einfach gelernt!

Dieter Klein, Inge Baumeister

VBA-Kochbuch

für Excel-Anwender

- *Praxistaugliche VBA-Rezepte für den Alltag*
- *Schnelle und effiziente Lösungen zum Nachschlagen*
- *Mit zahlreichen Beispielen und Tipps – auch zum Download*

Original BILDNER 20 Jahre Schulungserfahrung geprüft & bewährt

VBA-Kochbuch für Excel-Anwender

Dieter Klein, Inge Baumeister

Verlag:
BILDNER Verlag GmbH
Bahnhofstraße 8
94032 Passau

http://www.bildner-verlag.de
info@bildner-verlag.de

ISBN: 978-3-8328-5455-3

Autoren: Dieter Klein, Inge Baumeister
Herausgeber: Christian Bildner

Bildquellen:
Cover: © contrastwerkstatt - stock.adobe.com
Kapitelbild: © mehaniq41 - stock.adobe.com
S. 155, Kaffeetasse: © Inge Baumeister
S. 333, Hundefotos: © Dieter Klein

Auf einen Blick

Über dieses Buch

Ein Kochbuch für VBA? Was zunächst abwegig erscheinen mag, trifft bei näherer Betrachtung genau das Ziel unseres Buches, nämlich ein Nachschlagewerk für VBA-Einsteiger mit Grundkenntnissen und für Gelegenheitsprogrammierer bereitzustellen.

Wenn man nicht täglich Programmzeilen schreibt und nur gelegentlich Aufgaben per VBA zu lösen hat, fehlen oft nur kleine Bausteine und Tipps zur Herangehensweise, z. B. „Wie stelle ich den Umfang meiner Tabelle fest?“ oder „Wie sortiere ich Tabellenblätter?“ Wir bieten daher statt fertiger Programme eine Sammlung von Grundrezepten für den Alltag an, aus denen Sie nach Belieben eigene Lösungen kreieren können. Zusätzlich für Einsteiger sind die Kapitel drei und vier gedacht. Diese stellen in verkürzter Form auch die Grundzutaten, sprich die grundlegenden Sprachelemente von VBA vor, beispielsweise den Aufbau von Abfragen und Schleifen.

Die Grundrezepte werden ergänzt um Spezialitäten für fortgeschrittenere Anwender, etwa das Erstellen eigener Dialoge, Tricks im Umgang mit Diagrammen und den Datenaustausch mit anderen Office-Anwendungen. Mit etwas Know-How ist auch das Einfügen einer Adresse in einem Word-Brief, die Übergabe eines Diagramms an PowerPoint oder das Versenden einer Arbeitsmappe mit Outlook schnell realisiert.

Da Fertiggerichte die manchmal umständliche Suche nach einem passenden Rezept erheblich beschleunigen, beispielsweise beim Formatieren von Zellen, haben wir auch den Makrorecorder berücksichtigt. Mit etwas Grundwissen lässt sich anschließend die eigene Lösung schnell um die aufgezeichneten Anweisungen ergänzen.

Download der Beispieldateien

Sämtliche Rezepte stehen zum sofortigen Nachkochen bereit. Sie erhalten die verwendeten Beispiele nach Kapiteln geordnet, kostenlos zum Download. Außerdem finden Sie zu jedem Kapitel noch einen Ordner Module mit den Makros als Textdateien (.rtf), diese können jederzeit mit Microsoft Word oder mit WordPad geöffnet und in eigene Module eingebaut werden.

Die Dateien zum Download finden Sie unter folgender Adresse:

www.bildnerverlag.de/00418

Viel Spaß und Erfolg mit dem Buch wünschen Ihnen
BILDNER Verlag und die Autoren

Inhalt

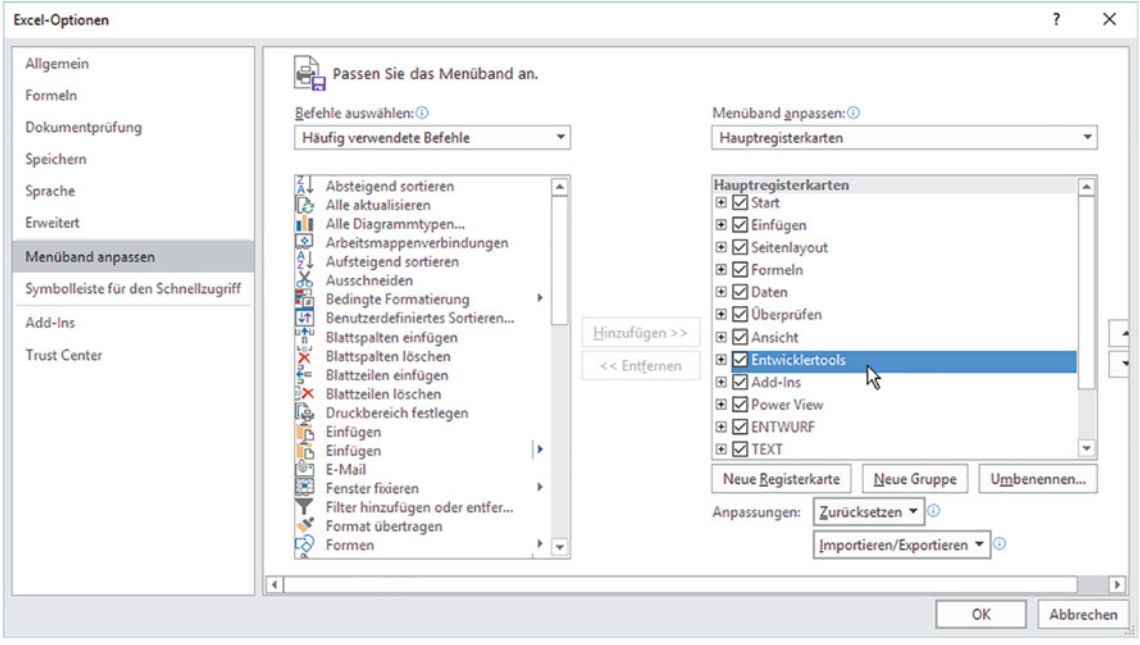
1 Grundeinstellungen

Die VBA Programmierung in Excel (und allen übrigen Microsoft-Office Anwendungen) findet in der Entwicklungsumgebung statt. Diese muss verfügbar gemacht und entsprechend Ihren Vorstellungen konfiguriert werden.

1.1 Entwicklertools im Menüband einbinden

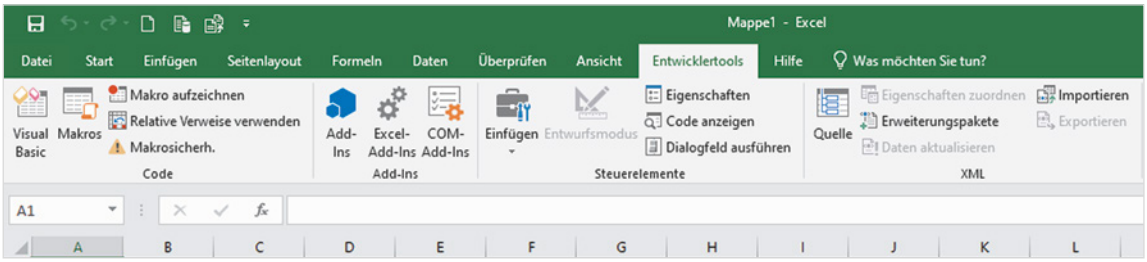
In Excel wird im Menüband die Registerkarte *Entwicklertools* benötigt: Zum Einblenden klicken Sie auf *Datei* ▶ *Optionen* ▶ *Menüband anpassen* oder Rechtsklick auf einen Registerkartenreiter im Menüband und den Befehl *Menüband anpassen*. Aktivieren Sie in den Excel-Optionen unter Hauptregisterkarten die *Entwicklertools* (Häkchen) und verlassen Sie die Optionen mit *OK*.

Menüband anpassen – Entwicklertools aktivieren



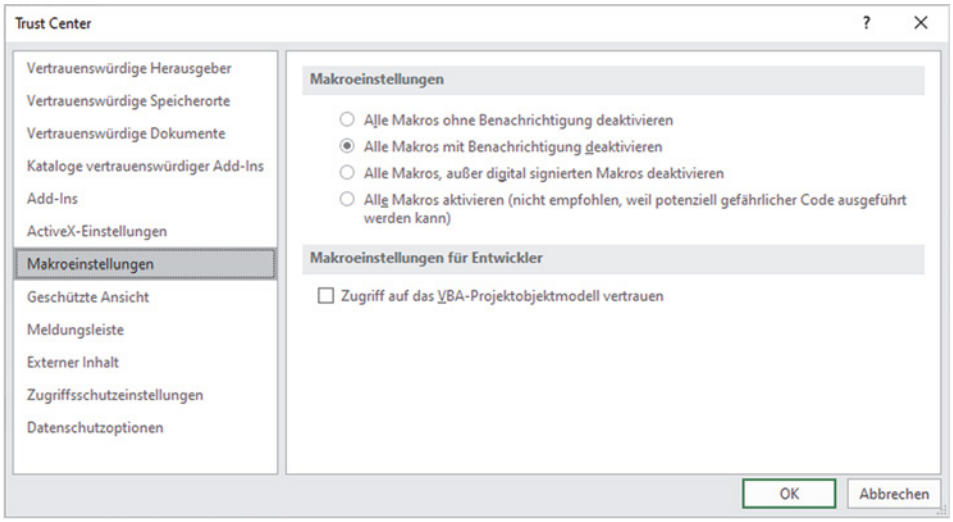
Nach dem Schließen der Excel-Optionen erscheint im Menüband das Register *Entwicklertools* mit den Entwicklerwerkzeugen. Dieses Register bleibt ab jetzt dauerhaft sichtbar, muss also nicht bei jedem Start neu eingeblendet werden

Register Entwicklertools im Menüband



1.2 Einstellungen im Excel-Sicherheitscenter (Trust Center)

In puncto Excel-Sicherheitseinstellungen müssen eigentlich keine Änderungen vorgenommen werden, die Standardeinstellung ist durchaus ausreichend. Sicherheitshalber sollten Sie diese aber dennoch kontrollieren: Sie gelangen dorthin entweder über die gerade eingerichtete Registerkarte *Entwicklertools* und die Schaltfläche *Makrosicherh.* oder über das Register *Datei* ▶ *Optionen* ▶ *Trust Center* ▶ *Einstellungen für das Trust Center…* (früher: *Sicherheitscenter*). Klicken Sie links auf *Makroeinstellungen*.



Die Einstellungen im Sicherheitscenter können unverändert bleiben

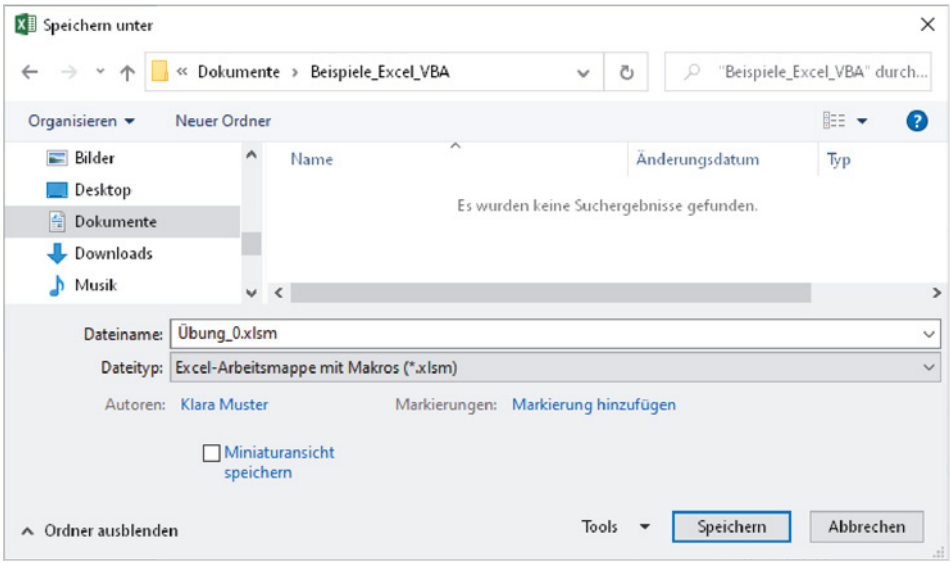
Empfehlung: Unter *Makroeinstellungen* sollte *Alle Makros mit Benachrichtigung deaktivieren* ausgewählt sein. Dies ist gleichzeitig auch die Standardeinstellung.

- Ein Heruntersetzen der Sicherheitsstufe (*Alle Makros aktivieren*) will gut überlegt sein! Zwar beschleunigt die Unterdrückung der Sicherheitsabfrage das Starten von .xlsm-Dateien, birgt aber bei Fremddateien ein erhöhtes Risiko für unliebsame Makros, die beim Öffnen der Datei automatisch gestartet werden und durchaus auch Schaden verursachen können.
- Den Haken bei *Zugriff auf das VBA-Projektobjektmodell vertrauen* können Sie weglassen.
- Über *OK* verlassen Sie das Trust Center, um in die normale Excel-Umgebung zurückzukehren.

1.3 Speichern der Arbeitsmappe mit Makros

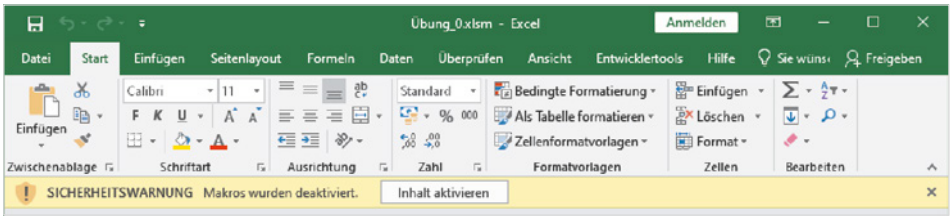
Speichern Sie Ihre Arbeitsmappe möglichst zu Beginn eines neuen VBA-Projektes und zwar unbedingt als *Excel-Arbeitsmappe mit Makros (*.xlsm)*! In der Entwicklungsumgebung besteht keine Möglichkeit, *Speichern unter* zu wählen. Dort können Sie zwar jederzeit über Strg+S oder das Symbol in der Menüleiste die aktuelle Situation speichern, nicht jedoch den Dateityp wählen.

Speichern der Arbeitsmappe mit Makros (xlsm)



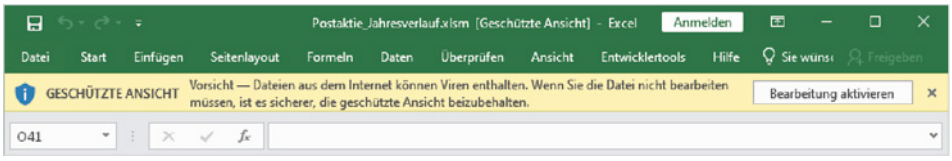
Wenn Sie eine Excel-Arbeitsmappe mit Makros zum ersten Mal öffnen, erfolgen Sicherheitsabfragen, entsprechend den Einstellungen im Trust Center. Sie können bei vertrauenswürdigen Quellen – meist sind es ja die eigenen Dateien – auf *Inhalt aktivieren* klicken. Nach dem ersten Öffnen und Speichern wird diese Datei künftig als vertrauenswürdig eingestuft, wenn sich Speicherort oder Dateiname nicht geändert haben.

Sicherheitsabfrage beim Öffnen von Mappen mit Makros



Stammt die Arbeitsmappe aus einer anderen Quelle, z. B. Netzwerk oder E-Mail Anhang, wird sie zunächst in der geschützten Ansicht geöffnet und kann nur gelesen werden. In diesem Fall müssen Sie zuvor noch auf *Bearbeitung aktivieren* klicken.

Geschützte Ansicht

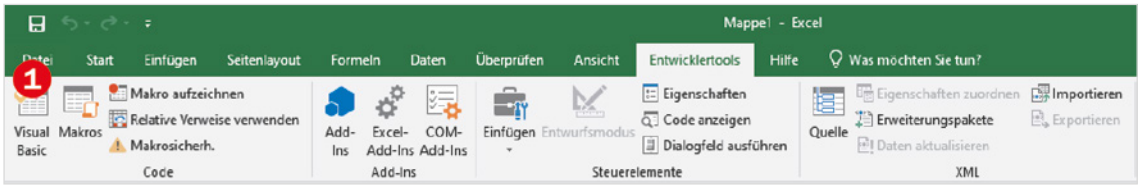


1.4 Die Entwicklungsumgebung (VBA-Editor)

Es gibt zwei Wege, um in die Excel-Entwicklungsumgebung zu gelangen:

- Über die Registerkarte *Entwicklertools* ▶ *Visual Basic* ❶ oder
- Mit der Tastenkombination **Alt** + **F11**.
  Hinweis: Auf manchen Rechnern kann diese Tastenkombination durch andere Anwendungen überlagert bzw. deaktiviert sein. Dann verwenden Sie die Tasten **Windows** + **Alt** + **F11**.

Entwicklungsumgebung öffnen

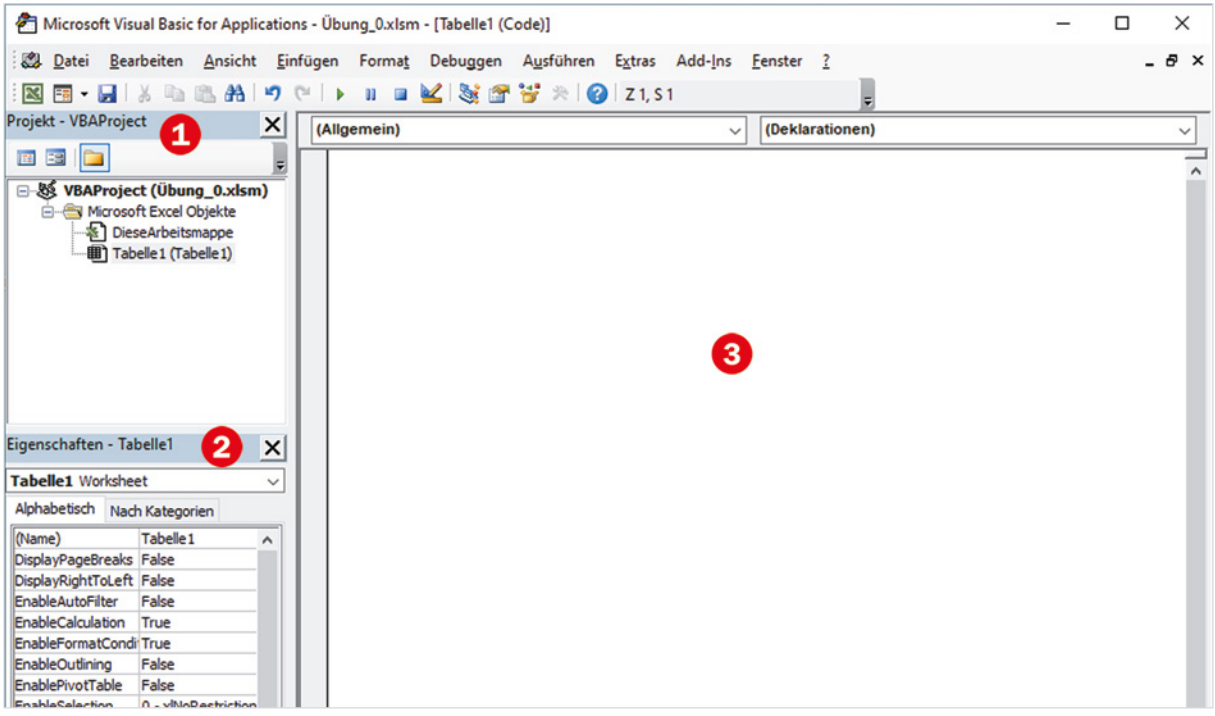


Da die VBA-Entwicklungsumgebung für gewöhnlich hinter der Excel-Oberfläche verborgen ist, nennen wir sie gerne auch Backstage-Bereich.

Fensteranzeige

Beim Öffnen des VBA-Editors präsentiert sich die Entwicklungsumgebung mit mehreren Fenstern, wie im Bild unten.

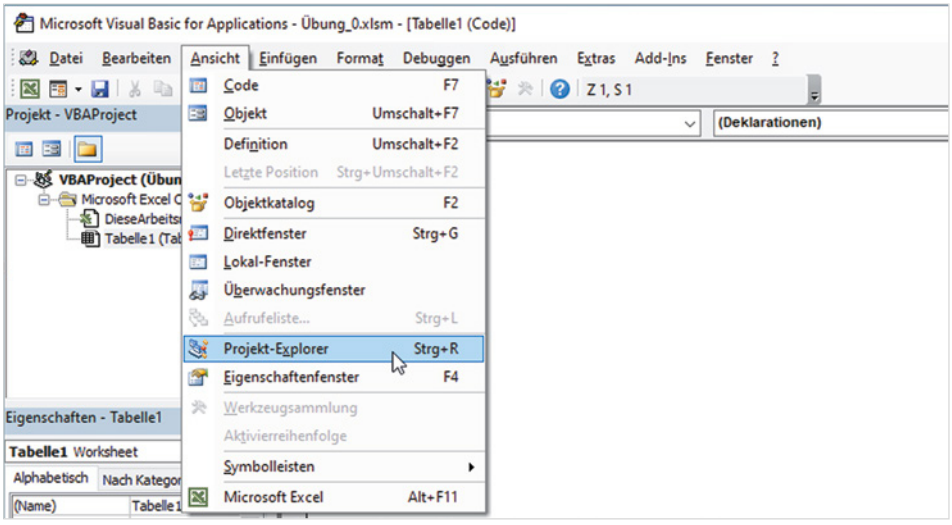
Die Entwicklungsumgebung mit Projektfenster, Eigenschaftenfenster und Codefenster



- Das **Projektfenster** ❶ zeigt alle zurzeit geöffneten Arbeitsmappen als Projekte an.
- Im **Eigenschaftenfenster** ❷ werden die (veränderlichen) Eigenschaften bzw. Besonderheiten der verwendeten Objekte dargestellt.
- Der Programmcode wird im **Codefenster** ❸ eingetragen, dem größten Fenster, das sich öffnet, sobald ein Modul oder ein Objekt (z. B. Tabelle, Formular) aktiviert (angeklickt) wird.
- Der **Direktbereich** – auch Direktfenster genannt und hier nicht abgebildet – dient als temporäres Ausgabefenster für Zwischenergebnisse oder kann zur Eingabe von Berechnungen, für Tests einzelner Programmzeilen oder Anweisungen verwendet werden.

Die einzelnen Fenster lassen sich nach Bedarf über die Menüleiste (*Ansicht*) ein- oder ausblenden. Sollten am linken Fensterrand die beiden Fenster *Projekt* und *Eigenschaften* nicht angezeigt werden, kommen Sie über die Einstellungen des Menüs *Ansicht* weiter, siehe Bild unten, oder die Tasten Strg+R bzw. F4. Die Fensteranordnung wie im Bild auf Seite 21 sollte als Standard eingestellt sein. Breite und/oder Höhe der Fenster lassen sich mit der Maus durch Verschieben der Begrenzungsleisten verändern.

Im Menü Ansicht lassen sich die Fenster ein- und ausblenden

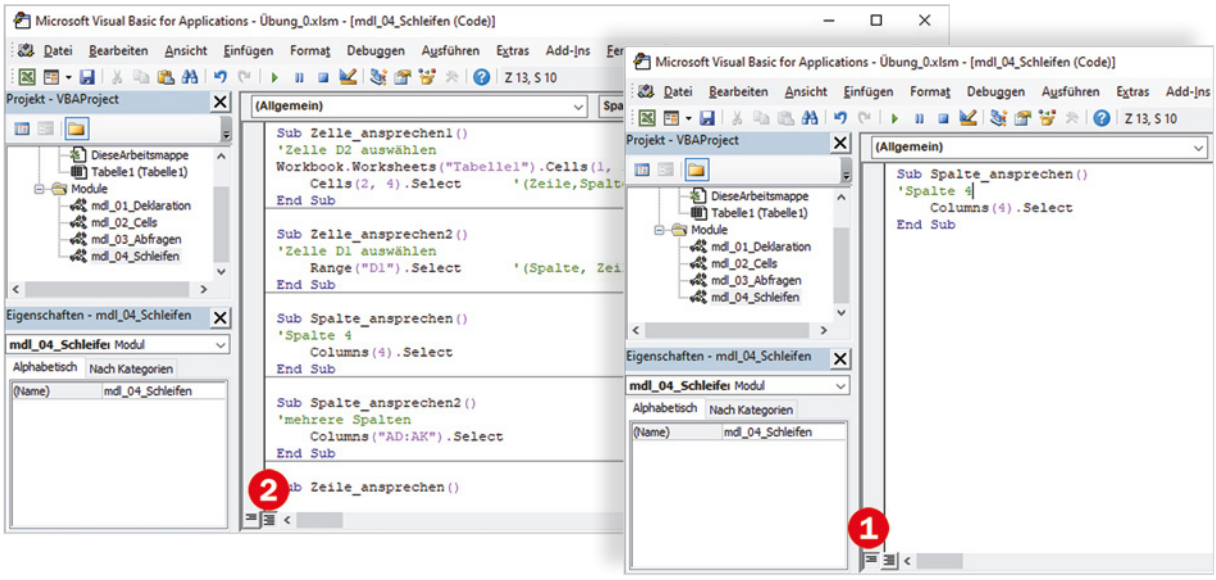


Hinweis: Vom Loslösen der am linken Fensterrand verankerten Fenster ist abzuraten, da das erneute Fixieren an angestammter Position umständlich und zeitraubend ist.

Das Codefenster

Das Codefenster des aktiven Moduls kann quasi nie zu groß sein, denn dort werden wir uns am meisten aufhalten. Hier werden standardmäßig alle, im Modul enthaltenen, Prozeduren bzw. Makros untereinander angezeigt. Falls Sie gezielt in einer bestimmten Prozedur arbeiten und die übrigen Prozeduren ausblenden wollen, dann klicken

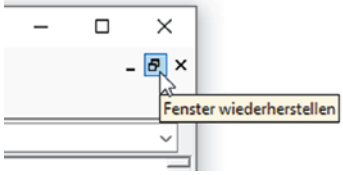
Sie in den Prozedurcode und anschließend auf das linke Symbol (*Prozeduransicht*) ❶ am unteren Rand des Codefensters. Ein Klick auf das Symbol rechts daneben ❷ zeigt wieder alle Prozeduren an.

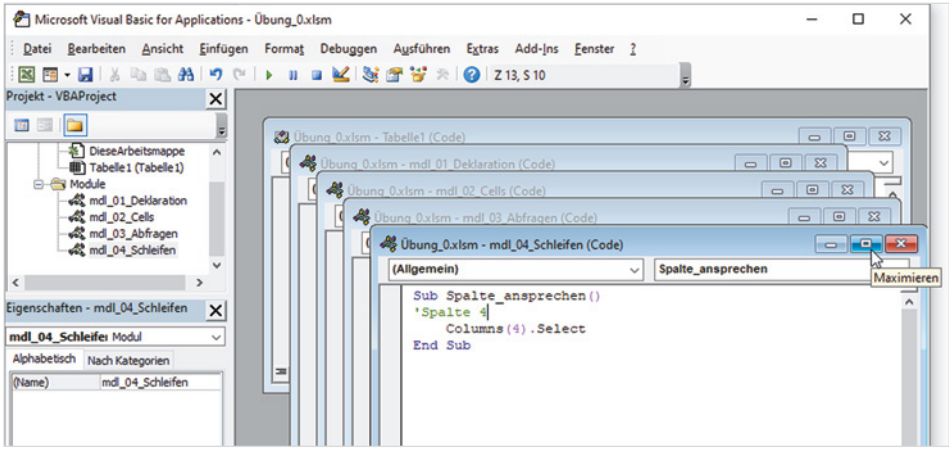


Alle Prozeduren (Standard)

Einzelansicht eines Makros

Prinzipiell lassen sich die Module auch in Fenstern nebeneinander oder überlappend anzeigen. Dazu klicken Sie auf das Symbol *Fenster wiederherstellen*. In der Praxis dürfte allerdings diese Darstellungsform kaum von Bedeutung sein. Um die Standardansicht wiederherzustellen, brauchen Sie nur eines der Fenster maximieren.



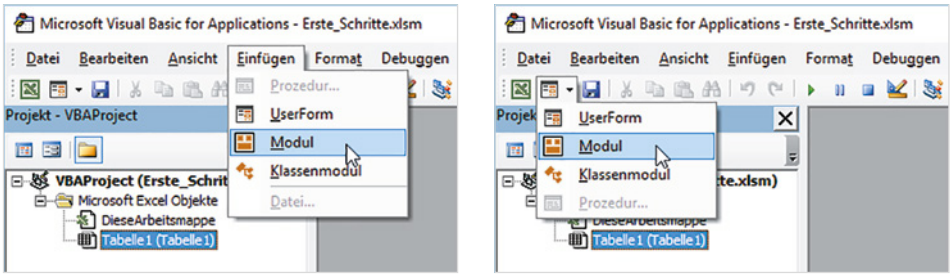


Überlappende Modulfenster im Codefenster

Module verwalten

Module sind Container für mehrere, meist sinngemäß zusammengehörige Prozeduren. Zum Erzeugen eines neuen Moduls klicken Sie auf das Menü *Einfügen* ▶ *Modul*. Oder klicken Sie in der Symbolleiste auf den Dropdown-Pfeil des Symbols *Einfügen* und wählen Sie hier *Modul*.

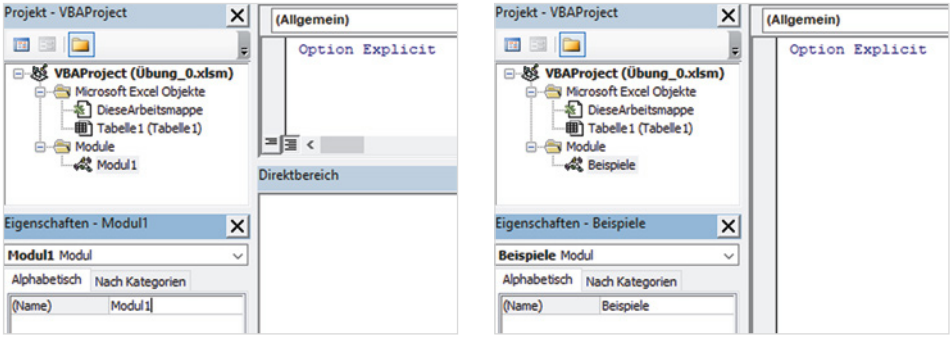
Modul einfügen



Das Modul wird mit dem Namen *Modul1* in den Ordner *Module* eingefügt. Falls noch kein Modul existiert, wird der Ordner *Module* automatisch erzeugt.

Anschließend sollten Sie jedes Modul mit einem aussagefähigen Namen versehen. Dazu klicken Sie im Projektfenster auf das Modul und geben im Eigenschaftenfenster unter *Name* einen Namen ein, im Bild unten *Beispiele* statt *Modul1*. Sollte rechts das dazugehörige Codefenster nicht sichtbar sein, so genügt zum Anzeigen ein Doppelklick auf das Modul im Projektfenster.

Modul umbenennen



Direktbereich

Anzeigen mit Strg+G

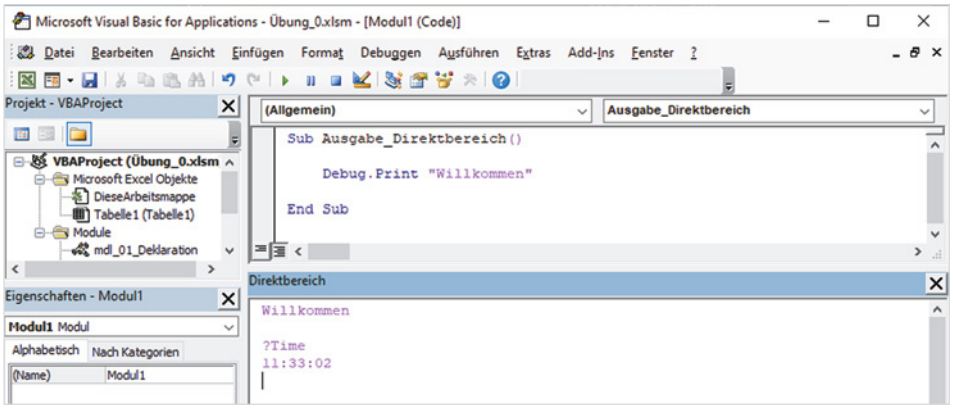
Das Fenster Direktbereich wird mit der Tastenkombination Strg+G oder den Menübefehl *Ansicht* ▶ *Direktfenster* eingeblendet und erscheint am unteren Rand der VBA-Entwicklungsumgebung unterhalb Codefensters.

Eine Ausgabe in diesem Fenster wird im Prozedurcode durch folgende Anweisung veranlasst:

```
Debug.Print
```

Im Direktbereich selbst kann ein Fragezeichen die Anweisung einleiten und anschließendes Betätigen der Enter-Taste bewirkt die Ausgabe; beispielsweise die Ausgabe der aktuellen Uhrzeit.

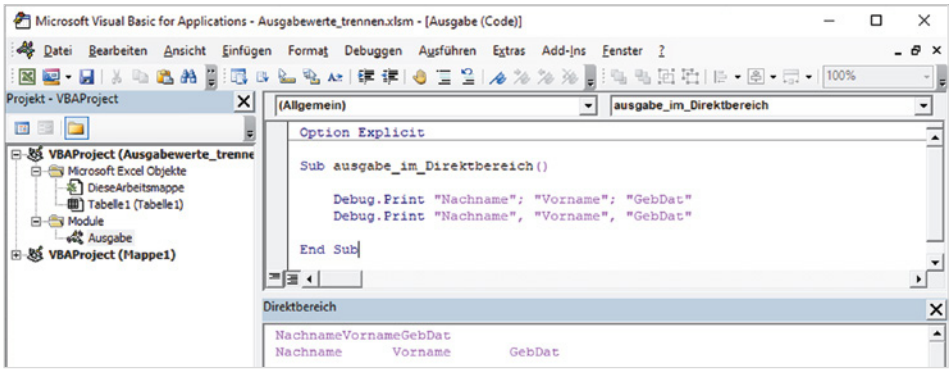
```
Debug.Print "Willkommen"
? Time
```



Ausgabe im Direktbereich

Ausgabewerte trennen

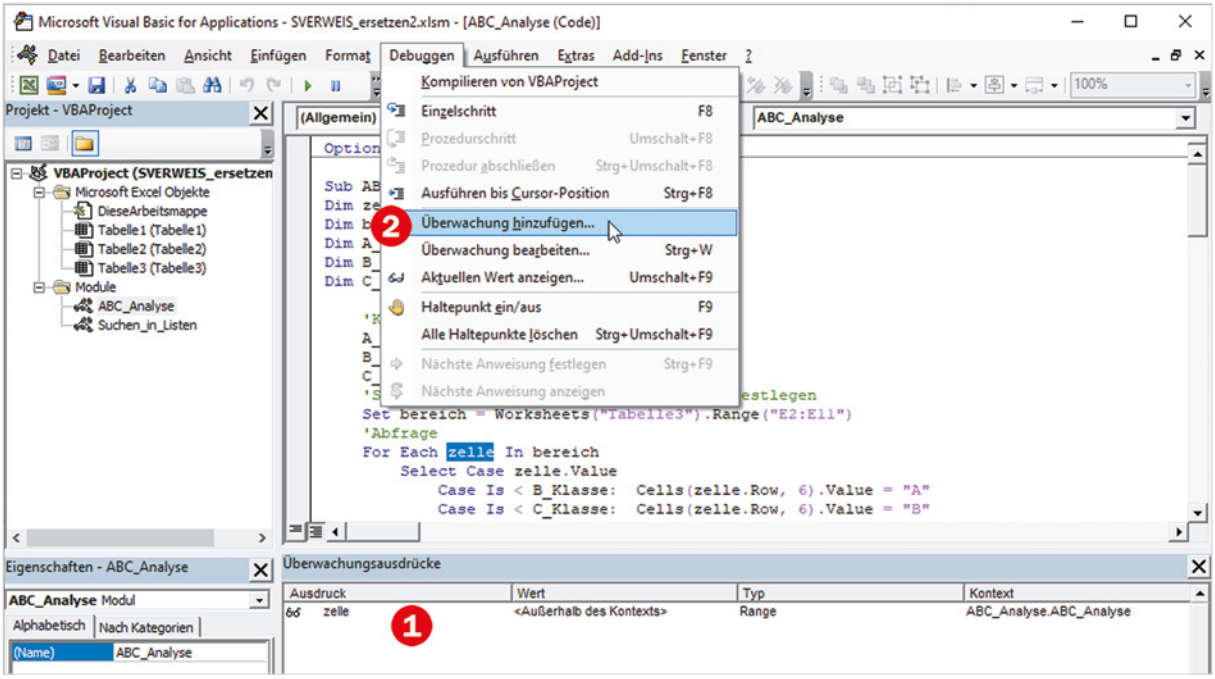
Werden in einer Debug.Print-Zeile mehrere Ausgabewerte durch Semikolon (;) getrennt, erhalten Sie im Direktbereich eine Ausgabe mit Leerzeichen getrennt. Die Trennung durch Komma in der Codezeile bewirkt die Ausgabe mit Tab-Abständen.



Getrennte Ausgabewerte

Das Überwachungsfenster

Wenn Sie während des Ablaufs einer Prozedur Variablen überprüfen wollen, um beispielsweise bei bestimmten Bedingungen den Programmlauf zu unterbrechen, steht Ihnen das Überwachungsfenster ❶ mit seinen Möglichkeiten zur Verfügung. Um eine Variable zu überwachen, markieren Sie die Variable im Programmcode und über einen Rechtsklick erhalten Sie den Befehl *Überwachung hinzufügen…* (oder über das Menü *Debuggen* ❷).



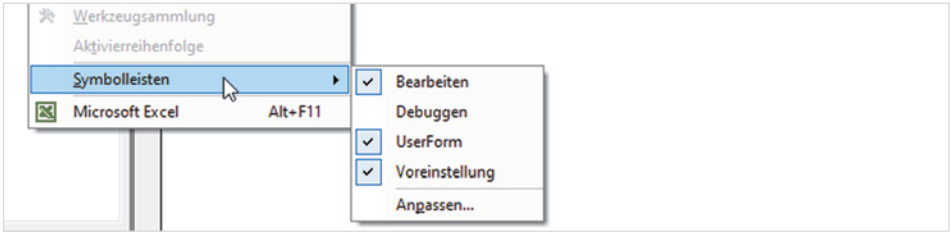
Überwachung einer Variablen hinzufügen

1.5 Einstellungen im VBA-Editor

Symbolleisten einblenden

Zu den Grundeinstellungen zählt auch die Anzeige und Anordnung der Symbolleisten. Aktivieren Sie über das Menü *Ansicht* ▶ *Symbolleisten* diese drei Symbolleisten (Bild unten), falls ausreichend Platz auf Ihrem Bildschirm ist.

Hilfreiche Symbolleisten



- Die Symbolleiste *Bearbeiten* bietet hilfreiche Zusatzfunktionen wie beispielsweise zum Einrücken (Tab) und *Auskommentieren*.

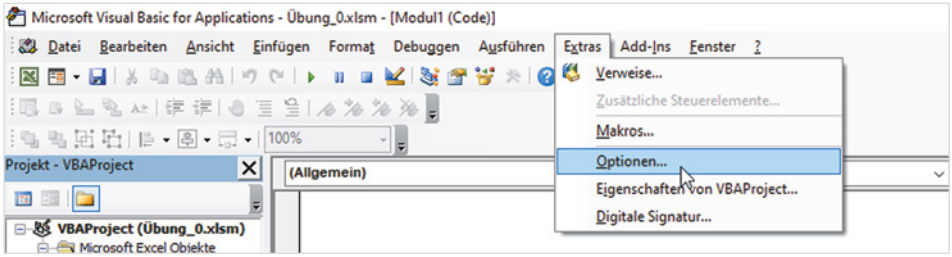
Symbolleiste Bearbeiten

- Beim Erstellen von Formularen (UserForms) kann die Symbolleiste *UserForm* beim Ausrichten und Aufteilen von Steuerelementen unterstützen.

Symbolleiste UserForm

Editier-Optionen

Einige Einstellungen der Entwicklungsumgebung sollten Sie unbedingt anpassen. Das zuständige Fenster öffnen Sie über *Extras* ▶ *Optionen…*.



Extras Optionen

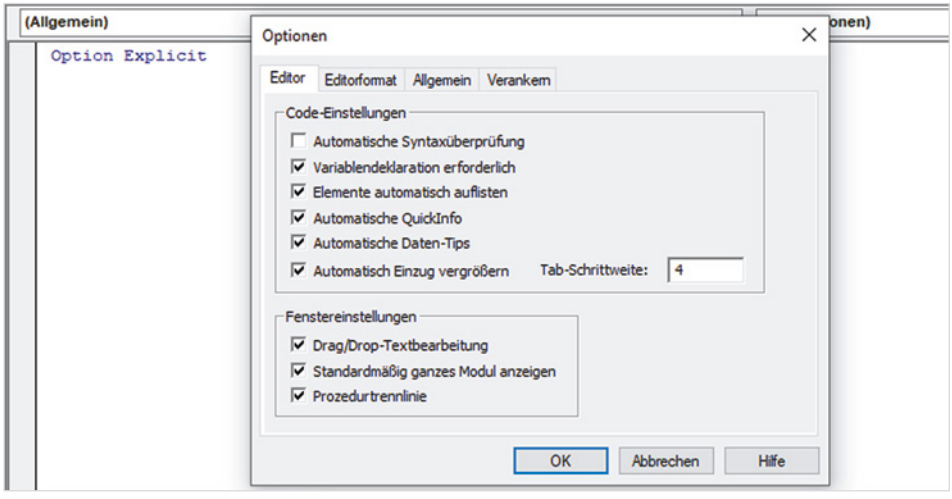
Variablendeklaration erzwingen

Diese Maßnahme zählt zu den wichtigen Grundeinstellungen und wird daher hier besonders hervorgehoben. In VBA müssen Variablen, die im Programm verwendet werden, nicht unbedingt vorab deklariert werden. Das Erzwingen der Variablendeklaration erhöht jedoch die Übersichtlichkeit im Makro, reduziert den Speicherplatz auf das Notwendigste durch Anpassung der von ihnen benötigten Byte-Tiefe und hilft, Tippfehler bei der Eingabe vermeiden. Die deklarierten Variablen werden Ihnen dadurch auch über die Tasten Strg+Leertaste angeboten (Auswahlliste *IntelliSense*).

Sie erzwingen die Deklaration von Variablen mit folgender Anweisung am Beginn eines Moduls:

```
Option Explicit
```

Damit diese Anweisung nicht jedes Mal erneut eingegeben werden muss, aktivieren Sie in den Optionen, Register *Editor*, das Kontrollkästchen *Variablendeklaration erforderlich*.

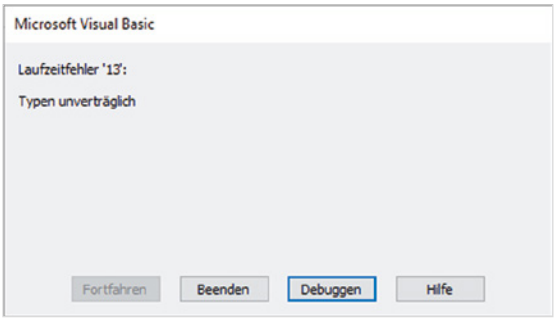


Variablendeklaration erforderlich als Standardeinstellung

Automatische Syntaxüberprüfung

Die *Automatische Syntaxüberprüfung* (siehe Bild unten können Sie problemlos deaktivieren. Syntaxfehler werden ohnehin auffällig angezeigt und können bei einer Fehlermeldung über *Debuggen* anschließend im Programmcode korrigiert werden.

Bei Fehlermeldung über Debuggen zum VBA-Code



Alle weiteren Einstellungen des Registers *Editor* können unverändert bleiben.

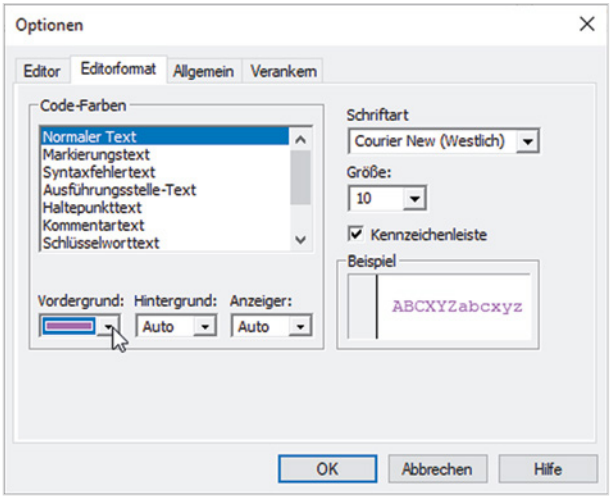
Schrifteinstellungen

Im Register *Editorformat* der Optionen können Sie verschiedene Schrifteinstellungen vornehmen. Die Standardschrift Courier New ist gut lesbar und zeichnet sich durch konstante Buchstabenbreiten aus, was das strukturierte/versetzte Schreiben von Codezeilen übersichtlich macht. Sie sollte daher beibehalten werden, die Schriftgröße können Sie dagegen ggf. verändern.

Als Schriftfarbe ist für Kommentare, d. h. Anweisungen, die mit Hochkomma ' beginnen, Grün bereits voreingestellt.

Empfehlung: Ändern Sie die Farbe für *Normalen Text* im Feld *Vordergrund* in ein kräftiges Pink.

Auffällige Farbe für Normalen Text im Programmcode



2 Prozeduren und Makros allgemein

2.1 Module und Prozeduren erzeugen

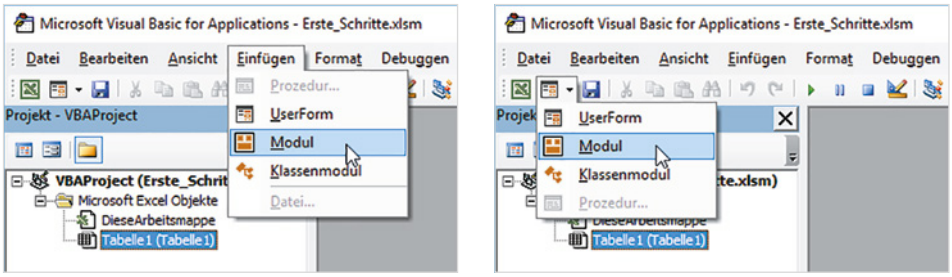
Prozeduren, auch als Makros bezeichnet, sind kleine, eigenständige Programmeinheiten mit einem eindeutigen Namen, unter dem sie später aufgerufen werden. Eine Prozedur beginnt immer mit der Anweisung *Sub*, gefolgt vom eigentlichen Namen und endet mit *End Sub*.

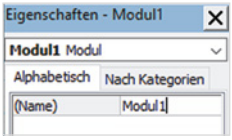
Prozedur erstellen

Modul einfügen

Module sind Container für mehrere, meist sinngemäß zusammengehörige Prozeduren. Zum Erzeugen eines neuen Moduls klicken Sie auf das Menü *Einfügen* ▶ *Modul*. Oder klicken Sie in der Symbolleiste auf den Dropdown-Pfeil des Symbols *Einfügen* und wählen Sie hier *Modul*.

Modul einfügen





Das Modul wird mit dem Namen *Modul1* in den Ordner *Module* eingefügt. Falls noch kein Modul existiert, wird der Ordner *Module* automatisch erzeugt. Zur besseren Übersicht sollten Sie anschließend das Modul im Eigenschaften-Fenster umbenennen.

Prozedur erzeugen

Eine neue Prozedur erzeugen Sie, indem Sie im Codefenster an die gewünschte Stelle klicken, *Sub* und den Prozedurnamen über die Tastatur eingeben und mit der Eingabetaste abschließen. *End Sub* wird automatisch hinzugefügt und dadurch ein sogenannter Prozedurrumpf erzeugt, in den die weiteren Anweisungen eingefügt werden.

Prozedurrumpf

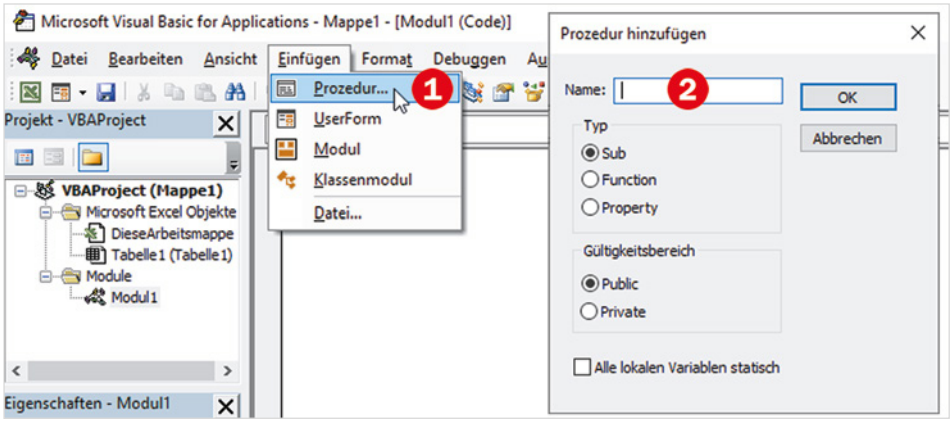
```
Option Explicit

Sub ErstesBeispiel()

End Sub
```

Beachten Sie: Prozedurnamen müssen eindeutig sein und dürfen keine Sonder- oder Leerzeichen enthalten. Die automatisch hinzugefügten Klammern sind zur Übernahme von Argumenten (Parameterwerten) vorgesehen und lassen sich nicht entfernen.

Oder klicken Sie auf *Einfügen* ▶ *Prozedur...* ❶ und geben anschließend einen Prozedurnamen ein ❷. Hier können Sie optional noch den Gültigkeitsbereich festlegen.



Prozedur einfügen

Gültigkeitsbereiche von Prozeduren

Wird beim Erzeugen einer neuen Prozedur einfach `sub` gefolgt vom Prozedurnamen eingegeben, ist die Prozedur automatisch *Public*, d. h. sie kann aus jeder anderen Prozedur innerhalb des Projekts heraus aufgerufen werden. Dies ist auch die Voreinstellung beim Erzeugen einer Prozedur über das Menü *Einfügen*. Optional können Sie auch der Prozedur das Schlüsselwort Public voranstellen.

```
Public sub Testbeispiel_1()

End Sub
```

Wenn Sie dagegen dem Prozedurkopf *Private* voranstellen, kann die Prozedur nur innerhalb des Moduls aufgerufen werden.

```
Private sub Testbeispiel_2()

End Sub
```

2.2 Der Makrorecorder als Programmierhilfe

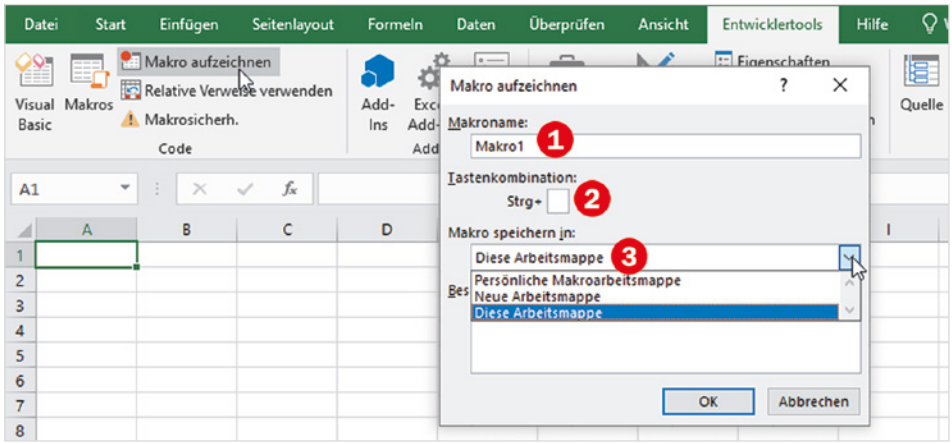
VBA ist eine sehr umfangreiche Sprache und gerade Einsteiger oder Gelegenheitsprogrammierer haben nicht immer sofort die passende Anweisung parat. Hier kann der Makrorecorder eine gute Hilfe sein, ohne die Online-Hilfe bzw. F1 oder Internet-Foren in Anspruch nehmen zu müssen. Durch Aufzeichnen der benötigten Aktion erhalten Sie in vielen Fällen schnell die richtigen Anweisungen.

Nachteil: Der Makrorecorder generiert neben der benötigten Anweisung häufig auch eine Vielzahl überflüssiger Codezeilen, die zwecks besserer Übersichtlichkeit manuell gelöscht werden sollten.

Aufzeichnen mit dem Makrorecorder

Klicken Sie im Tabellenblatt, Register *Entwicklertools* auf *Makro aufzeichnen*. Geben Sie einen Namen ein ❶, unter dem das Makro später aufgerufen werden soll, optional auch eine Tastenkombination ❷ und legen Sie den Speicherort ❸ des Makros fest. Nachdem Sie auf *OK* geklickt haben, läuft die Aufzeichnung.

Makro aufzeichnen



Achtung Speicherort: Standardvorgabe im Feld *Makro speichern in* ist *Diese Arbeitsmappe*. Dies ist in 99 % aller Fälle auch sinnvoll, da die meisten Makros auf spezielle Aufgaben der jeweiligen Arbeitsmappe zugeschnitten sind. Wenn Sie dagegen *Persönliche Makroarbeitsmappe* wählen, dann ist das Makro in jeder Excel-Arbeitsmappe verfügbar. Allerdings dürfen Sie beim Beenden von Excel in diesem Fall nicht vergessen, dass Ihre Änderungen in der persönlichen Makroarbeitsmappe extra gespeichert werden müssen. Klicken Sie also bei der entsprechenden Rückfrage auf *Ja*.

Umschalten auf relative Verweise

Wechsel zwischen festen und relativen Zellbezügen

Der Makrorecorder bietet zwei Aufzeichnungsmöglichkeiten an, zwischen denen Sie jederzeit wechseln können. Standardeinstellung ist die Aufzeichnung mit festen Verweisen (Zellbezügen). Mit Klick auf *Relative Verweise verwenden* schalten Sie vor oder während der Aufzeichnung relative Verweise ein und wieder aus.

Aufzeichnung beenden

Aufzeichnung beenden

Zum Beenden der Aufzeichnung klicken Sie auf *Entwicklertools* ▶ *Aufzeichnung beenden* oder links unten in der Statusleiste auf *Beenden*.

Beispiel: Zeilen und Spalten vertauschen

Die Adressliste

	A	B	C	D
1	1			
2	Hein			
3	Buche			
4	Bahnhofstr. 1			
5	32023			
6	Hameln			
7	2			
8	Fritz			
9	Box			
10	Hauptstr. 34			
11	56834			
12	Mückendorf			
13	3			
14	Hans			
15	Dampf			
16	Schulstr. 3			
17	49702			
18	Hannover			
19				

Eine Adresse liegt in der unten abgebildeten Form vor und soll im selben Arbeitsblatt in eine Tabelle umgewandelt werden, in der Name und Adressangaben in Spalten nebeneinander angeordnet sind. Dazu müssen die zweite bis sechste Adresszeile horizontal neben dem Namen angeordnet werden. Manuell bieten sich spontan zwei Vorgehensweisen an:

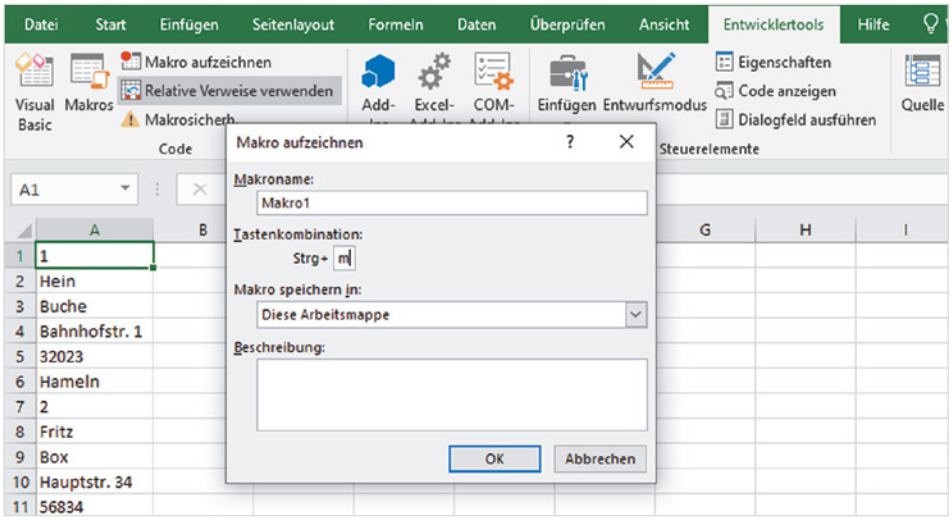
- Zellen einzeln ausschneiden und in horizontaler Reihenfolge einfügen, danach die leeren Zellen in Spalte A löschen.
- Oder Zellen als Block kopieren und transponiert horizontal ab Spalte B einfügen und danach die ursprünglichen Zellen in Spalte A löschen.

Die Handarbeit beider Methoden kann mittels *Makrorekorder* aufgezeichnet werden. Liegen mehrere Adressen in dieser Form vor, lassen sich die Einzelschritte später mehrfach wiederholen. Voraussetzung ist eine Aufzeichnung mit Relativen Verweisen.

Variante 1: Zellen einzeln ausschneiden

Das Makro wird zunächst nur für die erste Adresse in A1:A6 aufgezeichnet.

Markieren Sie die Zelle A1, diese dient als Ausgangspunkt und aktivieren Sie *Relative Verweise verwenden*. Dem Makro kann auf diesem Weg sofort eine Tastenkombination zugewiesen werden; beispielsweise Strg+m, die noch nicht vergeben ist. Nach dem Klick auf *OK* werden alle Aktionen als VBA-Code aufgezeichnet.



Markieren Sie A1 und starten Sie die Makroaufzeichnung

Adressen_Makro.xlsm

Führen Sie folgende Aktionen nacheinander aus:

1 Klick in *Zelle A2* und Inhalt ausschneiden, z. B. mit Strg+X.

2 Klick in *Zelle B1* und Inhalt einfügen, z. B. mit Strg+V.

3 Wiederholen Sie diese Schritte bis einschließlich A6.

4 Löschen Sie die Zellen A2:A6 mit der Option *Zellen nach oben verschieben*.

5 Klicken Sie auf A2, um die nächste Zelle auszuwählen.

6 Beenden Sie die Aufzeichnung.

Nach dem Wechsel in die Entwicklerumgebung mit Alt+F11 finden Sie im Projektfenster im Ordner *Modul1* das soeben aufgezeichnete Makro *Makro1*.

Das aufgezeichnete Makro

```
Sub Makro1()
'
' Makro1 Makro
'
' Tastenkombination: Strg+m
'
    ActiveCell.Offset(1, 0).Range("A1").Select
    Selection.Cut
    ActiveCell.Offset(-1, 1).Range("A1").Select
    ActiveSheet.Paste
    ActiveCell.Offset(2, -1).Range("A1").Select
    Selection.Cut
    ActiveCell.Offset(-2, 2).Range("A1").Select
    ActiveSheet.Paste
    ActiveCell.Offset(3, -2).Range("A1").Select
    Selection.Cut
    ActiveCell.Offset(-3, 3).Range("A1").Select
    ActiveSheet.Paste
    ActiveCell.Offset(4, -3).Range("A1").Select
    Selection.Cut
    ActiveCell.Offset(-4, 4).Range("A1").Select
    ActiveSheet.Paste
    ActiveCell.Offset(5, -4).Range("A1").Select
    Selection.Cut
    ActiveCell.Offset(-5, 5).Range("A1").Select
    ActiveSheet.Paste
    ActiveCell.Offset(1, -5).Range("A1:A5").Select
    Selection.Delete Shift:=xlUp
    ActiveCell.Select
End Sub
```

Zur Erklärung

A1	Bezugszelle
Offset(1,0): Cut	1 Zeile tiefer, gleiche Spalte: Ausschneiden
Offset(-1,1): Paste	1 Zeile höher, 1 Spalte nach rechts: Einfügen
Offset(2,-1): Cut	2 Zeilen tiefer, 1 Spalte nach links: Ausschneiden
Offset(-2,2): Paste	2 Zeilen höher, 2 Spalten nach rechts: Einfügen
...	
Offset(1,-5): Delete	1 Zeile tiefer, 5 Spalten nach links: 5 Zellen markieren und löschen
	Nächste Zelle markieren

Die nachfolgenden Adressen lassen sich nun einzeln nacheinander durch Aufruf des Makros mit der Tastenkombination Strg+m in Tabellenzeilen umwandeln. Allerdings muss dazu jeweils die erste Zelle der nächsten Adresse markiert sein!

Das Ergebnis im selben Tabellenblatt

	A	B	C	D	E	F	G	H
1	1	Hein	Buche	Bahnhofstr. 1	32023	Hameln		
2	2	Fritz	Box	Hauptstr. 34	56834	Mückendorf		
3	3	Hans	Dampf	Schulstr. 3	49702	Hannover		
4								
5								
6								

Unterschied relative und feste Verweise

Betrachten wir kurz den Unterschied zwischen festen und relativen Verweisen bei der Aufzeichnung, im beiden Fällen ist zu Beginn der Aufzeichnung A1 markiert und die aufgezeichnete Anweisungszeile markiert A2.