

---

Ricardo Hernández García

1. Ausgabe, März 2016

ISBN 978-3-86249-551-1

## Excel 2016

**Automatisierung,  
Programmierung**

EX2016P



**HERDT**

## 5 Makros bearbeiten und verwalten

### In diesem Kapitel erfahren Sie

- ✓ wie Makros aufgebaut sind
- ✓ wie Sie Makros im Visual Basic-Editor bearbeiten können

### Voraussetzungen

- ✓ Komponenten des Visual Basic-Editors



Beispieldatei: *Kap05.xlsm*

### 5.1 Aufbau von Makros

#### Aufbau des Makro-Codes

Der Makro-Code besteht aus Visual Basic-Anweisungen, die beim Ausführen des Makros nacheinander abgearbeitet werden. Ein aufgezeichnetes Makro hat den folgenden Aufbau:

```


① Sub GelberHintergrund()
  '
② ' GelberHintergrund Makro
③ ' Formatiert Hintergrund der Zelle gelb.
  '
④ ' Tastenkombination: Strg+Umschalt+G
  '
⑤     With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 65535
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
⑥ End Sub

```

- ① Jedes Makro wird mit dem Schlüsselwort `Sub` (für engl. Subroutine = Unterprogramm) eingeleitet. Anschließend folgt der Name des Makros und ein Klammernpaar `()`.
- ② Der Name des Makros, gefolgt von dem Begriff `Makro`, wird als Kommentar eingefügt. Ein Kommentar beginnt mit einem Hochkomma. Kommentare dienen nur der Beschreibung des Programmcodes und haben keinen Einfluss auf die Ausführung des Makros.
- ③ Wenn Sie vor der Aufzeichnung des Makros eine Beschreibung eingegeben haben, wird die Beschreibung als Kommentar eingefügt.
- ④ Haben Sie für das Makro eine Tastenkombination angegeben, wird die Tastenkombination in dieser Zeile als Kommentar eingefügt.
- ⑤ Dem Kommentarbereich folgen die Anweisungen des Makros.
- ⑥ Das Ende des Makros ist durch die Schlüsselwörter `End Sub` gekennzeichnet.

## Formatierung des VBA-Codes im Code-Fenster

Für die Anzeige im Code-Fenster wird der Code so aufbereitet, dass er für den Anwender bzw. den Programmierer gut lesbar ist.

- ✓ Die Anzeige verschiedener Teile des Codes erfolgt in unterschiedlichen Farben, z. B. werden standardmäßig Schlüsselwörter, z. B. `Sub` bzw. `End Sub`, blau und Kommentare grün dargestellt.
- ✓ Code-Zeilen untergeordneter Blöcke werden eingerückt. Beispielsweise bilden alle Anweisungen, die zwischen `Sub` und `End Sub`  stehen, einen Block und werden standardmäßig um 4 Zeichen eingerückt. Innerhalb eines Blockes kann es auch wieder Blöcke geben, die wiederum eingerückt werden können.

Den Einzug und die Farben können Sie im Visual Basic-Editor über das Dialogfenster *Optionen (Extras - Optionen)* Register *Editor* bzw. *Editorformat* individuell verändern.

Um die Lesbarkeit zu verbessern, werden die im Buch abgedruckten Beispiel-Codes auf ähnliche Weise formatiert. Schlüsselwörter werden hier **fett** hervorgehoben. Die Einrückung von Code-Zeilen erfolgt wie im Code-Fenster, allerdings teilweise nur um 2 Zeichen.



## 5.2 Makros bearbeiten

### Aufgezeichnetes Makro zur Bearbeitung öffnen

Sie haben nach der Aufzeichnung eines Makros in Excel die Möglichkeit, die Anweisungen des Makros im Visual Basic-Editor zu bearbeiten.

- ✓ Sie können Kommentare zur Erläuterung einfügen.
- ✓ Um den Code lesbarer zu machen oder unerwünschte Nebenwirkungen zu vermeiden, können Sie überflüssige Anweisungen löschen.
- ✓ Sie können Fehler in der Aufzeichnung des Makros korrigieren. Haben Sie beispielsweise bei der Positionierung des Cursors zu viele Schritte ausgeführt, können Sie diese im VBA-Code entfernen.
- ▶ Falls Sie den VBA-Editor noch nicht geöffnet haben, klicken Sie im Register *Entwicklertools* in der Gruppe *Code* auf das Symbol *Makros*.  
*oder* Rufen Sie im VBA-Editor den Menüpunkt *Extras - Makros* auf.
- ▶ Markieren Sie im Dialogfenster *Makro* das gewünschte Makro und betätigen Sie *Bearbeiten*.  
Im Code-Fenster des Visual Basic-Editors blinkt der Cursor im VBA-Code des gewählten Makros.





### Effektives Arbeiten mit dem VBA-Code

- ✓ Schließen Sie die Fenster des Visual Basic-Editors, die Sie momentan nicht zur Bearbeitung benötigen, und zeigen Sie das Code-Fenster in der Vollbilddarstellung an.
- ✓ Ordnen Sie das Anwendungsfenster von Excel und das Anwendungsfenster des Visual Basic-Editors so an, dass Sie die betreffenden Bereiche von Excel und den entsprechenden Code-Ausschnitt im Visual Basic-Editor sehen können. Damit können Sie beispielsweise beim Testen eines Makros das Ergebnis sofort in Excel sehen bzw. beim Aufzeichnen eines Makros verfolgen, wie die aufgezeichneten Befehle im Makro in VBA-Code umgesetzt werden.



### VBA-Code editieren

Im Code-Fenster des Visual Basic-Editors können Sie den VBA-Code editieren.

Bei der Bearbeitung gelten die in Windows üblichen Konventionen der Textbearbeitung. Sie können unter anderem mit Drag & Drop, den üblichen Tastenkombinationen, z. B.   und  , und den bekannten Markierungsmöglichkeiten arbeiten.



Am VBA-Code vorgenommene Änderungen werden sofort und ohne Bestätigung wirksam. Starten Sie das Makro nach der Änderung in Excel, wird das geänderte Makro ausgeführt (auch ohne Speicherung).

### Beispiel zur Bearbeitung von VBA-Code: *Kap05.xlsm*

Im Makro `Schrift14FettRot` sollen überflüssige bzw. unerwünschte Anweisungen gelöscht und Kommentare ergänzt werden. Bei der Aufzeichnung des Makros wurde zuerst der Schriftgrad 14 festgelegt (Register `Start`, Gruppe `Schriftart`, Feld `Schriftgrad`), dann der Schriftschnitt `Fett` bestimmt (Register `Start`, Gruppe `Schriftart`, Symbol `F`) und zuletzt die Schriftfarbe `Rot` festgelegt (Register `Start`, Gruppe `Schriftart`, Symbol `A`). Beim Erstellen des VBA-Codes wurden nicht nur die Einstellungen für die gewünschten Änderungen gespeichert, sondern noch weitere mit den Befehlen verbundene Einstellungen. Die überflüssigen Anweisungen sollen entfernt werden.

```

① Sub Schrift14FettRot ()
② '
  ' Schrift14FettRot Makro
  ' Formatiert die markierten Zellen mit dem Schriftgrad 14,
  ' mit dem Schriftschnitt fett und in der Schriftfarbe Rot.
  '
  '
  '
③   With Selection.Font
④       .Name = "Arial" ' abhängig von der eingestellten Standard-Schriftart
⑤       .Size = 14
④       .Strikethrough = False
        .Superscript = False
        .Subscript = False
        .OutlineFont = False
        .Shadow = False
        .Underline = xlUnderlineStyleNone
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .ThemeFont = xlThemeFontNone
⑥   End With
⑦   Selection.Font.Bold = True
      With Selection.Font
⑧       .Color = -16776961
④       .TintAndShade = 0
      End With
⑨ End Sub

```

- ①, ⑨ Das Makro mit dem Namen `Schrift14FettRot` wird durch das Schlüsselwort `Sub` eingeleitet und mit den Schlüsselwörtern `End Sub` beendet.
- ② Zeilen, die mit einem Hochkomma `'` beginnen, sind Kommentarzeilen und dienen nur der Erläuterung des VBA-Codes. Diese Kommentarzeilen wurden beim Aufzeichnen automatisch erzeugt.
- ③, ⑥ Mit der `with`-Anweisung wird eine vereinfachte Schreibweise ermöglicht. Hier beziehen sich die in der `with`-Anweisung eingeschlossenen Anweisungen alle auf das Objekt `Selection.Font`. In der `with`-Anweisung kann deshalb z. B. `.Name` anstelle von `Selection.Font.Name` verwendet werden.
- ③ - ⑧ Mit diesen Anweisungen werden die Eigenschaften der Schrift festgelegt.
- ④ Diese Anweisungen wurden bei der automatischen Aufzeichnung erzeugt und sind überflüssig.

- ⑤ Für den Text wird die Schriftgröße 14 festgelegt.
- ⑦ Durch diese Anweisung wird der Schriftschnitt auf Fett eingestellt.
- ⑧ Der markierte Text wird in der Farbe Rot formatiert.

### VBA-Code optimieren

- ▶ Versehen Sie die notwendigen Anweisungen ⑤, ⑦ und ⑧ mit Kommentaren.
- ▶ Löschen Sie nun die Anweisungen ④, die bei der Makroaufzeichnung zuviel generiert wurden, aus dem VBA-Code des Makros.

*oder* Kommentieren Sie die Zeilen aus, indem Sie ein `'`-Zeichen vor die Zeile setzen.

- ▶ Testen Sie das Makro erneut.  
Das Makro formatiert die markierten Zellen wie vor der Änderung, allerdings werden beispielsweise bereits bestehende Schriftarten beibehalten.

```
' Schriftgrad 14
  Selection.Font.Size = 14
' Text fett
  Selection.Font.Bold = True
' Schriftfarbe rot
  Selection.Font.Color = -16776961
```

VBA-Code bearbeitet (Kap05.xlsm, Makro Schrift14FettRotMinimal)

Kommentieren Sie Zeilen, die Sie entfernen möchten, zunächst mit einem Kommentarzeichen `'` am Anfang der Zeile aus und testen Sie das Makro. Arbeitet es wie gewünscht, können Sie die auskommentierten Zeilen entfernen.



## 5.3 Makros im VBA-Editor erstellen, speichern und drucken

### Vorteile beim direkten Programmieren von Makros im VBA-Editor

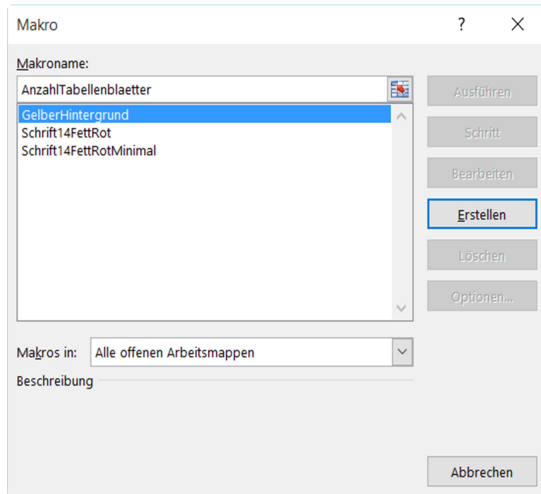
Der Code des Makros wird in der Programmiersprache Visual Basic for Applications (VBA) generiert. Die direkte Programmierung mit VBA bringt gegenüber dem Aufzeichnen von Makros folgende Vorteile:

- ✓ Der Sprachumfang von VBA ist wesentlich größer und bietet somit viel mehr Möglichkeiten, z. B. die Verwendung eigener Dialogfenster und das wiederholte Ausführen von Programmen bzw. Programmteilen.
- ✓ Mit VBA können Sie nicht nur auf Excel, sondern auch programmübergreifend auf alle Office-Anwendungen zugreifen.

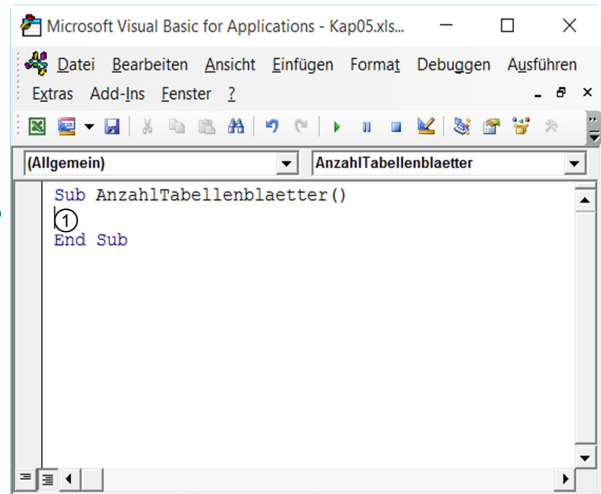
### Leeres Makro erstellen

Wenn Sie im Code-Fenster des Visual Basic-Editors ein Makro manuell erstellen möchten, erzeugen Sie zuerst ein leeres Makro.

- ▶ Klicken Sie im Register *Entwicklertools* in der Gruppe *Code* auf *Makros*.  
Alternative: `Alt` `F8`
- ▶ Geben Sie im Feld *Makroname* den Namen des Makros ein, z. B. *AnzahlTabellenblaetter*.
- ▶ Wählen Sie im Feld *Makros in* den Speicherort des Makros aus.
- ▶ Betätigen Sie *Erstellen*.  
Der Visual Basic-Editor wird geöffnet und das erzeugte leere Makro wird im Code-Fenster angezeigt. Der Cursor blinkt in der leeren Zeile, in die nun die erste Anweisung des Makros eingegeben werden kann.



Neues Makro erstellen




Neu erstelltes Makro im Code-Fenster

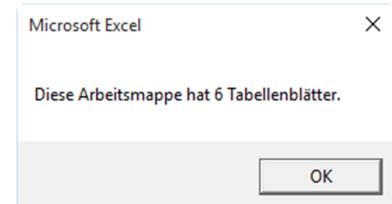
### Beispiel zum Erstellen eines neuen Makros im Visual Basic-Editor: *Kap05.xlsm*

- ▶ Tragen Sie im Code-Fenster an der Cursorposition ① die folgende Anweisung ② ein.

```
' Mit dem Befehl MsgBox können Sie einen Text in einem Dialogfenster
' ausgeben. Standardmäßig zeigt das Dialogfenster die Schaltfläche OK an.
' Worksheets.Count liefert die Anzahl der Tabellenblätter
```

```
② MsgBox "Diese Arbeitsmappe hat " & Worksheets.Count & " Tabellenblätter."
```

- ▶ Testen Sie das Makro, indem Sie eine Zelle markieren und es anschließend starten (Symbolleiste *Voreinstellung*, ).



Die Ausgabe des Makros „AnzahlTabellenblaetter“

### Makros speichern

Nachdem Sie mehrere Anweisungen bearbeitet haben oder bevor Sie ein Makro testen, sollten Sie die Arbeitsmappe, die das Makro enthält, speichern. Auch bevor Sie den Visual Basic-Editor verlassen, ist es ratsam, die Änderungen zu speichern, da diese sonst erst beim Speichern der Arbeitsmappe mitgesichert werden.

- ▶ Rufen Sie im Visual Basic-Editor den Menüpunkt *Datei - Arbeitsmappenname speichern* auf.

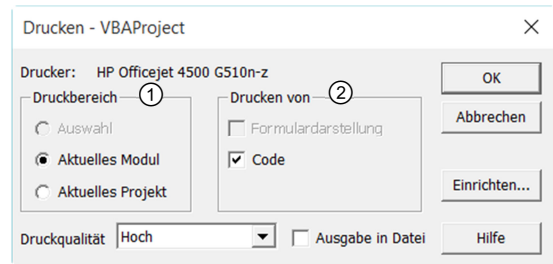
Alternativen:  bzw. **Strg** **S**

## VBA-Code oder Formular drucken

- ▶ Rufen Sie im Visual Basic-Editor den Menüpunkt *Datei - Drucken* auf.

Alternative: **Strg** **P**


- ▶ Wählen Sie den gewünschten Druckbereich ①.
  - ✓ Ist die Option *Aktuelles Modul* ausgewählt, werden der Code und/oder das Formular des aktuellen Code-Fensters gedruckt.
  - ✓ Aktivieren Sie die Option *Aktuelles Projekt*, werden der gesamte Code und/oder alle Formulare des Projekts gedruckt.



Im Bereich *Drucken von* ② können Sie wählen, ob nur Programmcode oder nur die Dialogfenster oder beides gedruckt werden soll.

- ▶ Bestätigen Sie mit *OK*, um den Ausdruck zu starten.

## Zu Excel zurückkehren

- ▶ Um zu Excel zurückzukehren, ohne den Visual Basic-Editor zu schließen, klicken Sie auf das Symbol  oder auf das Anwendungssymbol in der Taskleiste von Windows.

*oder* Um zu Excel zurückzukehren und den Visual Basic-Editor zu schließen, rufen Sie den Menüpunkt *Datei - Schließen und zurück zu Microsoft Excel* auf.

Alternative:  bzw. **Alt** **Q**



**Ergänzende Lerninhalte:** *Module und Makros kopieren und konvertieren.pdf*

Hier erfahren Sie, wie Sie die Makros und Module kopieren und konvertieren können.

## 5.4 Schnellübersicht

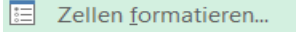
Sie möchten ...	
Makros bearbeiten	Register <i>Entwicklertools</i> , Gruppe <i>Code</i> , Symbol <i>Makros</i> , Makro markieren, Schaltfläche <i>Bearbeiten</i>
Makros erstellen	Register <i>Entwicklertools</i> , Gruppe <i>Code</i> , Symbol <i>Makros</i> , Makroname eingeben, Speicherort wählen, Schaltfläche <i>Erstellen</i>
im Visual Basic-Editor ein Makro speichern und Excel aktivieren	<i>Datei - Arbeitsmappenname speichern</i> , <i>Datei - Schließen und zurück zu Microsoft Excel</i>

## 5.5 Übung

### Makros bearbeiten

Übungsdatei: --

Ergebnisdatei: **Formatierung-E.xlsm**

1. Erstellen Sie eine neue Arbeitsmappe *Formatierung-E.xlsm* und zeichnen Sie zwei Makros auf.
2. Das erste Makro `SchriftFettDialog` soll den Text von selektierten Zellen fett anzeigen. Verwenden Sie bei der Aufzeichnung des Makros das Dialogfenster .
3. Erstellen Sie ein zweites Makro `SchriftFett`, in dem die Formatierung über das entsprechende Symbol (**F**) eingestellt wird.
4. Testen Sie die Makros jeweils an Beispielen, die z. B. verschiedene Schriftgrößen besitzen.
5. Öffnen Sie den Visual Basic-Editor und vergleichen Sie beide Makros miteinander.
6. Löschen Sie alle überflüssigen Anweisungen aus dem ersten Makro und testen Sie es.



---

# Impressum

Matchcode: EX2016P

Autor: Ricardo Hernández Garcia

Redaktion: Andrea Weikert

Produziert im HERDT-Digitaldruck

1. Ausgabe, März 2016

HERDT-Verlag für Bildungsmedien GmbH

Am Kümmerling 21-25

55294 Bodenheim

Internet: [www.herd.com](http://www.herd.com)

E-Mail: [info@herd.com](mailto:info@herd.com)

© HERDT-Verlag für Bildungsmedien GmbH, Bodenheim

Alle Rechte vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder einem anderen Verfahren) ohne schriftliche Genehmigung des Verlags reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Dieses Buch wurde mit grosser Sorgfalt erstellt und geprüft. Trotzdem können Fehler nicht vollkommen ausgeschlossen werden. Verlag, Herausgeber und Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Wenn nicht explizit an anderer Stelle des Werkes aufgeführt, liegen die Copyrights an allen Screenshots beim HERDT-Verlag. Sollte es trotz intensiver Recherche nicht gelungen sein, alle weiteren Rechteinhaber der verwendeten Quellen und Abbildungen zu finden, bitten wir um kurze Nachricht an die Redaktion.

Die in diesem Buch und in den abgebildeten bzw. zum Download angebotenen Dateien genannten Personen und Organisationen, Adress- und Telekommunikationsangaben, Bankverbindungen etc. sind frei erfunden. Eventuelle Übereinstimmungen oder Ähnlichkeiten sind unbeabsichtigt und rein zufällig.

Die Bildungsmedien des HERDT-Verlags enthalten Verweise auf Webseiten Dritter. Diese Webseiten unterliegen der Haftung der jeweiligen Betreiber, wir haben keinerlei Einfluss auf die Gestaltung und die Inhalte dieser Webseiten. Bei der Bucherstellung haben wir die fremden Inhalte daraufhin überprüft, ob etwaige Rechtsverstösse bestehen. Zu diesem Zeitpunkt waren keine Rechtsverstösse ersichtlich. Wir werden bei Kenntnis von Rechtsverstössen jedoch umgehend die entsprechenden Internetadressen aus dem Buch entfernen.

Die in den Bildungsmedien des HERDT-Verlags vorhandenen Internetadressen, Screenshots, Bezeichnungen bzw. Beschreibungen und Funktionen waren zum Zeitpunkt der Erstellung der jeweiligen Produkte aktuell und gültig. Sollten Sie die Webseiten nicht mehr unter den angegebenen Adressen finden, sind diese eventuell inzwischen komplett aus dem Internet genommen worden oder unter einer neuen Adresse zu finden. Sollten im vorliegenden Produkt vorhandene Screenshots, Bezeichnungen bzw. Beschreibungen und Funktionen nicht mehr der beschriebenen Software entsprechen, hat der Hersteller der jeweiligen Software nach Drucklegung Änderungen vorgenommen oder vorhandene Funktionen geändert oder entfernt.