

Für alle  
Excel-  
Versionen



Dieter Klein, Inge Baumeister

einfach gelernt!

# 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*



# **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](mailto: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

© 2020 BILDNER Verlag GmbH Passau

Die Informationen in diesen Unterlagen werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht. Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt. Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden. Verlag, Herausgeber und Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen. Für Verbesserungsvorschläge und Hinweise auf Fehler sind Verlag und Herausgeber dankbar.

Fast alle Hard- und Softwarebezeichnungen und Markennamen der jeweiligen Firmen, die in diesem Buch erwähnt werden, können auch ohne besondere Kennzeichnung warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Die in den Beispielen verwendeten Namen von Firmen, Personen, Produkten und E-Mail-Adressen sind frei erfunden. Jede Ähnlichkeit ist keinesfalls beabsichtigt, sondern zufällig.

Das Werk einschließlich aller Teile ist urheberrechtlich geschützt. Es gelten die Lizenzbestimmungen der BILDNER Verlag GmbH Passau.

## Auf einen Blick

- 1 Grundeinstellungen 17
  - 2 Prozeduren und Makros allgemein 29
  - 3 Deklarieren und dimensionieren 63
  - 4 Abfragen und Wiederholungsschleifen 77
  - 5 Zellen und Bereiche 91
  - 6 Arbeitsblätter und Arbeitsmappen 115
  - 7 Die Anwendung Excel, Dialogelemente 143
  - 8 Zeichenfolgen in VBA 167
  - 9 Datum und Uhrzeit 181
  - 10 Konsolidieren, sortieren, filtern und durchsuchen 205
  - 11 Dateizugriff mit VBA 227
  - 12 Diagramme 241
  - 13 Farben mit VBA anwenden 279
  - 14 Performance 299
  - 15 Datenaustausch mit Office-Anwendungen 311
- Stichwortverzeichnis 341



# 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](http://www.bildnerverlag.de/00418)**

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



# Inhalt

## 1

### **Grundeinstellungen 17**

- 1.1 Entwicklertools im Menüband einbinden 18**
- 1.2 Einstellungen im Excel-Sicherheitscenter (Trust Center) 19**
- 1.3 Speichern der Arbeitsmappe mit Makros 20**
- 1.4 Die Entwicklungsumgebung (VBA-Editor) 21**
  - Fensteranzeige 21
  - Das Codefenster 22
  - Direktbereich 24
  - Das Überwachungsfenster 25
- 1.5 Einstellungen im VBA-Editor 26**
  - Symbolleisten einblenden 26
  - Editier-Optionen 27

## 2

### **Prozeduren und Makros allgemein 29**

- 2.1 Module und Prozeduren erzeugen 30**
  - Prozedur erstellen 30
  - Gültigkeitsbereiche von Prozeduren 31
- 2.2 Der Makrorecorder als Programmierhilfe 31**
  - Aufzeichnen mit dem Makrorecorder 32
  - Beispiel: Zeilen und Spalten vertauschen 33
- 2.3 Makros, Prozeduren und Formulare ausführen 37**
  - Im Dialogfenster Makros starten 37
  - Tastenkombination zuweisen 38
  - Der Schnellzugriffsleiste und/oder dem Menüband hinzufügen 38
  - Beim Öffnen der Arbeitsmappe ausführen 40
  - Prozedur aus anderen Prozeduren heraus aufrufen 41
  - Über Befehlsschaltflächen starten 42
  - Makro per Rechtsklick bzw. Kontextmenü ausführen 45
  - Makros im Menüband, Register Add-Ins integrieren 48



- 2.4 Programmausführung testen, Fehlerbehandlung 50**
  - Einzelschritte testen 50
  - Laufzeitfehler abfangen 51
- 2.5 Eigene Funktionen erstellen 53**
  - Aufbau von Funktionen 53
  - Einfache Funktionen ohne Parameter 54
  - Parameterübergabe an Funktionen 54
- 2.6 Weitergeben von Makros und Funktionen 57**
  - Über die Zwischenablage 57
  - Exportieren / Importieren 58
  - In der persönlichen Arbeitsmappe ablegen 59
  - Arbeitsmappe weitergeben und VBA-Projekt schützen 60
  - Als Add-In speichern/laden 61

# 3

## **Deklarieren und dimensionieren 63**

- 3.1 Variablen 64**
  - Deklaration erzwingen 64
  - Datentypen 65
  - Deklaration mit der Dim-Anweisung 66
  - Objektvariablen 66
  - Gültigkeitsbereiche 67
- 3.2 Namenskonventionen 69**
- 3.3 Operatoren 70**
- 3.4 Datenfelder 71**
- 3.5 Konstanten 72**
- 3.6 Enumerationsvariablen 74**

# 4

## **Abfragen und Wiederholungsschleifen 77**

- 4.1 Abfragen 78**
  - If ... Then-Anweisung 78
  - Die IIF-Funktion 80
  - Die Fallauswahl mit Select Case 80

- 4.2 Prüfabfragen 82**
  - Eingaben auf Leereingaben prüfen 82
  - Eingaben anhand einer vorgegebenen Anzahl Zeichen prüfen 83
  - Zeichenabfolge einer Eingabe überprüfen 84
- 4.3 Anweisungen wiederholen - Programmschleifen 85**
  - Die For...Next-Schleife 86
  - Verschachtelte For-Next-Schleifen 87
  - Die For...Each-Schleife 87
  - Die While...Wend-Schleife 88
  - Do...Loop-Schleifen 89
  - Zusammenfassung und Übersicht Programmschleifen 89

# 5

## Zellen und Bereiche 91

- 5.1 Zellen und Bereiche ansprechen 92**
  - Die Excel-Objekthierarchie 92
  - Zellen und Bereiche auswählen 93
  - Ganze Spalten und Zeilen 93
  - Zellbereiche 94
- 5.2 Zeilen und Spalten 96**
  - Ausblenden 96
  - Zeilen und Spalten einfügen 97
- 5.3 Löschmethoden 98**
  - Tabellenblatt löschen 100
  - Zeilen und Spalten löschen 100
  - Inhalte löschen 100
- 5.4 Verschieben oder kopieren 101**
- 5.5 Zellen formatieren 102**
  - Farben zuweisen 102
  - Schriftattribute 103
  - Zahlen-, Datums- und andere Formate 104
- 5.6 Tabellenbereiche 105**
  - Bereichsnamen vergeben 105
  - Tabellenbereich zur Eingabe eingrenzen 106
  - Tabellenumfang bzw. letzte Zeile/Spalte ermitteln 106

- 5.7 Zellinhalte 109**
  - Besondere Zelltypen (SpecialCells) 109
  - Nicht druckbare Zeichen entfernen (säubern) 111
  - Zellen mit gleichen Inhalten (Duplikate) markieren 112
- 5.8 Formeln in Zellen schreiben 112**
  - Bezüge in der A1-Schreibweise 112
  - Formeln mit flexiblem Zellbezug 113
  - Arbeitsblattfunktionen 114

# 6

## Arbeitsblätter und Arbeitsmappen 115

- 6.1 Arbeitsblätter zählen, auswählen, hinzufügen, verschieben 116**
  - Anzahl und Namen aller Arbeitsblätter ermitteln 116
  - Auswählen und Umbenennen 117
  - Arbeitsblatt einfügen und benennen 117
- 6.2 Arbeitsblätter adressieren 119**
  - Namen und Indizes 119
  - Blatt auswählen oder aktivieren 121
  - Prüfen, ob der Name eines Arbeitsblatts vorhanden ist 121
- 6.3 Arbeitsblätter verbergen, aus- und einblenden 123**
  - Einfaches Aus- oder Einblenden 123
  - Einblenden mit der Maus verhindern ("sehr versteckt") 123
  - Blattregister aus- und einblenden 124
- 6.4 Tabellenblätter sortieren 125**
  - Nach Namen 125
  - Nach Registerfarben 125
- 6.5 Tabellenblätter schützen, löschen, drucken und speichern 126**
  - Blatt schützen 126
  - Blatt löschen 127
  - Drucken 128
  - Speichern als Arbeitsmappe 128
- 6.6 Ereignisse in Arbeitsblättern nutzen 129**
  - Ereignisprozeduren 130
  - Ein Inhaltsverzeichnis über Arbeitsblätter einfügen 132
  - Das aktuelle Blattregister farblich hervorheben 133

- 6.7 Arbeitsmappen 133**
  - Dateiname und Pfad von Arbeitsmappen ermitteln 134
  - Neue Arbeitsmappe erstellen 134
  - Arbeitsmappe speichern 134
  - Arbeitsmappe als Kopie speichern 135
  - Arbeitsmappe schließen 136
- 6.8 Ereignisse von Arbeitsmappen 137**
  - Tabellenanzeige nach Passworteingabe 138
  - Arbeitsblatt mit dem aktuellen Monat hervorheben und aktivieren 140
  - Zugriffe auf Arbeitsmappe dokumentieren 141
  - Arbeitsmappe ohne Makroausführung öffnen 141

# 7

## Die Anwendung Excel, Dialogelemente 143

- 7.1 Das Application-Objekt Excel 144**
  - Informationen zu Excel erhalten 144
  - Warnhinweise abschalten, einschalten 145
  - Bildschirmaktualisierung deaktivieren 145
  - Excel beenden 145
- 7.2 Einfache Dialoge 146**
  - Meldung per Dialogfenster ausgeben (MsgBox) 146
  - Benutzereingaben mit InputBox einlesen 150
- 7.3 Die integrierten Dialogfenster Öffnen und Speichern 153**
- 7.4 Mit UserForms eigene Dialoge und Formulare erstellen 155**
  - UserForm einfügen und Werkzeugsammlung 155
  - Formular zur Kennwortabfrage 158
  - Steuerelemente abfragen 159
  - Listen- und Kombinationsfelder mit Werten füllen 160
  - Eingaben löschen 162
  - Werte von Optionsfeldern und Kontrollkästchen 163
  - Bild oder Logo einbauen 163
  - Steuerelemente verwenden 164
  - Steuerelement-Ereignisse 165
  - Starten eines Formulars 165

# 8

## Zeichenfolgen in VBA 167

- 8.1 Zeichenfolgen in Zahlen konvertieren 168**
  - Punkt als Dezimalzeichen durch Komma ersetzen 168
  - Konvertierungsfunktionen einsetzen 169
  - Behandlung leerer Zellen 171
- 8.2 Umgang mit Zeichenketten (Strings) 172**
  - Länge von Zeichenfolgen ermitteln 173
  - Leerzeichen entfernen 173
  - Zeichenfolgen auf eine feste Länge auffüllen 174
  - Teile einer Zeichenkette ermitteln 175
  - Zeichenketten durchsuchen 175
  - Zeichenketten mit der Split-Funktion trennen 176
  - Mehrere Zeichenfolgen aneinanderfügen (verketten) 179
  - Umwandlung in Groß- oder Kleinbuchstaben 179
  - Sonderzeichen verwenden 180

# 9

## Datum und Uhrzeit 181

- 9.1 Allgemeine Datums- und Zeitangaben 182**
  - Systemdatum und Systemzeit 182
  - Datum- und Uhrzeit formatieren 182
  - Teile von Datumswerten 184
- 9.2 Berechnungen mit Datum und Uhrzeit 186**
  - Datums- und Zeitwerte umwandeln 186
  - Addieren und subtrahieren 186
  - Zahlungsziel mit Berücksichtigung der Wochentage 187
  - Differenz zwischen zwei Datumswerten berechnen 189
  - Kalenderwoche und Quartal mit DatePart 191
  - Zeitmessungen mit der Timer-Funktion 191
- 9.3 Datentypen und Konvertierungsfunktionen 192**
  - Der Datentyp Date 192
  - Ausdrücke in Datum oder Uhrzeit umwandeln 193
  - Zeitangaben mit eigenen Funktionen umwandeln 195
  - Datum und Uhrzeit in Textfeldern (UserForms) 197
  - Termine mit Wochentaganzeige eintragen 199

- 9.4 Zeitdifferenzen berechnen 201**
  - Zeitdifferenz aus Datum über Mitternacht hinaus 201
  - Zeitdifferenz aus 4-stelligen Zeichenfolgen 202
  - Zeitdifferenz aus Formular-Textfeldern 203

# 10

## **Konsolidieren, sortieren, filtern und durchsuchen 205**

- 10.1 Tabellen zusammenführen (konsolidieren) 206**
  - Konsolidieren innerhalb einer Arbeitsmappe 206
  - Konsolidieren aus mehreren Arbeitsmappen 207
- 10.2 Sortieren 208**
  - Sortieren in Spalten 208
  - Benutzerdefiniertes Sortieren 209
  - Nach Farben sortieren 210
- 10.3 Tabellen filtern 214**
  - Filtern mit dem AutoFilter 214
  - Duplikate mit dem erweiterten Filter ausschließen 216
  - Tops und Flops filtern 217
  - Gefilterte Werte in eine neue Tabelle kopieren 217
  - Top-Werte mit der bedingten Formatierung hervorheben 218
- 10.4 Top-Werte und Rangfolge berechnen 219**
  - Top-Werte und Rangfolge als Formeln einfügen 219
  - Top-Werte und Rangfolge mit WorksheetFunctions berechnen 220
- 10.5 VBA statt SVERWEIS 221**
  - SVERWEIS als Formel einfügen 221
  - SVERWEIS in VBA durch Suche ersetzen 222
  - ABC-Analyse ohne SVERWEIS 224

# 11

## Dateizugriff mit VBA 227

- 11.1 Laufwerke und Ordner 228**
  - Laufwerke 228
  - Ordner verwalten 228
  - Ordnerinhalte 231
- 11.2 Dateien kopieren, löschen, umbenennen 234**
- 11.3 Per VBA auf Textdateien zugreifen 235**
  - Sequenzieller Zugriff 235
  - Neue Textdatei speichern 236
  - An Textdatei anfügen 236
  - Textdateien lesen 237
  - Zugriff auf Arbeitsmappe dokumentieren 239

# 12

## Diagramme 241

- 12.1 Mehrere gleichartige Diagramme automatisch erstellen 242**
- 12.2 Diagrammobjekte per VBA bearbeiten 248**
  - Chart und ChartObject 248
  - Diagrammtitel 249
  - Datenbereich anpassen 250
  - Achsoptionen 251
- 12.3 Datenreihen und Datenpunkte 253**
  - Farben zuweisen 253
  - Maximal- und Minimalwert hervorheben 254
  - Mittelwert als Linie einfügen 255
  - Datenpunkte mit Wert beschriften 256
  - Datenpunkte in Linien- und Punktdiagrammen 257
  - Mehrere Datenpunkte hervorheben 259
- 12.4 Spezialfälle 261**
  - Zahlen als Achsenbeschriftung hinzufügen 261
  - Min/Max und Intervalle der Y-Achse 262
  - Achsenformatierung XY-Diagramm 263
  - Skalierung der X-Achse auf 100 Prozent (XY-Diagramm) 264
  - Vertikale Linien einfügen (Zeitintervalle) 265
  - Horizontale Linien 267

- 12.5 Diagrammgröße anpassen 269**
  - Diagramm skalieren 269
  - Genaue Diagrammgröße und -position 270
- 12.6 Diagramme als Bilder exportieren 271**
  - Einzelne Diagramme exportieren 271
  - Export mehrerer Diagramme 272
  - Diagramm als Bildschirmkopie (Screenshot) speichern 273
  - Diagramm in ein Formular (UserForm) übertragen 274

# 13

## Farben mit VBA anwenden 279

- 13.1 Farben und Farbpaletten 280**
  - Der Color Farbcode 280
  - Die Excel-Farbkonstanten 280
  - Die ColorIndex-Farbpalette 281
  - Die Farbpalette der XIRgbColor-Enumeration 284
  - Die RGB-Funktion 285
  - Farbverläufe programmieren 288
- 13.2 Farbwerte ermitteln 288**
  - RGB-Farbwert aus ColorIndex berechnen 288
  - RGB-Farbwerte aus Bildvorlagen 289
- 13.3 Designfarben und individuelle Farben 290**
  - Designfarben mit VBA nutzen 290
  - Eigene Farbzusammenstellungen 292
  - Diagrammfarben zusammenstellen 294
  - Die ColorIndex-Farbpalette anpassen 297



# 14

## Performance 299

- 14.1 Makros beschleunigen - Performance steigern 300**
  - Nicht benötigte Voreinstellungen vorübergehend deaktivieren 300
  - Optimierungen im Programmcode 301
  - Filtermethoden anstelle von Abfrageschleifen 303
  - Bearbeitung großer Datenmengen 306
- 14.2 Laufzeitmessung 306**
  - Vergleich unterschiedlicher Kopiermethoden 307
  - Die Verwendung der Timer-Funktion 309

# 15

## Datenaustausch mit Office-Anwendungen 311

- 15.1 Die Objektbibliotheken 312**
  - Verweise in der Entwicklungsumgebung setzen 312
  - Verweis zur Laufzeit setzen 313
- 15.2 Microsoft Word 313**
  - Adresse aus Excel-Tabelle in einen Brief einfügen 313
  - Aus mehreren Briefvorlagen auswählen 318
  - Excel-Tabelle in ein Word-Dokument einfügen 321
  - Excel-Diagramm nach Word übergeben 325
- 15.3 Excel und PowerPoint 325**
  - Excel-Diagramme nach PowerPoint exportieren 325
  - Präsentation aus Excel-Tabelle erstellen 330
- 15.4 Excel und Outlook 335**
  - Formularinhalt versenden 335
  - Arbeitsblatt als E-Mail Anhang versenden 339

## Stichwortverzeichnis 341



# 1

## Grundeinstellungen

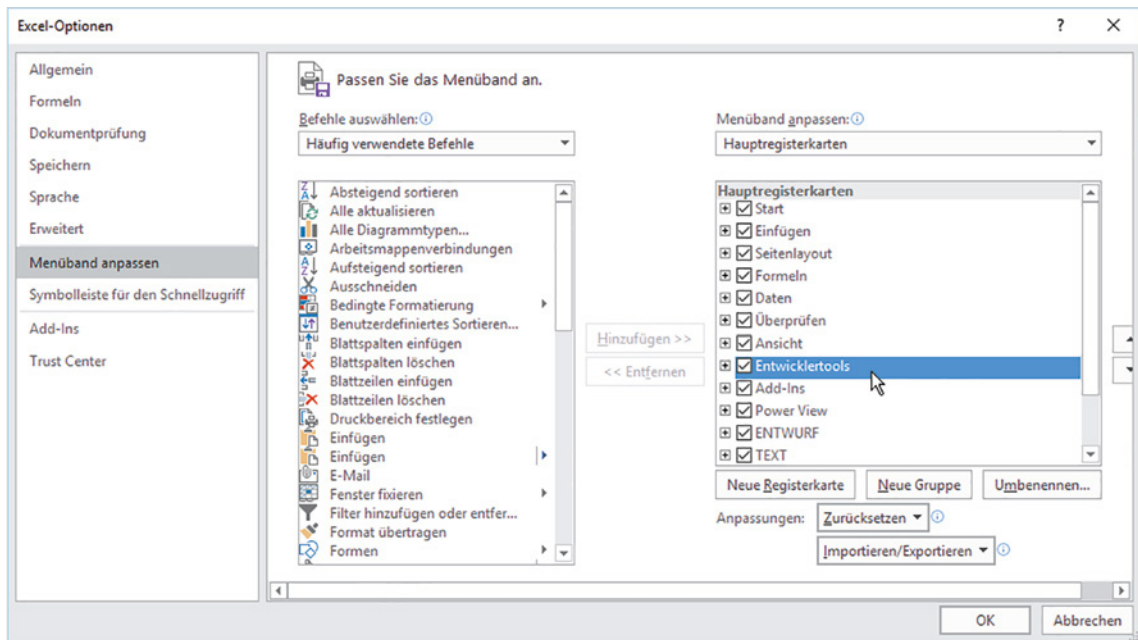
- 1.1 Entwicklertools im Menüband einbinden 18
- 1.2 Einstellungen im Excel-Sicherheitscenter (Trust Center) 19
- 1.3 Speichern der Arbeitsmappe mit Makros 20
- 1.4 Die Entwicklungsumgebung (VBA-Editor) 21  
Fensteranzeige; Das Codefenster; Direktbereich; Das Überwachungsfenster
- 1.5 Einstellungen im VBA-Editor 26  
Symbolleisten einblenden; Editier-Optionen

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

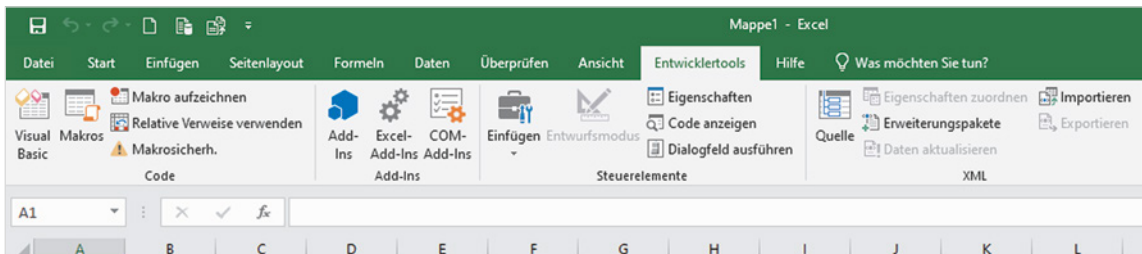
In Excel wird im Menüband die Registerkarte *Entwicklertools* benötigt: Zum Einblenden klicken Sie auf *Datei* ► *Optionen* ► *Menüband anpassen* oder Rechtsklick auf einen Registerkartenreiter im Menüband und den Befehl *Menüband anpassen*. Aktivieren Sie in den Excel-Optionen unter Hauptregisterkarten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

*Menüband anpassen –  
Entwicklertools aktivieren*



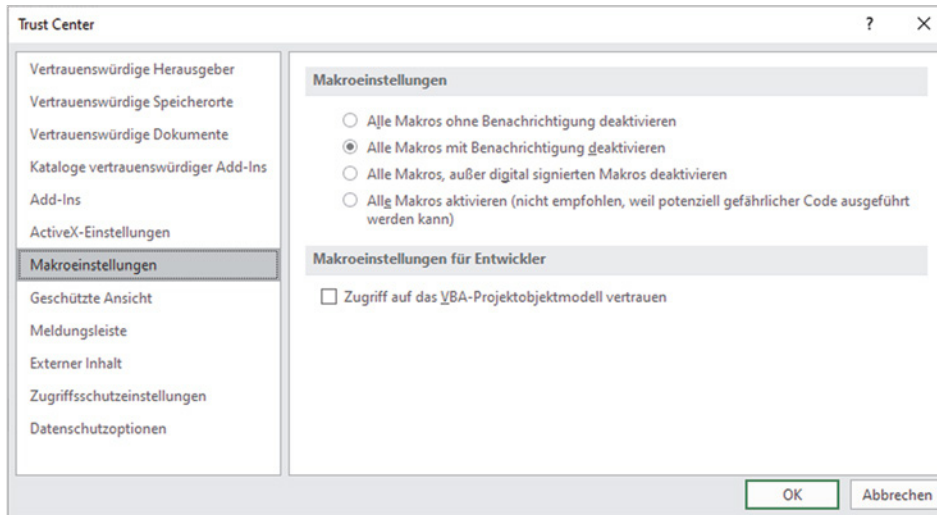
Nach dem Schließen der Excel-Optionen erscheint im Menüband das Register *Entwicklertools* mit den Entwicklerwerkzeugen. Dieses Register bleibt ab jetzt dauerhaft sichtbar, muss also nicht bei jedem Start neu eingeblendet werden

*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*

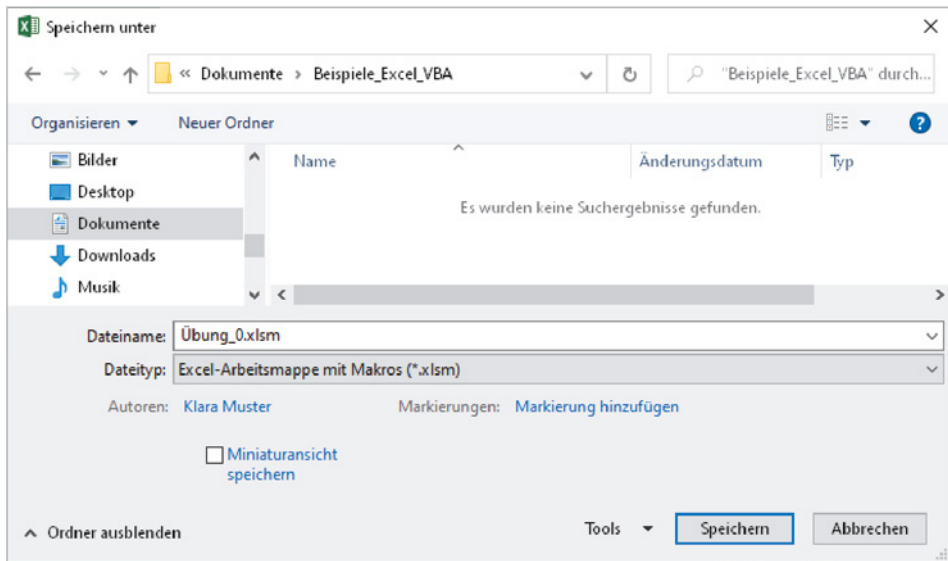
**Empfehlung:** Unter *Makroeinstellungen* sollte *Alle Makros mit Benachrichtigung deaktivieren* ausgewählt sein. Dies ist gleichzeitig auch die Standardeinstellung.

- ▶ 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

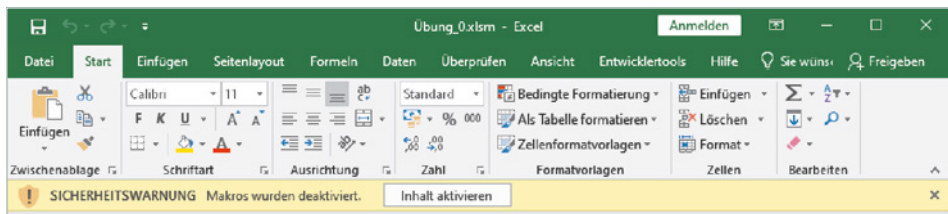
Speichern Sie Ihre Arbeitsmappe möglichst zu Beginn eines neuen VBA-Projektes und zwar unbedingt als *Excel-Arbeitsmappe mit Makros (\*.xlsm)*! In der Entwicklungsumgebung besteht keine Möglichkeit, *Speichern unter* zu wählen. Dort können Sie zwar jederzeit über Strg+S oder das Symbol in der Menüleiste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

*Speichern der Arbeitsmappe mit Makros (xlsm)*



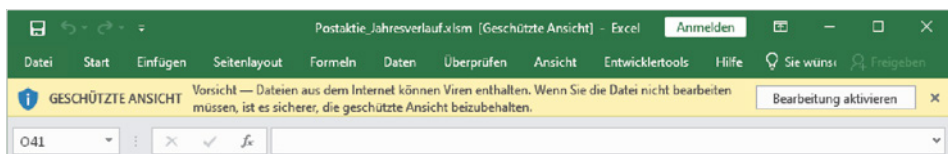
Wenn Sie eine Excel-Arbeitsmappe mit Makros zum ersten Mal öffnen, erfolgen Sicherheitsabfragen, entsprechend den Einstellungen im Trust Center. Sie können bei vertrauenswürdigen Quellen – meist sind es ja die eigenen Dateien – auf *Inhalt aktivieren* klicken. Nach dem ersten Öffnen und Speichern wird 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*



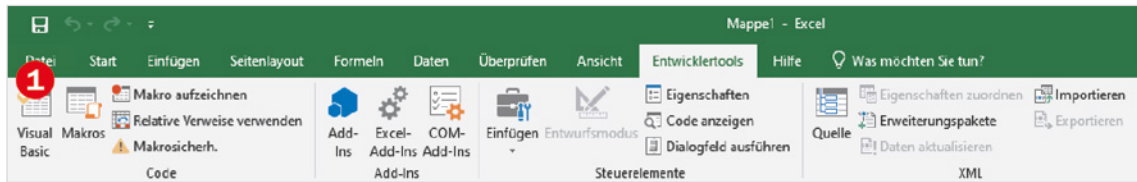
## 1.4 Die Entwicklungsumgebung (VBA-Editor)

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

- ▶ Über die Registerkarte *Entwicklertools* ▶ *Visual Basic* **1** 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*

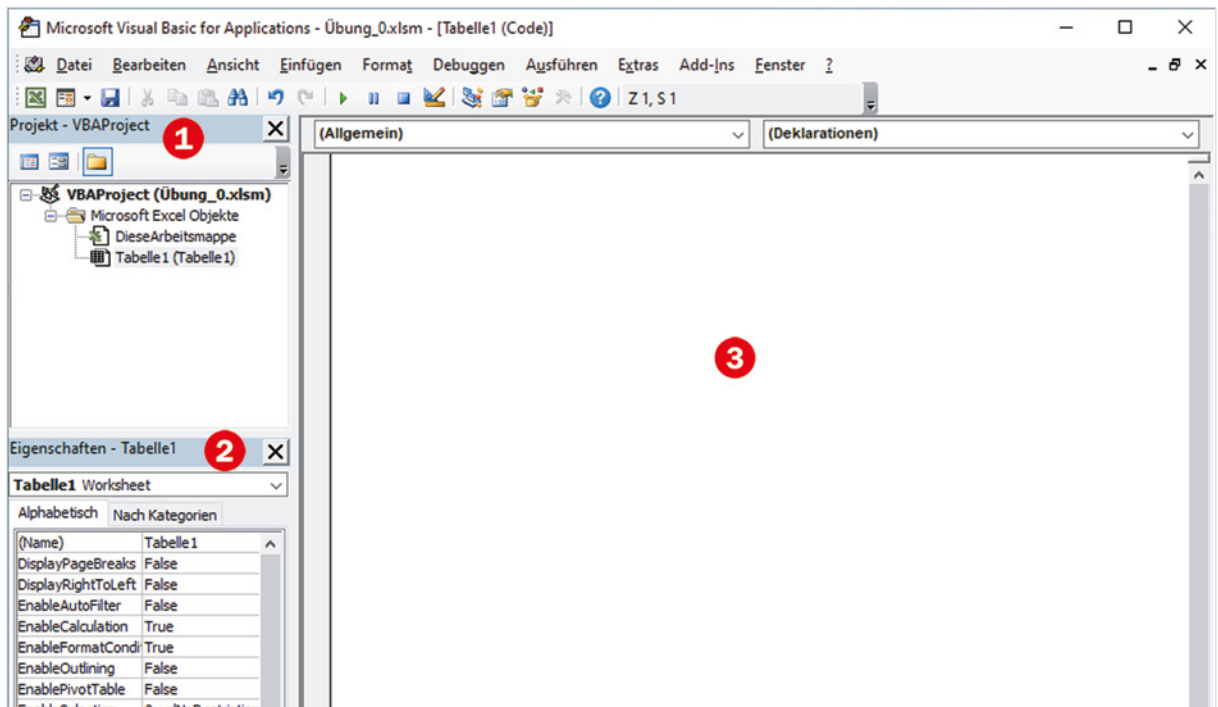


Da die VBA-Entwicklungsumgebung für gewöhnlich hinter der Excel-Oberfläche verborgen ist, nennen wir sie gerne auch Backstage-Bereich.

### Fensteranzeige

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

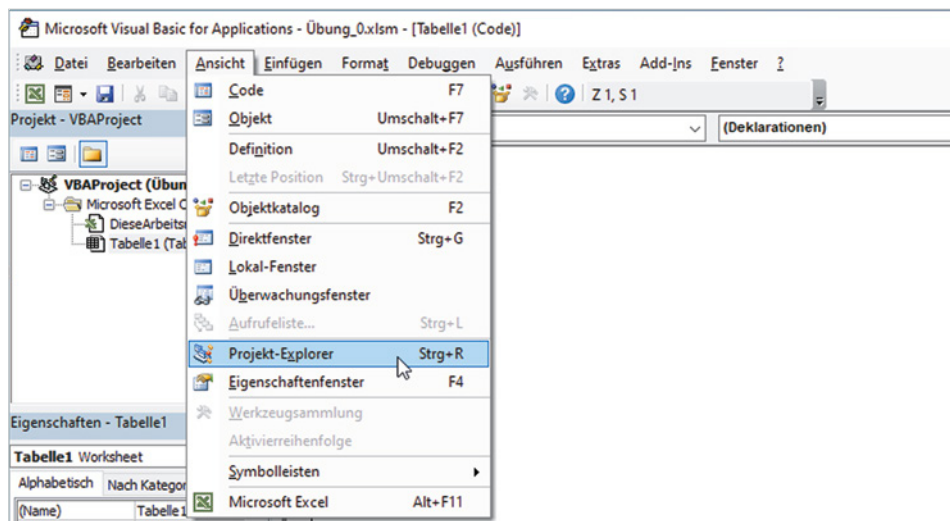
*Die Entwicklungsumgebung mit Projektfenster, Eigenschaftsfenster und Codefenster*



- ▶ Das **Projektfenster** ① zeigt alle zurzeit geöffneten Arbeitsmappen als Projekte an.
- ▶ Im **Eigenschaftfenster** ② werden die (veränderlichen) Eigenschaften bzw. Besonderheiten der verwendeten Objekte dargestellt.
- ▶ Der Programmcode wird im **Codefenster** ③ 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 *Eigenschaft* 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.

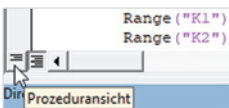
*Im Menü Ansicht lassen sich die Fenster ein- und ausblenden*



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

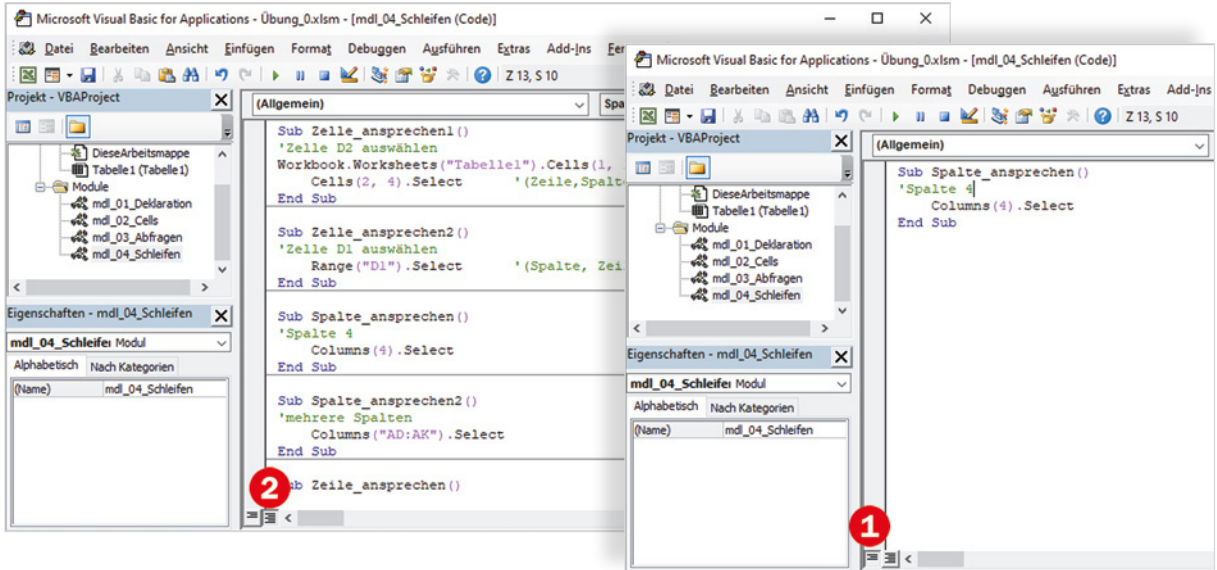
## Das Codefenster

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

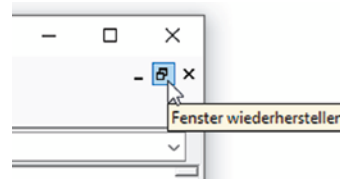




Sie in den Prozedurcode und anschließend auf das linke Symbol (*Prozeduransicht*) ❶ am unteren Rand des Codefensters. Ein Klick auf das Symbol rechts daneben ❷ zeigt wieder alle Prozeduren an.

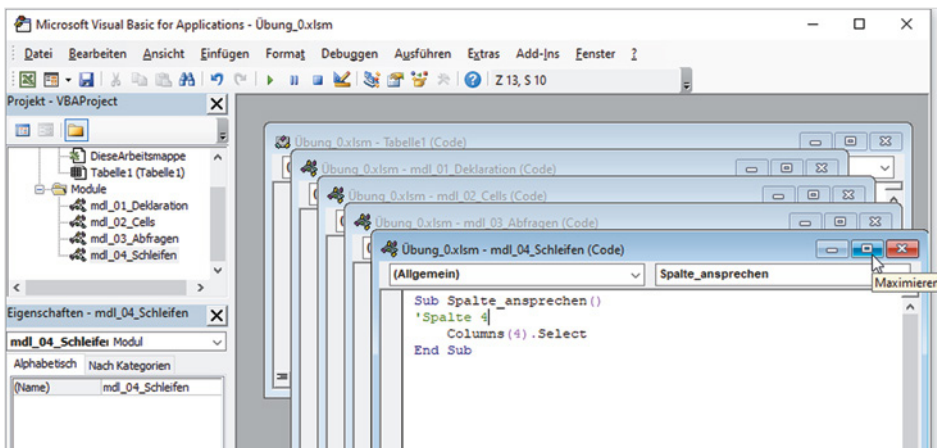


Prinzipiell lassen sich die Module auch in Fenstern nebeneinander oder überlappend anzeigen. Dazu klicken Sie auf das Symbol *Fenster wiederherstellen*. In der Praxis dürfte allerdings diese Darstellungsform kaum von Bedeutung sein. Um die Standardansicht wiederherzustellen, brauchen Sie nur eines der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



*Alle Prozeduren (Standard)*

*Einzelansicht eines Makros*



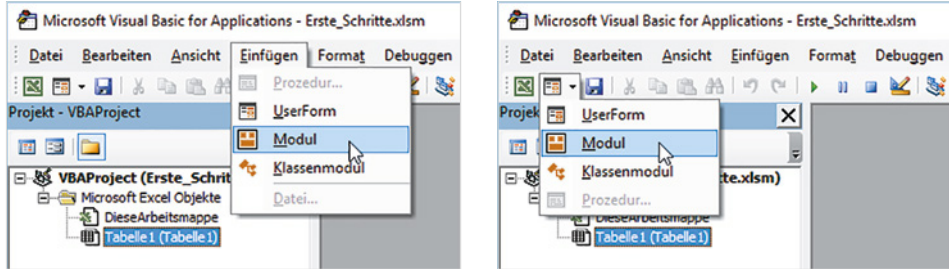
*Überlappende Modulfenster im Codefenster*



## 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*.

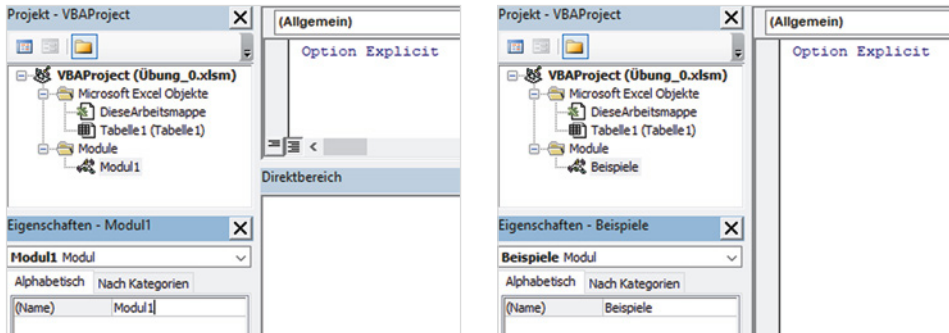
### Modul einfügen



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

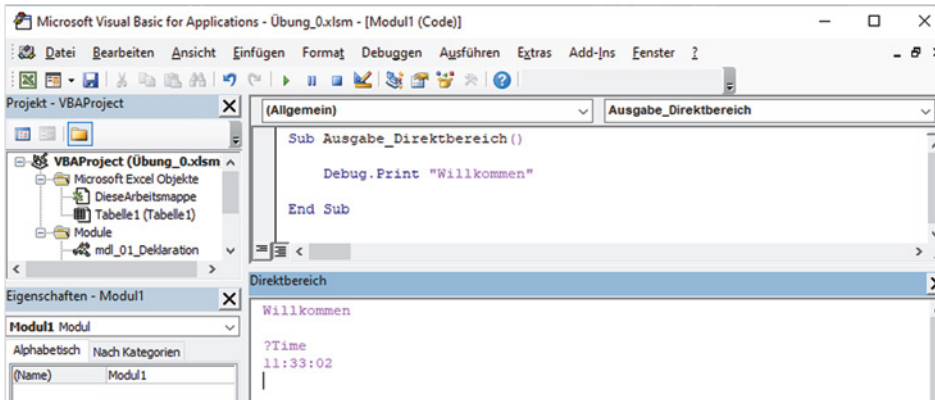
Das Fenster Direktbereich wird mit der Tastenkombination Strg+G oder den Menübefehl *Ansicht* ► *Direktfenster* eingeblendet und erscheint am unteren Rand der VBA-Entwicklungsumgebung unterhalb Codefensters.

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.

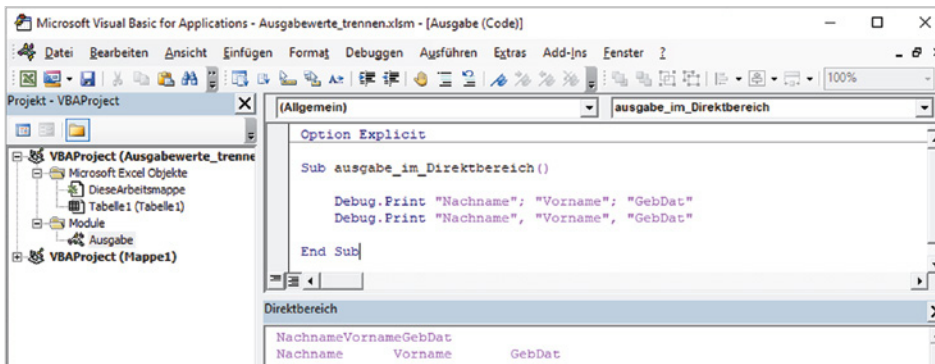
```
Debug.Print "Willkommen"
? Time
```



Ausgabe im Direktbereich

## Ausgabewerte trennen

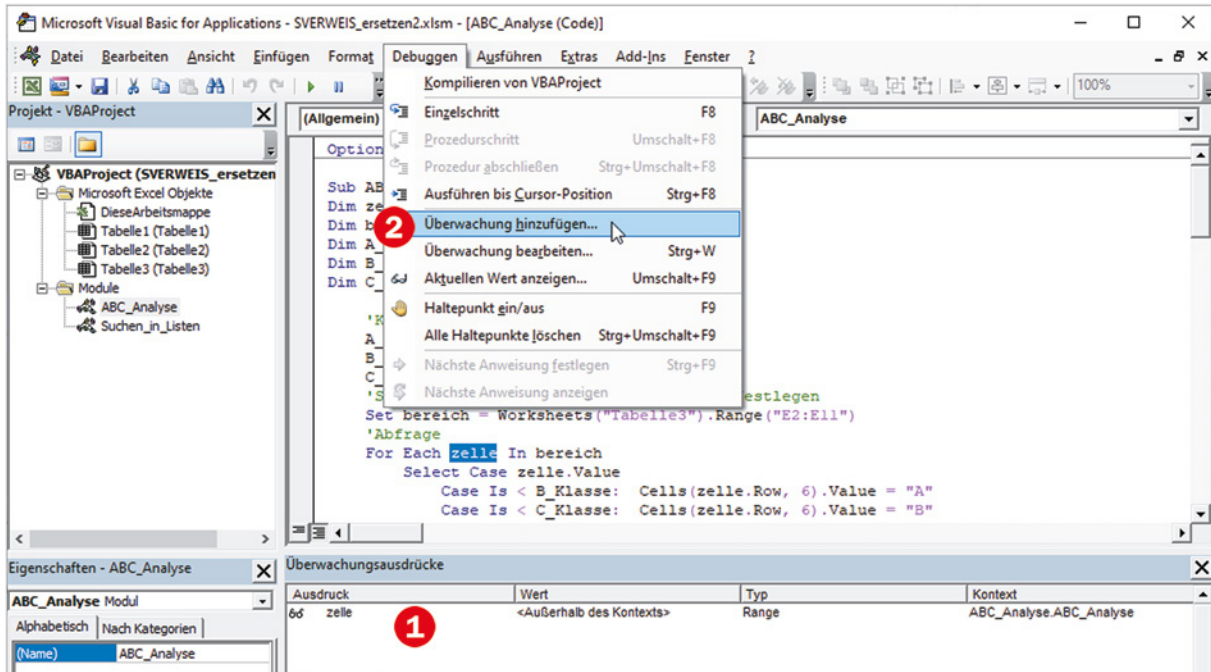
Werden in einer Debug.Print-Zeile mehrere Ausgabewerte durch Semikolon (;) getrennt, erhalten Sie im Direktbereich eine Ausgabe mit Leerzeichen getrennt. Die Trennung durch Komma in der Codezeile bewirkt die Ausgabe mit Tab-Abständen.



Getrennte Ausgabewerte

## Das Überwachungsfenster

Wenn Sie während des Ablaufs einer Prozedur Variablen überprüfen wollen, um beispielsweise bei bestimmten Bedingungen den Programmablauf zu unterbrechen, steht Ihnen das Überwachungsfenster **1** 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* **2**).



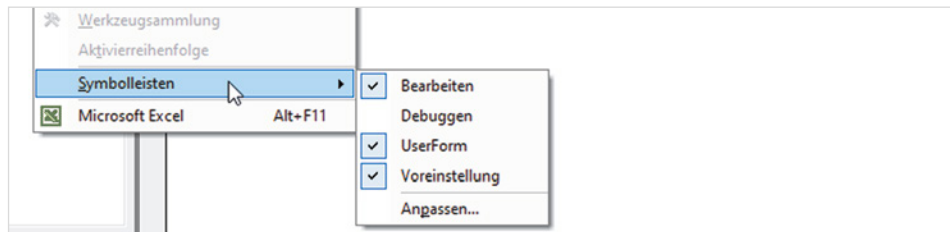
Überwachung einer Variablen hinzufügen

## 1.5 Einstellungen im VBA-Editor

### Symboleisten einblenden

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

Hilfreiche Symboleisten



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

Symboleiste Bearbeiten

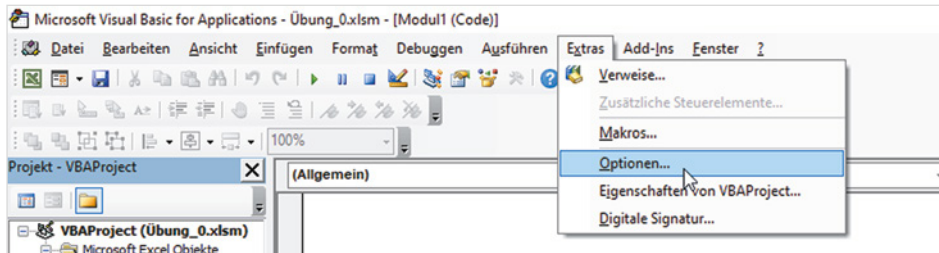
► Beim Erstellen von Formularen (UserForms) kann die Symboleiste *UserForm* beim Ausrichten und Aufteilen von Steuerelementen unterstützen.

Symboleiste UserForm



## Editier-Optionen

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



*Extras Optionen*

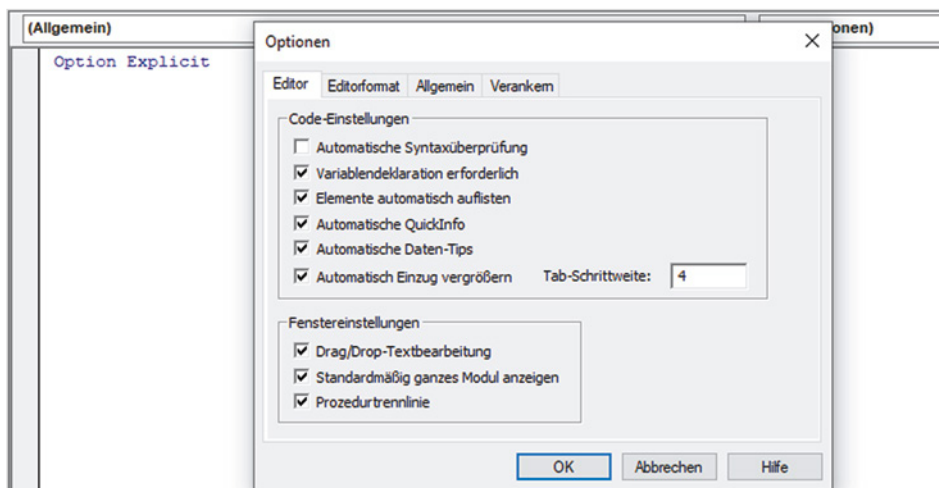
## Variablendeklaration erzwingen

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*.

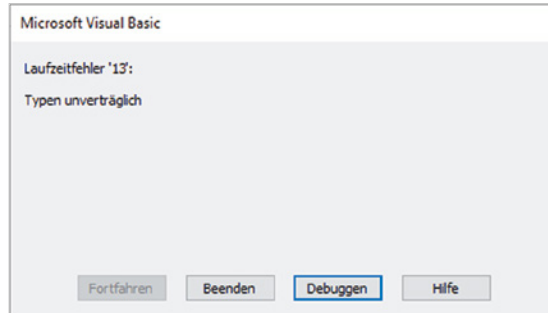


*Variablendeklaration erforderlich als Standardeinstellung*

### 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.

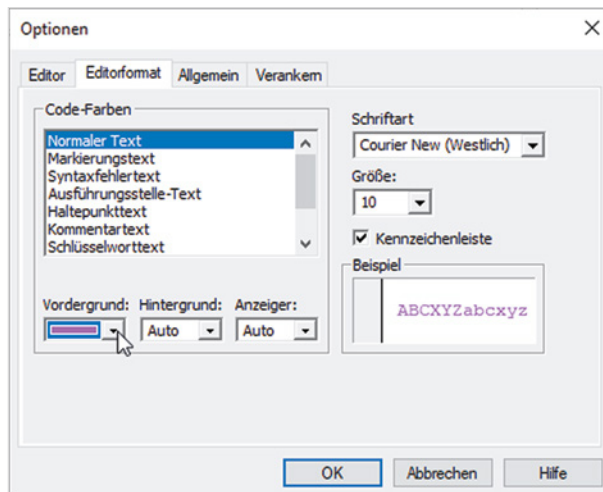
### Schrifteinstellungen

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 30**  
Prozedur erstellen; Gültigkeitsbereiche von Prozeduren
- 2.2 **Der Makrorecorder als Programmierhilfe 31**  
Aufzeichnen mit dem Makrorecorder; Beispiel: Zeilen und Spalten vertauschen
- 2.3 **Makros, Prozeduren und Formulare ausführen 37**  
Im Dialogfenster Makros starten; Tastenkombination zuweisen; Der Schnellzugriffsleiste und/oder dem Menüband hinzufügen; Beim Öffnen der Arbeitsmappe ausführen; Prozedur aus anderen Prozeduren heraus aufrufen; Über Befehlsschaltflächen starten; Makro per Rechtsklick bzw. Kontextmenü ausführen; Makros im Menüband, Register Add-Ins integrieren
- 2.4 **Programmausführung testen, Fehlerbehandlung 50**  
Einzelschritte testen; Laufzeitfehler abfangen
- 2.5 **Eigene Funktionen erstellen 53**  
Aufbau von Funktionen; Einfache Funktionen ohne Parameter; Parameterübergabe an Funktionen
- 2.6 **Weitergeben von Makros und Funktionen 57**  
Über die Zwischenablage; Exportieren / Importieren; In der persönlichen Arbeitsmappe ablegen; Arbeitsmappe weitergeben und VBA-Projekt schützen; Als Add-In speichern/laden

## 2.1 Module und Prozeduren erzeugen

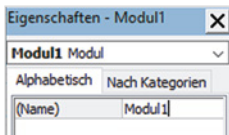
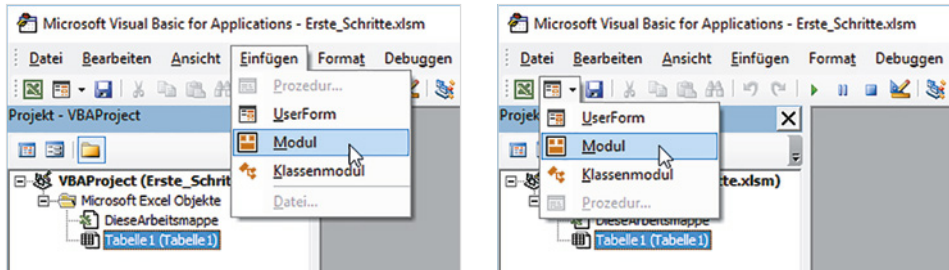
Prozeduren, auch als Makros bezeichnet, sind kleine, eigenständige Programmeinheiten mit einem eindeutigen Namen, unter dem sie später aufgerufen werden. Eine Prozedur beginnt immer mit der Anweisung *Sub*, gefolgt vom eigentlichen Namen und endet mit *End Sub*.

### 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.

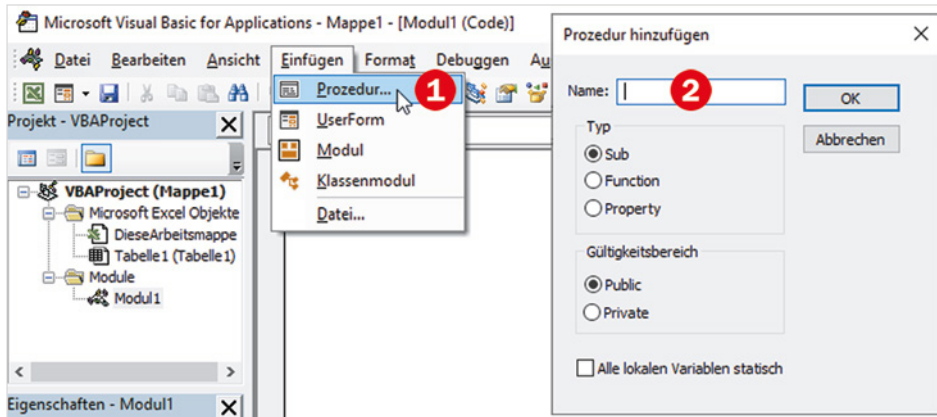
Prozedurrumpf



**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...* **1** und geben anschließend einen Prozedurnamen ein **2**. 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.

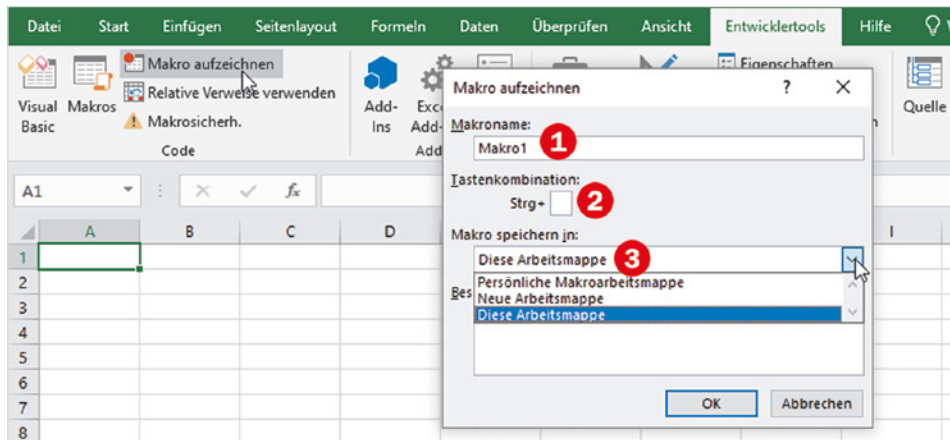


**Nachteil:** Der Makrorecorder generiert neben der benötigten Anweisung häufig auch eine Vielzahl überflüssiger Codezeilen, die zwecks besserer Übersichtlichkeit manuell gelöscht werden sollten.

## Aufzeichnen mit dem Makrorecorder

Klicken Sie im Tabellenblatt, Register *Entwicklertools* auf *Makro aufzeichnen*. Geben Sie einen Namen ein **1**, unter dem das Makro später aufgerufen werden soll, optional auch eine Tastenkombination **2** und legen Sie den Speicherort **3** des Makros fest. Nachdem Sie auf *OK* geklickt haben, läuft die Aufzeichnung.

*Makro aufzeichnen*

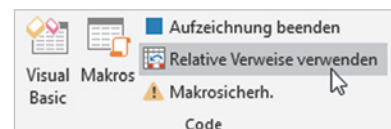


**Achtung Speicherort:** Standardvorgabe im Feld *Makro speichern in* ist *Diese Arbeitsmappe*. Dies ist in 99 % aller Fälle auch sinnvoll, da die meisten Makros auf spezielle Aufgaben der jeweiligen Arbeitsmappe zugeschnitten sind. Wenn Sie dagegen *Persönliche Makroarbeitsmappe* wählen, dann ist das 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

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 *Makrorecorder* aufgezeichnet werden. Liegen mehrere Adressen in dieser Form vor, lassen sich die Einzelschritte später mehrfach wiederholen. Voraussetzung ist eine Aufzeichnung mit Relativen Verweisen.

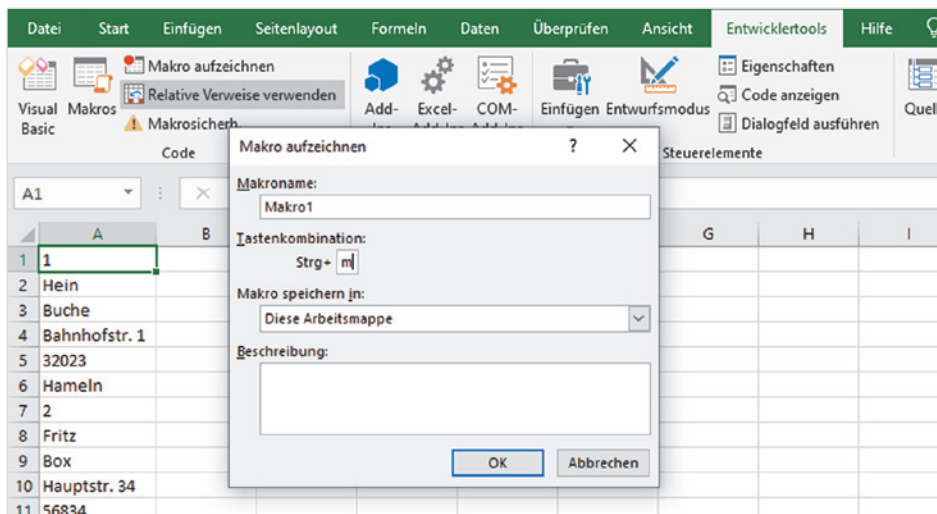
*Die Adressliste*

	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				

### 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 Makrol ()
'
' Makrol 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	Hamel		
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.