benutzer zu kommunizieren. In Kapitel 7 erfahren Sie, wie Sie mit Formularen arbeiten können.
- ✔ **Module:** Enthalten den nicht visuellen Code für Ihre Anwendung. Sie können beispielsweise ein Modul zum Speichern einer speziellen Berechnung verwenden. Dieses Buch enthält in erster Linie Module.
- ✔ **Klassenmodule:** Enthalten neue Objekte, die Sie selbst erstellt haben. Sie können mit einem Klassenmodul beispielsweise einen neuen Datentyp erstellen. In Kapitel 8 erfahren Sie, wie Sie mit Objekten arbeiten können.

Um ein Objekt zum Anzeigen und Ändern der Eigenschaften auszuwählen, markieren Sie es im Projekt-Explorer. Um das Objekt zum Ändern zu öffnen, doppelklicken Sie auf das Objekt im Projekt-Explorer.

Die rechte Maustaste für alle Fälle

Der Projekt-Explorer verfügt über eine Reihe von verborgenen Talenten, die Sie entdecken können, wenn Sie mit der rechten Maustaste auf Objekte klicken, so dass angezeigt wird, was Sie mit diesen Objekten tun können. Klicken Sie beispielsweise mit der rechten Maustaste auf den Eintrag VBAProject (Mappe1) in Abbildung 1.6, um das in Abbildung 1.7 dargestellte Kontextmenü aufzurufen.

Es ist erstaunlich, was sich hinter diesem Menü alles verbirgt. Hier geht es erst einmal nicht um die Verwendung all dieser Menüeinträge. Die Menüeinträge werden mindestens einmal, häufig sogar mehrere Male in diesem Buch behandelt. In Kapitel 3 wird beispielsweise beschrieben, wie der Eintrag Eigenschaften von VBAProject verwendet wird. Hier geht es erst einmal darum, dass Sie wissen, dass die meisten Objekte über ein Kontextmenü verfügen, das Sie durch einen Klick mit der rechten Maustaste oder über die Kontextmenütaste auf Ihrer Tastatur aufrufen können.

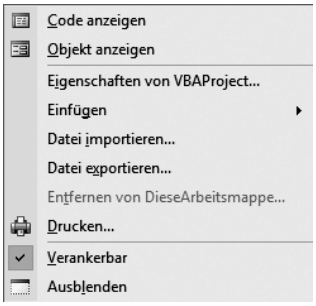

Abbildung 1.7: Klicken Sie mit der rechten Maustaste auf VBA-Objekte, um Kontextmenüs aufzurufen.

Mit speziellen Einträgen arbeiten

Es kann vorkommen, dass im Projekt-Explorer spezielle Einträge angezeigt werden. Wenn Sie beispielsweise mit einem Word-Dokument arbeiten, kann es vorkommen, dass der Ordner Verweise angezeigt wird. Dieser Ordner enthält Verweise, die im Word-Dokument erstellt werden. In der Regel enthält er eine Liste mit Vorlagen, die im Dokument zum Formatieren verwendet werden.

Häufig lassen sich die Objekte in den speziellen Ordnern nicht ändern. Dies trifft auch auf den Ordner Verweise zu, der von Word-Dokumentobjekten verwendet wird. Der Ordner Verweise dient lediglich der Information. Um die Vorlage zu ändern, auf die verwiesen wird, müssen Sie nach dem entsprechenden Objekt im Projekt-Explorer suchen. In diesem Buch gehe ich auf die speziellen Objekte nicht weiter ein, da Sie mit diesen normalerweise nicht arbeiten müssen.

Das Eigenschaftenfenster

Die meisten Objekte, auf die Sie in der VBA-IDE klicken, verfügen über Eigenschaften, die das Objekt beschreiben. Falls Sie mit Eigenschaften noch nicht gearbeitet haben, finden Sie weiter vorne in diesem Kapitel eine kurze Beschreibung. In den nachfolgenden Abschnitten wird das Eigenschaftenfenster (siehe Abbildung 1.3) ausführlicher beschrieben.

Grundlegendes zu Eigenschaftentypen

Eine Eigenschaft muss das Objekt beschreiben. Beim Betrachten eines Objekts setzen Sie selbstverständlich etwas über die von einer bestimmten Eigenschaft angegebenen Informationen voraus. Wenn beispielsweise die Farbe eines Apfels beschrieben wird, erwarten Sie, dass Eigenschaften wie *rot*, *gelb* oder *grün* verwendet werden. Auf ähnliche Art und Weise verfügen VBA-Objekte über bestimmte Typen.

Einer der am häufigsten verwendeten Eigenschaftentypen ist der Typ `Text`. Die `Caption`-Eigenschaft eines Formulars ist vom Typ `Text`. Der hier eingegebene Text wird beim Öffnen des Formulars am oberen Rand des Formulars angezeigt.

Ein weiterer, recht häufig verwendeter Eigenschaftentyp ist ein logischer oder Boolescher Wert. Wenn ein Steuerelement beispielsweise über eine `Visible`-Eigenschaft verfügt und für diese Eigenschaft der Wert `True` festgelegt wurde, wird das Steuerelement auf dem Bildschirm angezeigt. Wenn Sie für diese Eigenschaft `False` festlegen, wird das Steuerelement nicht angezeigt, obwohl es nach wie vor Teil der Anwendung ist.

Objekteigenschaften können auch numerische Werte annehmen. Zum Beschreiben der Stelle, an der ein Steuerelement auf dem Bildschirm angezeigt werden soll, geben Sie für die `Top`- und `Left`-Eigenschaften jeweils einen numerischen Wert ein. Diese Werte geben an, wie viele Pixel sich zwischen der linken oberen Ecke des Bildschirms und der linken oberen Ecke des Steuerelements befinden.

Häufig kann eine Eigenschaft eine Drop-down-Liste anzeigen, aus der Sie den passenden Wert auswählen können. Andere Eigenschaften zeigen ein Dialogfeld wie das für Farbe an (siehe Abbildung 1.8).

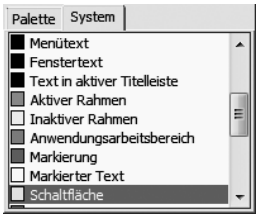

Abbildung 1.8: Manche Eigenschaften zeigen ein Dialogfeld zum Auswählen des passenden Wertes an.

Die Hilfe für Eigenschaften anzeigen

Sie werden sich nicht alle Eigenschaften für alle von VBA-Anwendungen erstellbaren Objekte merken können. Das können nicht einmal die echten Gurus. Um zu erfahren, was eine bestimmte Eigenschaft für eine Anwendung tun kann, markieren Sie einfach die Eigenschaft, drücken die Taste F1, und bekommen von VBA (in der Regel) ein Hilfefenster angezeigt wie das in Abbildung 1.9.

Die älteren Versionen der Office-Hilfe enthalten nicht so viele Funktionen, wie in Abbildung 1.9 zu sehen sind. So gibt es beispielsweise keine Funktion, mit der man Microsoft mitteilen kann, ob die Informationen hilfreich waren oder nicht. Des Weiteren hat die neue Hilfe unten eine Statusleiste, die Ihnen anzeigt, ob die dargestellten Informationen statisch sind oder direkt von der Microsoft-Website kommen. Des Weiteren gibt es nun in der Symbolleiste eine Schaltfläche, die wie eine Heftzwecke aussieht. Klicken Sie diese an, so bleibt das Hilfefenster immer oben, so dass Sie es immer sehen können, unabhängig davon, was Sie gerade machen. Ist die Heftzwecke in der anderen Position, so wird das Hilfefenster wie jedes andere Fenster versteckt, wenn Sie es mit einem anderen Fenster überdecken.

Hilfefenster wie diese geben Informationen zur Eigenschaft und deren Verwendung und enthalten Verknüpfungen zu weiteren Informationen. Die weiteren Informationen sind besonders wichtig, wenn Sie beginnen, die Eigenschaftenwerte im Anwendungscode zu ändern. Wenn

Sie beispielsweise auf die Verknüpfung Beispiel klicken, zeigt das Hilfesystem an, wie Sie Code unter Verwendung dieser Eigenschaft schreiben können.

Wenn Sie in einem Hilfefenster auf die Verknüpfung Siehe auch klicken, werden weitere Informationen zum Thema angezeigt, zum Beispiel Informationen zu Objekten, Eigenschaften, Methoden und Ereignissen zum Thema. Gelegentlich werden auch empfohlene Möglichkeiten für die Arbeit mit einem Objekt, einer Eigenschaft, einer Methode oder einem Ereignis angezeigt.

Abbildung 1.9: In der Hilfe werden die von VBA unterstützten Eigenschaften beschrieben.

Das Code-Fenster

Das Code-Fenster ist die Stelle, an der Sie den Anwendungscode eingeben. Dieses Fenster funktioniert wie andere Editoren, die Sie bereits kennen, und unterscheidet sich nur dadurch, dass Sie in einer speziellen Sprache schreiben: VBA. In Abbildung 1.10 ist ein Beispiel für ein Code-Fenster mit geladenem Code dargestellt.

Ein vorhandenes Code-Fenster öffnen

Es kann vorkommen, dass Sie eine Anwendung nicht fertig schreiben können und zu einem späteren Zeitpunkt weiterschreiben möchten. Um ein vorhandenes Code-Fenster zu öffnen,

suchen Sie das gewünschte Modul im Projekt-Explorer. Doppelklicken Sie auf den Moduleintrag, und der Code wird im Code-Fenster angezeigt.

Es gibt darüber hinaus auch noch andere Möglichkeiten, das Code-Fenster aufzurufen. Wenn Sie beispielsweise auf eines der Steuerelemente in einem Formular doppelklicken, wird das Code-Fenster angezeigt, so dass Sie zum Standardereignis-Handler Code hinzufügen können. VBA ruft den Ereignis-Handler (spezieller Code, der auf das Ereignis reagiert) jedes Mal auf, wenn ein festgelegtes Ereignis eintritt.

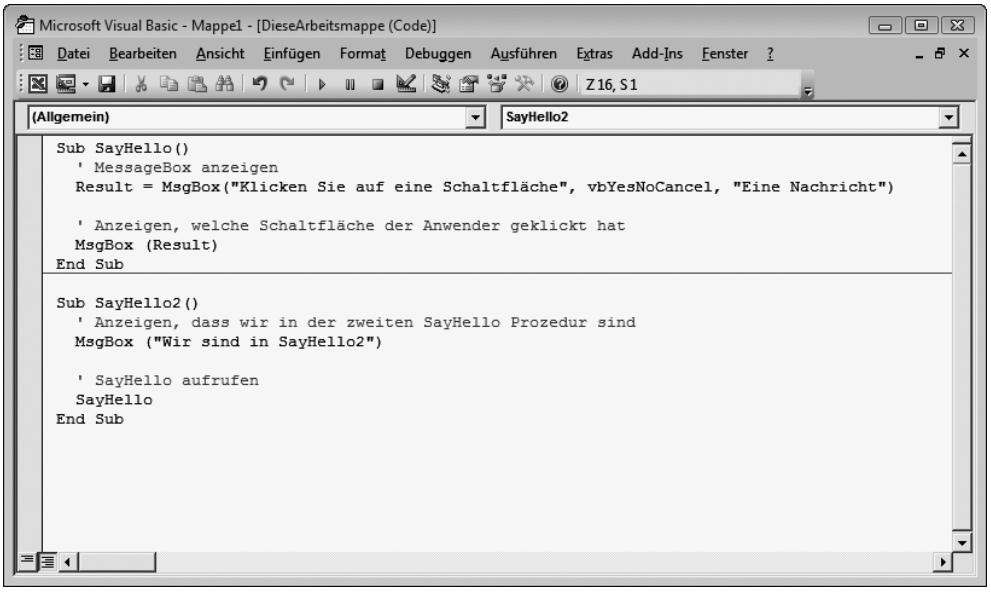

Abbildung 1.10: Verwenden Sie das Code-Fenster zum Ändern Ihres Programms.

Ein neues Code-Fenster erstellen

Wenn Sie innerhalb eines vorhandenen Dokuments oder einer vorhandenen Vorlage ein neues Modul starten, öffnen Sie ein neues Code-Fenster. Verwenden Sie dazu entweder den Befehl Einfügen|Modul oder den Befehl Einfügen|Klassenmodul. Nach dem Speichern dieses Moduls beziehungsweise dieses Klassenmoduls wird dieses im Projekt-Explorer zusammen mit den anderen Modulen und Klassenmodulen im Projekt angezeigt.

Im Code-Fenster Text eingeben

Wenn Sie Code eingeben, wird dieser von VBA überprüft. Wenn Sie einen größeren Fehler machen und beispielsweise ein Wort eingeben, das VBA nicht kennt, wird eine Fehlermeldung angezeigt, die Ihnen sagt, dass Sie einen Fehler gemacht haben (siehe Abbildung 1.11). Wenn Sie die Fehlermeldung nicht verstehen, klicken Sie auf die Schaltfläche Hilfe, um weitere Informationen anzuzeigen.

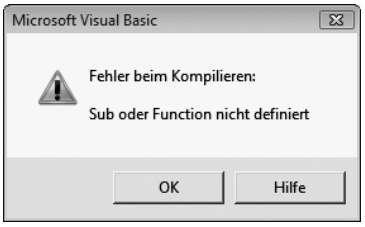

Abbildung 1.11: VBA zeigt eine Fehlermeldung an, wenn Sie einen Fehler machen.

Wenn Sie den Code für Ihre Anwendung eingeben, wird dieser gleich von VBA formatiert. Wenn Sie beispielsweise ein Schlüsselwort in Kleinbuchstaben eingeben, wird dies von VBA geändert, so dass es wie in der Hilfedatei dargestellt wird.

Schlüsselwörter werden zum leichteren Erkennen außerdem in einer anderen Farbe angezeigt. In diesem Buch sind Beispiele für die am häufigsten verwendeten VBA-Schlüsselwörter enthalten.

Weitere Code-Fensterfeatures entdecken

Das Code-Fenster verfügt wie andere Objekte in VBA ebenfalls über ein Kontextmenü. Wenn Sie mit der rechten Maustaste auf das Code-Fenster klicken, wird eine Liste mit optionalen Aktionen zum Ausführen angezeigt. Es kann beispielsweise eine Liste mit Eigenschaften und Methoden angezeigt werden, die für das Objekt gelten, das Sie derzeit im Fenster geöffnet haben. In Kapitel 3 wird die Verwendung vieler dieser speziellen Code-Fensterfeatures beschrieben.

Die Hilfe für Code anzeigen

Da es schwierig ist, sich zu merken, wie genau die einzelnen von VBA unterstützten Funktionen und Methoden verwendet werden, können Sie die VBA-Hilfe verwenden. Sie können jedes Schlüsselwort, das Sie im Code-Fenster eingeben, markieren und die Taste F1 drücken, damit VBA die Hilfe zum ausgewählten Wort anzeigt.

Achten Sie darauf, dass Sie das ganze Schlüsselwort auswählen, da es sonst sein kann, dass VBA die gesuchten Informationen nicht findet. Doppelklicken Sie auf das Schlüsselwort, um sicherzustellen, dass Sie das ganze Wort markieren.

Das Direktfenster

Obwohl Sie das Direktfenster natürlich zum Debuggen von Anwendungen verwenden können, kann Ihnen dieses Fenster aber auch helfen, VBA kennen zu lernen, ohne seitenweise Code schreiben zu müssen. Damit können Sie zum Beispiel Anweisungen schrittweise ausführen. Rufen Sie das Direktfenster mit dem Befehl Ansicht|Direktfenster auf. Dieses Fenster wird in der Regel unten in der IDE angezeigt und enthält erst Informationen, wenn Sie etwas eintippen.

Im Direktfenster eine Variable erstellen

Die meisten Entwickler verbringen viel Zeit mit dem Direktfenster und überprüfen damit ihre Anwendungen auf Fehler. Sie können das Direktfenster beispielsweise verwenden, um VBA nach dem Wert einer Variablen zu fragen. (Eine Variable dient als Speicherposition für einen Wert wie `Hallo Welt`.) Dieses Feature ist in der VBA-IDE immer verfügbar, auch wenn Sie VBA momentan nicht verwenden. Probieren Sie dieses Feature doch einfach einmal aus. Geben Sie hierzu im Direktfenster `MyVal = "Hallo Welt"` ein und drücken Sie die Taste ↵. (Achten Sie darauf, dass Sie die doppelten Anführungszeichen verwenden.) Geben Sie jetzt `? MyVal` ein und drücken Sie die Taste ↵. In Abbildung 1.12 ist das Ergebnis dieses kleinen Experiments dargestellt.

Abbildung 1.12: Überprüfen Sie mit dem Direktfenster den Wert einer Variablen.

Sie haben VBA gebeten, die Variable `MyVal` zu erstellen und dieser den Wert `Hallo Welt` zuzuweisen. Im nächsten Schritt haben Sie VBA mit dem Operator `?` gefragt, was `MyVal` enthält. In Abbildung 1.12 ist dargestellt, dass `MyVal` die Zeichenfolge `Hallo Welt` enthält.

Ein einzeiliges Programm schreiben

Experimente mit dem Direktfenster stellen die schnellste Möglichkeit dar, VBA kennen zu lernen, da die Ergebnisse sofort sichtbar werden. Erfolgreiche Experimente können Sie im Direktfenster kopieren und im Code-Fenster einfügen. Damit ist sichergestellt, dass Ihr Code weniger Fehler enthält, als wenn Sie ihn direkt im Code-Fenster eingeben.

Wenn Sie bis hierher gefolgt sind, haben Sie bereits die Variable `MyVal` erstellt. Diese Variable existiert so lange, bis Sie VBA schließen. Sie können diese Variable für ein kleines Experiment, Ihr erstes Programm, verwenden. Geben Sie im Direktfenster `MsgBox MyVal` ein und drücken Sie die Taste ↵. Daraufhin wird das in Abbildung 1.13 dargestellte Dialogfeld angezeigt.

Abbildung 1.13: Mit der Funktion `MsgBox` wird ein Dialogfeld wie dieses erstellt.

Herzlichen Glückwunsch! Sie haben soeben Ihre erste VBA-Anwendung geschrieben! Mit dem Code, den Sie geschrieben haben, haben Sie VBA aufgefordert, den Text in der Variablen `MyVal` unter Verwendung der Funktion `MsgBox` anzuzeigen. Klicken Sie auf OK, um das Dialogfeld zu schließen.

Der Objektkatalog

Bei VBA haben Sie auf mehr Objekte Zugriff, als Sie je für ein Programm verwenden werden. Angesichts all der Objekte kann es durchaus einmal vorkommen, dass Sie den Namen eines oder mehrerer Objekte vergessen. Mit dem Objektkatalog werden Sie das gesuchte Objekt dennoch finden. Sie können mit dem Objektkatalog auch neue Objekte finden, die für Ihr nächstes Projekt vielleicht ganz praktisch sind. Wählen Sie Ansicht|Objektkatalog, um den Objektkatalog aufzurufen (siehe Abbildung 1.14). In der Regel müssen Sie die Informationen filtern.

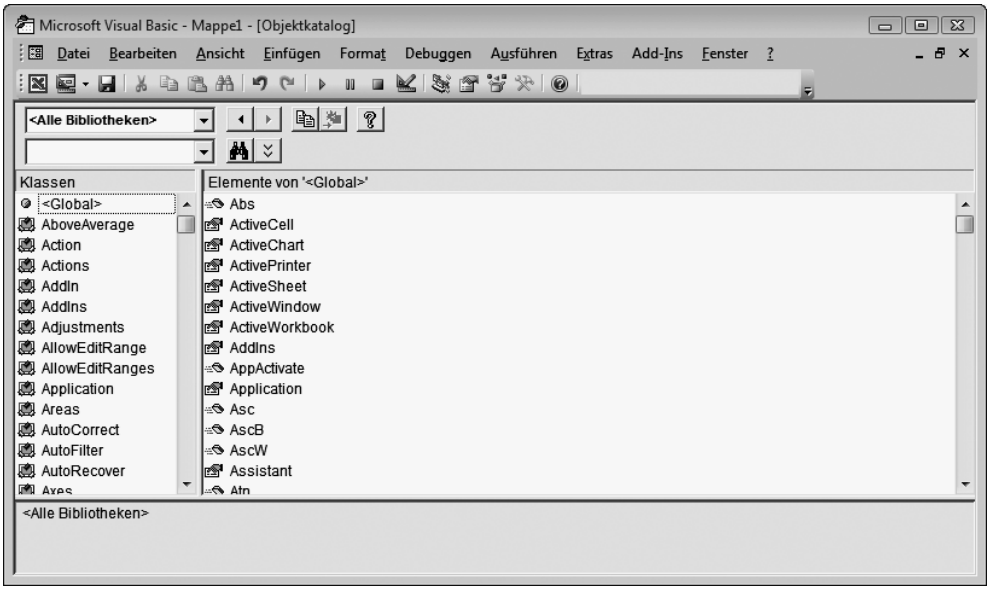

Abbildung 1.14: Zeigen Sie die in VBA verfügbaren Objekte im Objektkatalog an.

Objekte suchen

Der Objektkatalog enthält eine Liste mit dem Inhalt aller für die VBA-IDE geladenen Projekte und Bibliotheken. Sie können die Liste mit den Projekten und Bibliotheken mithilfe des Drop-down-Listenfelds Projekt/Bibliothek anzeigen. Beim Öffnen des Objektkatalogs steht im Listenfeld <Alle Bibliotheken>. Das bedeutet, dass das gesamte Angebot von VBA angezeigt wird. Das ist in der Regel jedoch viel zu viel.

Projekte und Bibliotheken sind nicht dasselbe. Aber darum brauchen Sie sich bei der Verwendung der darin enthaltenen Objekte normalerweise keine Gedanken zu machen. Ein *Projekt* ist der VBA-Code, der in einer der Dateien enthalten ist, die Sie in die Anwendung laden. In der Regel verwenden Sie ein Projekt zum Speichern des von Ihnen erstellten Codes. Eine *Bibliothek* ist externer Code, der in einer DLL-Datei (Dynamic Link Library) gespeichert ist. Die DLL-Datei enthält Routinen zur Unterstützung, die von der Anwendung oder von VBA verwendet werden. Dieser Code wird meist von einem Entwickler in einer Sprache wie Visual Basic oder Visual C++ geschrieben. Der Code in einer DLL-Datei lässt sich nicht einfach bearbeiten.

Auf den ersten Blick mag die Liste mit Projekten und Bibliotheken recht kompliziert erscheinen. Aber Sie haben die Möglichkeit, die Anzahl der angezeigten Einträge zu begrenzen. Ihre Projektvorlagen werden natürlich immer angezeigt. Neben den Projektvorlagen werden in der Liste die folgenden Bibliotheken angezeigt:

- **Anwendung:** Diese Bibliothek trägt den Namen der Anwendung, zum Beispiel Excel oder Word, und enthält außerdem die Features, die die Anwendung für den VBA-Benutzer bereitstellt. So verfügt die Excel-Bibliothek beispielsweise über ein `Chart`-Objekt, das eine Liste mit Methoden, Eigenschaften und Ereignissen für von Excel unterstützte Diagramme enthält.
- **Office:** Diese Bibliothek enthält eine Liste mit Objekten, die von Microsoft Office unterstützt werden. An dieser Stelle finden Sie beispielsweise die Objekte zur Unterstützung des Office-Assistenten. Wenn Sie mit einer Anwendung arbeiten, die nicht zu Microsoft Office gehört, wird diese Bibliothek natürlich nicht angezeigt. Dafür hält Ihre Anwendung möglicherweise andere Bibliotheken bereit.
- **StdOLE:** Diese Bibliothek enthält einige der OLE-Features (Object Linking and Embedding), die Sie in der Anwendung verwenden. Wenn Sie beispielsweise in einem Word-Dokument ein Bild einfügen, stellt diese Bibliothek die erforderliche Unterstützung bereit. Sie können diese Bibliothek auch in Ihren VBA-Anwendungen verwenden. Aber die Bibliothek von Office oder von der Anwendung ermöglicht den Zugriff auf Objekte, die sich einfacher und schneller nutzen lassen.
- **VBA:** Diese Bibliothek enthält Hilfsprogrammobjekte, die von VBA-Entwicklern benötigt werden. So enthält diese Bibliothek beispielsweise die weiter vorne in diesem Kapitel vorgestellte `MsgBox`-Funktion.

Wenn Sie in den Bibliotheken nach einem bestimmten Objekt suchen, sollten Sie die Menge an zu durchsuchenden Informationen mit den Optionen im Drop-down-Listenfeld Projekt/Bibliothek einschränken (den Inhalt *filtern*). Dies empfiehlt sich übrigens auch beim Durchführen von Suchvorgängen.

Im Objektkatalog nach Namen und Features suchen

Sie suchen nach dem Namen einer Methode oder eines anderen Programmierfeatures, das Sie verwenden möchten, und er fällt Ihnen einfach nicht mehr ein? Da kann Ihnen vielleicht die

Suchfunktion des Objektkatalogs weiterhelfen. Geben Sie einfach im Objektkatalog im Feld Suchtext (das leere Feld unterhalb des Felds <Alle Bibliotheken>) den zu suchenden Text ein und klicken Sie auf die Schaltfläche Suchen (die Schaltfläche mit einem Fernglas). Im Bereich Suchergebnisse in Abbildung 1.15 wird das Ergebnis für die Suche nach `MsgBox` angezeigt.

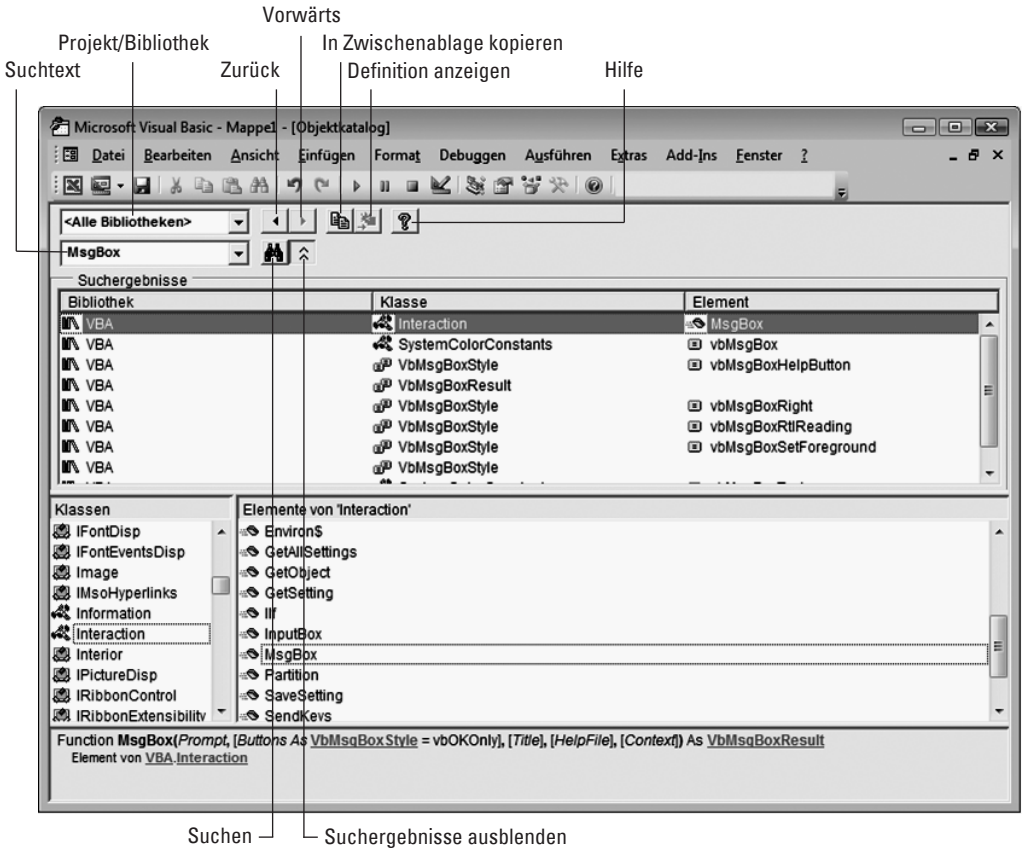

Abbildung 1.15: Suche nach der Methode, die Sie verwenden möchten

Wenn Sie einen der Einträge im Bereich Suchergebnisse auswählen (markieren), wird dieser Eintrag in den beiden unteren Fenstern angezeigt. So können Sie nach bestimmten Informationen zum Suchergebnis suchen und diese zusammen mit Methoden, Eigenschaften und Ereignissen anzeigen. Im unteren Fenster finden Sie Informationen zum ausgewählten Element. In diesem Beispiel erfahren Sie, wie Sie die `MsgBox`-Funktion verwenden können.

Ausschneiden und Einfügen im Objektkatalog

Wenn Sie im Objektkatalog eine Methode, eine Eigenschaft oder ein Ereignis finden, das Sie verwenden möchten, können Sie diese Daten in die Zwischenablage kopieren. Klicken Sie hierzu auf die Schaltfläche In Zwischenablage kopieren (die Schaltfläche mit dem Symbol, das

wie zwei Dokumente aussieht) und fügen Sie die Daten dann direkt in Ihren Anwendungscode ein. Damit müssen Sie nicht so viel Code eintippen. Außerdem können auf diese Weise nicht so viele Fehler passieren.

Die Hilfe im Objektkatalog anzeigen

Manchmal reichen die Informationen, die unten im Objektkatalog zu einem Element angezeigt werden, nicht aus. Wenn dies der Fall ist, markieren Sie das Element, zu dem Sie ausführlichere Informationen erhalten möchten, und drücken Sie die Taste F1. VBA zeigt daraufhin das Hilfefenster für dieses Element an.