Inhaltsverzeichnis

Vorwort 4
Anforderungen 4
Grenzen 4
Hilfestellung notwendig? 4
Die Entwicklungsumgebung entdecken 5
Verschiedene VBA-Versionen 5
Den VBA-Editor starten 5
Die Entwicklungsumgebung im Überblick 6
Der Projektexplorer 6
Anweisungen im Direktfenster ausführen 7
Die VBA-Hilfe aufrufen 7
Module erstellen und bearbeiten 8
Code strukturieren 13
Einfache Berechnungen 20
Grundlegendes zu Ausdrücken 20
Mathematische Berechnungen 20
Arbeiten mit Zeichenketten und Zeichen 24
Verzweigungen und Schleifen 26
Verzweigungen mit If 26
Verzweigungen verschachteln 26
Schleifen – Code mehrfach ausführen 27
Zugreifen auf Excel-Daten 31
Objektorientierte Programmierung 31
Makros mit dem Makrorekorder aufzeichnen 31
Der Objektkatalog 33
Arbeiten mit Excel-Objekten 34
Excel manipulieren 43
Excel-Version und VBA-Version ermitteln 43
Betriebssystem abfragen 44
Standardarbeitsverzeichnis auslesen und setzen 45
Prüfen, ob eine bestimmte Arbeitsmappe geöffnet ist 45
Dateien öffnen 46
Eine neue Arbeitsmappe erstellen und speichern 47
Eigene Benutzeroberflächen gestalten 48
Meldungen und Eingabeaufforderungen anzeigen 48
Datei- und Verzeichnisnamen auswählen 49
Eigene Dialogfelder erstellen 51
Auf Ereignisse und Fehler reagieren 54
Code bei Ereigniseintritt ausführen 54
Das Formular optimieren 57
Fehlersuche und Fehlerbehandlung 58
Wichtige VBA-Datentypen 61
Stichwortverzeichnis 62

Vorwort

Auch wenn Excel schon sehr viel kann, gibt es immer noch Aufgaben, die sich so ganz mit Excel-Bordmitteln nicht lösen lassen. Hier kommt VBA ins Spiel. Die Grundlagen der VBA-Programmierung sind ganz schnell gelernt. Wer etwas logisch denken kann und mathematisch nicht ganz unbegabt ist, kann so schnell eigene Funktionen und Prozeduren erstellen, die diese Probleme lösen.

VBA ist für alle die richtige Programmiersprache, die eine Microsoft-Office-Anwendung wie Word, Excel oder Access haben und diese steuern und manipulieren möchten. Außerdem ist VBA recht einfach zu erlernen, so dass Sie damit auch sehr anschaulich die ersten Programmiererfahrungen machen können.

Anforderungen

Wenn Sie erfolgreich mit diesem Heft arbeiten möchten, sollten Sie die Grundlagen von Windows kennen und wissen, wie Sie Dateien erstellen, kopieren und verschieben können. Sie sollten sich auch soweit mit Excel 2010 oder höher auskennen, dass Ihnen Begriffe wie Zelle, Zelladresse und Tabellenblatt bekannt sind.

Außerdem brauchen Sie natürlich noch Software. Da VBA in Excel integriert ist, benötigen Sie nur Excel 2010 oder höher für Windows.

Auch frühere Versionen sind möglich, allerdings werden Sie einige Einstellungen und Vorgehensweisen im Heft dann nicht 1:1 nachvollziehen können. Der Code sollte in aller Regel aber auch in älteren Versionen funktionieren.

Grenzen

Natürlich kann ein Heft in dieser Stärke nicht in die Tiefen der VBA-Programmierung vordringen, aber am Ende des Heftes kennen Sie sich so gut aus, dass Sie sich fortbilden können, ohne in teure Literatur investieren zu müssen. Die Online-Hilfe enthält nämlich alles, was man zu VBA wissen muss. Es geht also vor allem darum, die in der Hilfe verwendeten Begriffe zu verstehen – und genau das, oder die Grundlagen dazu, lernen Sie an praktischen Beispielen auf den folgenden Seiten.

Hilfestellung notwendig?

Sollten Sie mit einem Beispiel oder einer Erklärung aus diesem Heft nicht zurechtkommen, können Sie sich gerne an mich wenden.

Besuchen Sie dazu einfach meine Webseite: www.helma-spona.de

Hier finden Sie ein Kontaktformular, über das Sie Ihre Frage loswerden können.

Ich bitte aber um Verständnis dafür, dass ich wirklich nur Fragen zu meinen Veröffentlichungen beantworte und keine darüber hinausgehende kostenlose Hilfe leisten kann. Wenn Sie etwas programmiert haben möchten, wäre das nur als kostenpflichtige Dienstleistung möglich.

Und nun viel Spaß und viel Erfolg beim Programmieren mit VBA!

Helma Spona

Die Entwicklungsumgebung entdecken

Jede Programmiersprache benötigt eine Entwicklungsumgebung. Dieser Begriff bezeichnet ein Programm, in dem Sie Ihren Code erfassen, testen und sofern notwendig kompilieren können. Kompilieren bedeutet, dass der Quellcode, der aus einfachen Textanweisungen besteht, in eine Form überführt wird, die von einem Computer ausgeführt werden kann.

> Das Programm, das die Kompilierung durchführt, heißt Compiler.

VBA-Code wird automatisch kompiliert, wenn Sie ihn ausführen. Ein separater Schritt ist dazu nicht erforderlich.

Allerdings führt die Kompilierung von VBA-Code nicht zu einer eigenständigen Anwendung, wie das eine EXE-Datei wäre, die Sie einfach per Doppelklick ausführen können. VBA-Code benötigt eine sogenannte VBA-Hostanwendung. Dies kann prinzipiell jede Anwendung sein, die VBA in der entsprechenden Version unterstützt, also sowohl Word als auch Excel, Access oder PowerPoint.

Der VBA-Code wird grundsätzlich in der Datei der entsprechenden Hostanwendung gespeichert. Wenn Sie den Code also in Excel ausführen möchten, fügen Sie ihn in eine Excel-Arbeitsmappe ein und öffnen diese dann in Excel. Bearbeiten können Sie den Code in jeder VBA-Hostanwendung im gleichen Editor – dem VBA-Editor.

Verschiedene VBA-Versionen

Es gibt nicht nur eine Version von VBA. In Excel 5.0 und Excel 7.0, den ersten VBA-Hostanwendungen, war genau wie in Office 97 die Version 5.0 integriert. Die danach folgenden Office-Anwendungen für Windows, also Office 2000, XP und 2003, unterstützen hingegen VBA 6.0 bzw. VBA 6.3 (Office 2003 und 2007). In Office 2010 ist VBA 7 und in Office 2013 ist VBA 7.3 integriert.

> Die VBA-Version wird allerdings erst dann wirklich interessant, wenn Sie in die Tiefen der Programmierung vordringen. Für den Einstieg und einfache Lösungen für den Alltag brauchen Sie sich darum nicht großartig zu kümmern. VBA 6.0 und höher, sollten Sie allerdings für die folgenden Beispiele schon verwenden. In den Bildschirmausdrucken der folgenden Anleitungen wird Excel 2013 gezeigt und auf Änderungen gegenüber Excel 2010 hingewiesen. Wenn Sie noch ältere Versionen nutzen, können die gezeigten Bedienelemente und Schritte daher abweichen.

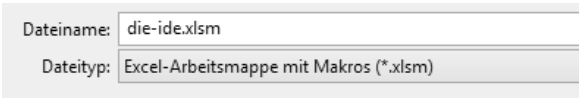
Den VBA-Editor starten

Um den VBA-Editor zu starten, müssen Sie VBA nicht extra installieren – er wird automatisch mit der VBA-Hostanwendungen installiert. Sie können also gleich loslegen:

- Starten Sie Excel und erstellen Sie eine neue, leere Arbeitsmappe, sofern diese nicht automatisch erzeugt wird.

- Alternativ können Sie natürlich auch eine vorhandene Arbeitsmappe öffnen, wenn Sie dort Ihren Code integrieren möchten.
- Speichern Sie nun die Arbeitsmappe in einem geeigneten Dateiformat. Das ist ab Excel 2007 das Format ".xlsm", denn Dateien im "xlsb"- oder xlsx-Format können keine VBA-Projekte speichern.

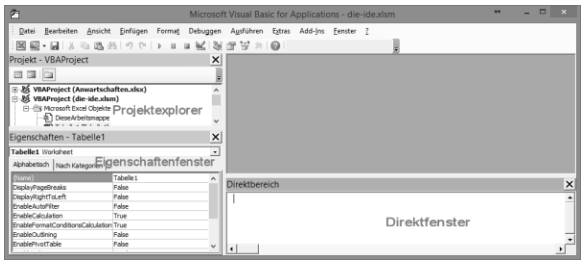
- Klicken Sie dazu auf DATEI und wählen Sie SPEICHERN UNTER aus.
- Geben Sie nun einen geeigneten Dateinamen ein und wählen Sie als Dateiformat EXCEL-ARBEITSMAPPE MIT MAKROS (*.XLSM) aus und klicken Sie auf SPEICHERN.



- Drücken Sie nun [ALT] + [F11], um den VBA-Editor zu starten.

Die Entwicklungsumgebung im Überblick

Die Entwicklungsumgebung besteht aus mehreren Fenstern. Die wichtigsten sind links oben der Projektexplorer, links unten das Eigenschaftenfenster sowie das Direktfenster.



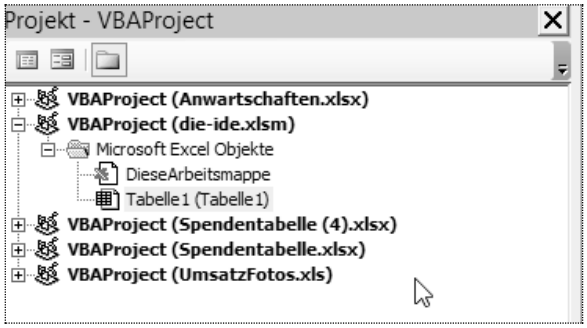
Alle Fenster der Entwicklungsumgebung können Sie über ANSICHT ein- oder ausblenden.

Im Eigenschaftsfenster legen Sie die Eigenschaften von Tabellen, Formularen und anderen Elementen der Excel-Arbeitsmappe fest oder können diese einsehen. Näheres dazu folgt in den Abschnitten „Module erstellen und bearbeiten“ auf Seite 8. Im Direktfenster, auch Direktbereich oder Testfenster genannt, können Sie Befehle direkt eingeben und ausführen – Sie können aber auch Ausgaben zur Kontrolle der Anwendung machen lassen.

Der Projektexplorer

Mit dem Projektexplorer verwalten Sie das VBA-Projekt. Hier können Sie auf Module zugreifen und diese manipulieren oder neue Module einfügen.

Module enthalten den VBA-Quellcode der Anwendung und werden als Eintrag im Projektexplorer angezeigt.



Die Einträge im Projektexplorer werden als Baumstruktur dargestellt. Ganz oben steht immer der Name des VBA-Projekts, in Klammern dahinter der Name der Arbeitsmappe – z.B. „die-die.xlsm“. Auch geladene Add-Ins werden so angezeigt. Diesen Einträgen sind die Inhalte des VBA-Projekts untergeordnet. Sie sind nach Typen gruppiert. Die erste Gruppe fasst alle Excel-Objekte (in Word wären es Word-Objekte) zusammen. Jedes Excel-Tabellenblatt verfügt über ein Modul und wird daher hier aufgeführt. Enthält die Arbeitsmappe Diagramme, werden diese ebenfalls in der Gruppe „Microsoft Excel Objekte“ angezeigt.

Einfache Module, die Sie erzeugen können, um Code zu erstellen, der unabhängig von einem Tabellenblatt ist, werden in einer Rubrik „Module“ aufgeführt, die mit dem ersten Modul erzeugt wird. Darüber hinaus kann es Klassenmodule und UserForms (Formulare) geben. Beide werden später noch genauer erläutert.

Möchten Sie ein Modul öffnen, das im Projektexplorer angezeigt wird, klicken Sie dazu doppelt auf den Eintrag.

Es gibt VBA-Projekte, etwa die von AddIns, die geschützt sind. Falls Sie zur Kennworteingabe aufgefordert werden, können Sie das Modul nur mit gültigem Kennwort öffnen; ansonsten wird der Inhalt des Moduls rechts neben dem

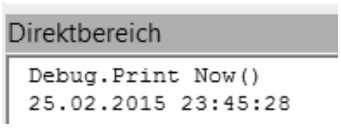
Projektexplorer im Modulfenster angezeigt.

Anweisungen im Direktfenster ausführen

Das Direktfenster ist vor allem anfänglich ein sehr nützliches Hilfsmittel. Sie können dort Anweisungen testen oder Werte ausgeben, die Sie zur Kontrolle des VBA-Codes benötigen.

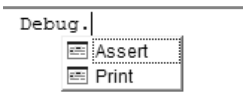
Das können Sie gleich testen:

1. Setzen Sie den Cursor in das Direktfenster, indem Sie einfach hineinklicken.
2. Geben Sie den Code `Debug.Print Now()` ein, und drücken Sie [ENTER].



Die Entwicklungsumgebung führt nun diesen Befehl aus. Zunächst wird die Anweisung `Now()` ausgewertet. Dabei handelt es sich um eine Funktion, die das aktuelle Datum und die aktuelle Uhrzeit zurückgibt. Dieser zurückgegebene Wert wird mit `Debug.Print` im Testfenster ausgegeben, was zur Folge hat, dass in der nächsten Zeile Datum und Uhrzeit erscheinen.

Vielleicht haben Sie bei der Eingabe des Befehls gemerkt, dass nach dem Punkt ein kleiner Tooltipp erscheint. Dies ist die integrierte Programmierhilfe IntelliSense. Sie hilft Ihnen bei der Eingabe des Codes, indem sie die verfügbaren Befehle zur Auswahl anbietet. Sie können mit der [TAB]-Taste und den Pfeiltasten einen Eintrag wählen und ihn mit [ENTER] in den Code übernehmen.



Neben der Elementliste gibt es noch weitere Programmierhilfen, wie z.B. die Parameterliste. Sie wird später im Abschnitt „Module erstellen und bearbeiten“ auf Seite 8 näher erläutert.

Die VBA-Hilfe aufrufen

Neben der Programmierhilfe liefert natürlich auch die VBA-Hilfe zusätzliche Informationen. Sie können Sie nur aus der Entwicklungsumgebung (=IDE, vom englischen Begriff *integrated development environment*) aufrufen. Im Excel-Anwendungsfenster erscheint beim Aufruf der Hilfe mit [F1] die Hilfe zu Excel, nicht die zu VBA. Wenn Sie die VBA-Hilfe aufrufen möchten, gibt es dazu zwei Möglichkeiten.

- Sie drücken einfach [F1], dann wird die allgemeine VBA-Hilfe angezeigt.
- Alternativ können Sie auch den Cursor in einen VBA-Befehl stellen und dann [F1] drücken. In diesem Fall wird die Hilfe zu dem aktuellen VBA-Befehl angezeigt.

In Excel 2010 und 2013 wird die Hilfe online über den Browser geladen. Das heißt: sind Sie nicht mit dem Internet verbunden, steht leider auch die Hilfe nicht zur Verfügung. Zudem handelt es sich bei der Online-Hilfe leider um teilweise automatisch übersetzte Texte, was dann auch schon mal dazu führt, dass die VBA-Befehle fälschlicherweise mit übersetzt werden.

Die Hilfe zu einem bestimmten Befehl

Möchten Sie beispielsweise die Hilfe zum `Debug`-Befehl aufrufen gehen Sie folgendermaßen vor:

- Geben Sie im Direktbereich den Befehl `Debug` ein und setzen Sie den Cursor in das Wort. Steht der Befehl bereits im Testfester, können Sie auch einfach den Cursor in das vorhandene Wort setzen ohne es neu einzugeben.
- Drücken Sie [F1].

Auf der linken Seite der Website, in der die Hilfeinformationen angezeigt werden, können Sie nun in aller Regel zwischen den verfügbaren weiteren verwandten Elementen von VBA wählen, bzw. zu den anderen Objekten, der Sprachreferenz Informationen aufrufen.

Hilfe durchsuchen

Wenn Sie gezielte Informationen zu anderen Themen als speziellen VBA-Befehlen suchen, können Sie dazu in der Hilfe das Suchfeld nutzen.

> Kombinieren Sie in diesem Fall immer die Suchbegriffe mit dem Suchwort „VBA", weil Sie sonst auch Suchergebnisse zu anderen Programmiersprachen angezeigt bekommen.

Gehen Sie folgendermaßen vor, um beispielsweise Informationen zum Thema „Variablen" zu bekommen:

1. Öffnen Sie die VBA-Hilfe, indem Sie in der Entwicklungsumgebung [F1] drücken.
2. Geben Sie rechts oben in das Suchfeld den Suchbegriff ein, beispielsweise `Variablen VBA` und drücken Sie [Enter].
3. Klicken Sie in der Ergebnisliste auf das gewünschte Suchergebnisse.

Module erstellen und bearbeiten

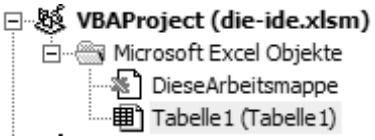
VBA-Quellcode wird in Modulen gespeichert. Die maximale Anzahl Codezeilen in einem Modul ist begrenzt, allerdings werden Sie erst bei sehr großen Anwendungen an diese Grenzen stoßen. Für den Anfang können Sie dies also vernachlässigen.

> Grundsätzlich gibt es in einem VBA-Projekt zwei Typen von Modulen: einfache Module und Klassenmodule.

Nicht alle Befehle sind in beiden Modultypen zulässig; bei den meisten ist es jedoch egal, in welchem Modultyp Sie sie verwenden. Einfache Module, der Name sagt es, stellen die einfachste Art von Modulen dar. Sie sind fast universell verwendbar.

> Nachfolgend werden immer einfache Module verwendet – es sei denn es wird explizit darauf hingewiesen, dass ein Klassenmodul notwendig ist.

Klassenmodule sind eine besondere Form von Modulen. Für jedes Tabellenblatt der Arbeitsmappe und die Arbeitsmappe selbst, erzeugt Excel automatisch ein solches Klassenmodul, das Sie dann im Projekt-Explorer in der Gruppe MICROSOFT EXCEL OBJEKTE angezeigt bekommen. Aus diesen Klassenmodulen erzeugt Excel zur Laufzeit ein Objekt, das in diesem Fall das Tabellenblatt darstellt.



Sie können aber auch eigene Klassenmodule erstellen und aus diesen Objekte ableiten. Klassenmodule entsprechen den Klassen anderer objektorientierter Programmiersprachen.

Sie können sich eine Klasse wie eine Art Schablone vorstellen, mit der Sie gleichartige Objekte erstellen können.

Jedes Objekt, dass Sie aus einer Klasse erzeugen, hat alle durch die Klasse bestimmte Eigenschaften, unterscheidet sich aber dennoch von jedem anderen Objekt, das aus der Klasse abgeleitet wurde, durch die Werte und Ausprägungen der Eigenschaften.

Die Ableitung eines Objektes aus einer Klasse wird Instanziierung genannt. Objekte werden alternativ auch als Instanzen bezeichnet.

Nehmen Sie an, Sie haben eine Schablone für Kreise. Dann bestimmt diese Schablone die Größe des Kreises und seine Form, eben einen Kreis. Jeder Kreis, den Sie mit der Schablone zeichnen, ist also gleich groß – und da es ein Kreis ist, sieht er damit auch immer gleich aus. Dennoch unterscheidet sich jeder gezeichnete Kreis von jedem anderen, mindestens durch die Position auf dem Blatt; es handelt sich jeweils um eine individuelle Instanz.

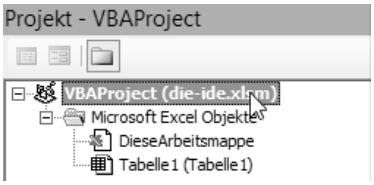
In der Programmierung entspricht dies der Position im Hauptspeicher des Rechners, an dem das Objekt gespeichert wird. Möglicherweise unterscheiden sich die Kreise aber zusätzlich in der Farbe, in der sie gemalt wurden oder der Linienstärke durch den verwendeten Stift. Dies sind wiederum Werte oder Ausprägungen von Eigenschaften, die der Kreis sonst noch hat.

Klassenmodule benötigen Sie nur für ganz spezifische Vorhaben. Sie werden im Abschnitt „Eigene Benutzeroberflächen gestalten“ auf Seite 48 Näheres dazu erfahren.

Ein Modul erzeugen und benennen

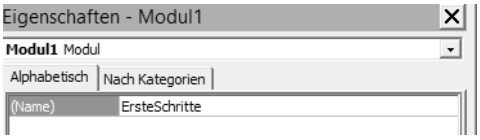
Wenn Sie ein Modul erstellen möchten, gehen Sie wie folgt vor:

1. Stellen Sie sicher, dass die Arbeitsmappe, in der Sie das Modul speichern möchten, im „xls“-Format (altes Excel-Format) oder im „xlsm“-Format gespeichert ist. Falls das nicht der Fall ist, speichern Sie die Arbeitsmappe mit „Datei/Speichern unter“ in diesem Format ab.
2. Wechseln Sie in das Fenster der Entwicklungsumgebung oder öffnen Sie die Entwicklungsumgebung mit [Alt] + [F11], falls diese noch nicht geöffnet ist.
3. Falls mehr als eine Arbeitsmappe geöffnet ist, markieren Sie im Projektexplorer die Arbeitsmappe, der Sie das Modul hinzufügen möchten, indem Sie den entsprechenden Eintrag anklicken.



4. Wählen Sie nun EINFÜGEN/MODUL.
5. Setzen Sie den Cursor in das Feld NAME des Eigenschaftenfensters.
6. Überschreiben Sie den vorhandenen Namen, zum Beispiel „Modul1“ durch einen Namen Ihrer Wahl, wie etwa „ErsteSchritte“.

7 Schließen Sie die Eingabe mit [Enter] ab.



Damit haben Sie ein leeres Modul erzeugt und benannt.

Wichtig ist, dass alle Module innerhalb eines Projektes einen eindeutigen Namen haben. Sie können also nicht zwei oder mehr Modulen den gleichen Namen geben.

Ein Modulname darf keine Leer- und Sonderzeichen enthalten und sollte auch keine Umlaute oder ein „ß“ enthalten. Lediglich ein Unterstrich ist als Sonderzeichen zulässig.

Code eingeben

Wenn Sie nun in ein Modul Code eingeben möchten, geht das wie folgt:

- Klicken Sie im Projektexplorer doppelt auf das Modul, damit es als Modulfenster geöffnet wird.
- Klicken Sie an die Stelle im Modul, an der Sie den Code eingeben möchten, z.B. an den Anfang des Moduls