

WORKSHOP



VBA-CODE-DATENBANK

MAKROS AUF KNOPFDRUCK

Stellen Sie Ihre Bürohelfer für Microsoft Office einfach per Mausklick aus knapp 2200 Prozeduren und Makrovorlagen zusammen. Mit Hilfe der VBA-Code-Datenbank Personal-VBA-Repository finden Sie so schnell Lösungen für Ihre Excel-Probleme.

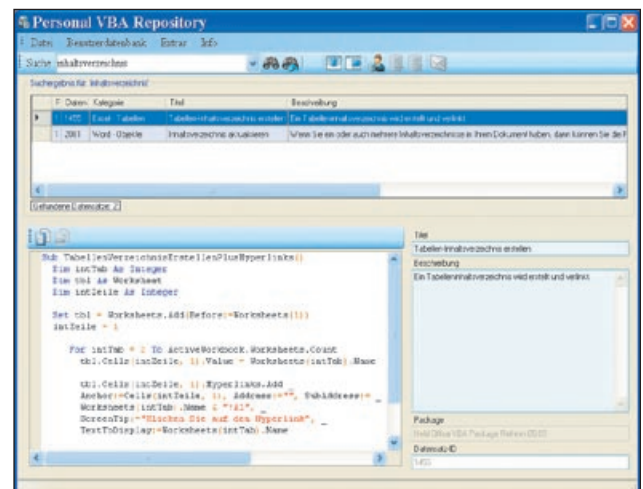
VON **STEFAN LEIBING** UND **BERND HELD**

Wenn Sie oft Excel mit Makros erweitern, um Ihren Arbeitsablauf zu optimieren, wissen Sie, dass Makros schreiben nicht immer einfach ist. Vielleicht haben Sie auch schon einmal stundenlang im Internet recherchiert und an einem Problem mal etwas länger gegessen als Ihnen lieb war. Wie wäre es, wenn Sie auf eine fertige Datenbank mit Makros und Prozeduren zugreifen könnten, die Sie 1:1 in Ihre eigenen Projekte einsetzen könnten?

Das Software-Haus Held-Office stellt dazu das *Personal-VBA-Repository* zur Verfügung (siehe Kasten). In der VBA-Code-Datenbank finden Sie derzeit zirka 2200 Prozeduren zu Excel, Access, Word, PowerPoint, Outlook und übergreifenden VBA-Themen. Über eine ausgefeilte Suchfunktion und eine professionelle Oberfläche finden Sie in Sekundenschnelle die

gewünschte VBA-Lösung. Es besteht zusätzlich die Möglichkeit, eigene Prozeduren in dieser Software abzulegen. Diese benutzerdefinierten Prozeduren werden dann automatisch mit den anderen Prozeduren gespeichert und in die Suchfunktion integriert.

Sämtlicher Programmcode wird der Syntax entsprechend eingefärbt und ist daher im Codefenster des VBA-Repository so komfortabel zu lesen, wie in Ihrer VBA-Entwicklungs-umgebung.



Mit der VBA-Code-Datenbank Personal-VBA-Repository wird die Programmierung und die Entwicklung von Microsoft-Office-Makros zu einem Kinderspiel.

Datenbank verwenden

Zum besseren Verständnis des Tools sollen nun in diesem Artikel einige beispielhafte Lösungen aus dem Bereich Excel mit konkretem Bezug auf die ID-Nummer des Code-Listings vorgestellt werden. Darüber hinaus finden Sie auf der Heft-DVD eine Demoversion des *Personal-VBA-Repository*, die Sie 30 Tage kostenfrei nutzen können. Die nachfolgenden Beispiele entstammen dieser Demoversion. Sie können die Code-Beispiele somit umgehend testen und in Ihre eigenen Office-Dokumente integrieren.

Inhaltsverzeichnis (ID 1455)

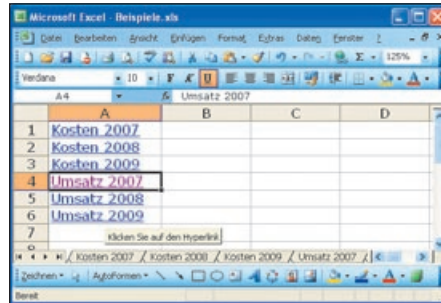
Im ersten Beispiel aus dieser Demoversion wird eine praktische Lösung vorgestellt, mit Hilfe derer Sie in einem Zug in einer Arbeitsmappe zu Beginn eine neue Tabelle einfügen, danach alle in der Mappe befindlichen Tabellennamen dort einfügen und diese über die Hyperlinkfunktion verlinken. So behalten Sie den Überblick über die in Ihren Mappen enthaltenen Tabellen und können diese blitzschnell mit einem Klick aktivieren.

```
Sub TabellenverzeichnisErstellenPlus _
Hyperlinks()
    Dim intTab As Integer
    Dim tbl As Worksheet
    Dim intZeile As Integer
    Set tbl = Worksheets.Add(Before: =
Worksheets(1))
    intZeile = 1
    For intTab = 2 To ActiveWork
book.Worksheets.Count
        tbl.Cells(intZeile, 1).Value = _
Worksheets(intTab).Name
        tbl.Cells(intZeile, 1).Hyperlinks.Add _
Anchor:=tbl.Cells(intZeile, 1), _
Address:="", SubAddress:= _
"" & Worksheets(intTab).Name & "!A1",
ScreenTip:="Klicken Sie auf den
Hyperlink", _
TextToDisplay:=Worksheets(intTab).Name
        intZeile = intZeile + 1
    Next intTab
End Sub
```

Unikatsliste automatisch erzeugen (ID 355)

In dem Beispiel wird aus einer Liste mit doppelten Daten eine so genannte Unikatsliste erzeugt. Diese Unikatslisten werden gewöhnlich als Vorlage für Gültigkeitslisten oder für Dropdown-Felder in Dialogen verwendet.

```
Sub DatenSpezialfilter()
    Dim Bereich As Range
    Set Bereich = Tabelle1.UsedRange.
Columns(1)
    Bereich.AdvancedFilter
Action:=xlFilterCopy, _
CriteriaRange:=Bereich,
CopyToRange:=Tabelle1.Range("E1"),
Unique:=True
End Sub
```



Alle Tabellen in einer Übersicht aufführen und verlinken.

Tabellen mit Passwort versehen (ID 372)

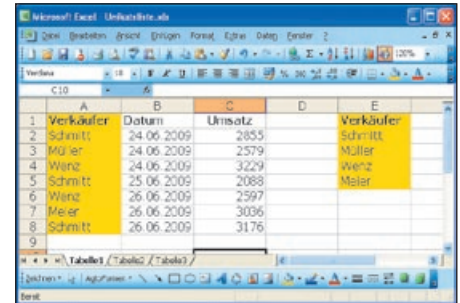
In diesem Beispiel werden alle Tabellen einer Arbeitsmappe mit einem Passwort versehen und geschützt. Änderungen bezüglich der Formatierung einer Tabelle sind jedoch weiterhin möglich.

```
Sub AlleTabellenSchützen()
    Dim Blatt As Worksheet
    For Each Blatt In ThisWorkbook. _
Worksheets
        Blatt.Protect Password:="Test",
AllowFormattingCells:=True
        Next Blatt
    End Sub
```

Exemplarisch wird beim Schutz der Tabellen das Passwort *Test* verwendet.

Analog dazu setzen Sie die Prozedur aus dem folgenden Listing mit der ID 373 ein, um den gerade eingestellten Blattschutz für alle Tabellen der Arbeitsmappe wieder zu entfernen.

```
Sub AlleTabellenEntschützen()
    Dim Blatt As Worksheet
    For Each Blatt In ThisWorkbook.
```



Aus einer Spalte mit Duplikaten wird eine Spalte mit Unikaten erzeugt.

```
Worksheets
    Blatt.Unprotect
Password:="Test"
    Next Blatt
End Sub
```

Die Extra-Fußzeile mit Kick (ID 420)

Für Normal-Anwender relativ schwer zu realisieren – über VBA kein Problem! Erstellen Sie auf Knopfdruck eine benutzerdefinierte mehrzeilige Fußzeile mit dem Erstellungsdatum, dem letzten Änderungsdatum sowie dem Ersteller und dem Pfad- und Dateinamen der Arbeitsmappe.

```
Sub FußzeileSpezialAktiveTabelle()
    With ActiveSheet.PageSetup
        .BottomMargin = 56
        .FooterMargin = 42
        .LeftFooter = "&8" & _
Application.WorksheetFunction.
Rept("-", 35) & vbCr & _
"Erstellungsdatum: " & ActiveWorkbook
.BuiltinDocumentProperties _
("Creation date") & vbCr & _
"Letzte Änderung: " & _
ActiveWorkbook.BuiltinDocument _
Properties _
("last save time") & vbCr & _
```

Personal-VBA-Repository

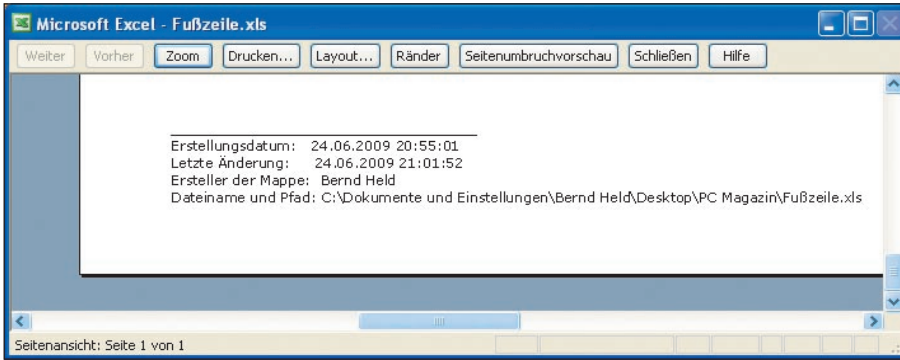
➤ In der VBA-Code-Datenbank *Personal-VBA-Repository* finden Sie derzeit zirka 2200 Prozeduren zu Excel, Access, Word, PowerPoint, Outlook und übergreifenden VBA-Themen. Alle zwei Monate kommen weitere Lösungen dazu. Haben Sie bei



der Integration einer Prozedur eine Frage, dann können Sie diese per Mausclick direkt an Held-Office weiterleiten. Via e-Mail bekommen Sie in der Regel noch am selben Tag eine weiterführende Antwort. Käufer des VBA-Repository können jederzeit den E-Mail-Support von Held-Office nutzen. Das *Personal-VBA-Repository* von Held-Office kostet einmalig 106,51 Euro

inkl. MwSt. und beinhaltet sechs Updates pro Jahr. Das Update wird Ihnen als Download zur Verfügung gestellt und kann sehr einfach und schnell installiert werden.

Für Studenten und Auszubildende wird eine vollwertige Version mit einem 50%igen Rabatt angeboten. Nach Ablauf des ersten Jahres kann ein Folge-Abo für 20 Euro pro weiteres Folgejahr abgeschlossen werden. Das *Personal-VBA-Repository* spart Zeit und Kosten. Die Prozeduren daraus können frei eingesetzt werden. Weitere Informationen finden Sie auf der Homepage des Anbieters unter www.held-office.de.



Benutzerdefinierte Angaben für die Fußzeile über eine Prozedur erzeugen.

```

    "Ersteller der Mappe: " & _
    ActiveWorkbook.BuiltinDocument _
Properties _
    ("author") & vbCr & "Dateiname
und Pfad: " & _
    ActiveWorkbook.FullName
    End With
End Sub

```

```

    Target.Interior.ColorIndex = 46
Case 11 To 15 'GRÜN
    Target.Interior.ColorIndex = 4
Case 16 To 30 'BLAU
    Target.Interior.ColorIndex = 5
Case 31 To 40 'GELB
    Target.Interior.ColorIndex = 6
Case Else
    Target.Interior.ColorIndex = _
xlColorIndexNone
End Select
End If
End Sub

```

HINWEIS

Beachten Sie, dass Sie insgesamt nur 255 Zeichen für die Gestaltung der Kopf- und Fußzeile zur Verfügung haben.

Bedingte Formatierung (ID 1332)

Bis zur Excel-Version 2003 können über die bedingte Formatierung lediglich drei Bedingungen hinterlegt werden. Dies ist jedoch nicht so, wenn Sie eine Ereignisprozedur dafür einsetzen, die Sie genau „hinter“ die gewünschte Tabelle legen (rechter Mausklick auf Tabelleregisterkarte, dann Befehl *Code anzeigen* aus dem Kontextmenü wählen und den Programmcode aus dem Code-Beispiel mit der ID 1332 kopieren).

```

Private Sub Worksheet_Change(ByVal _
Target As Range)
    If Target.Column = 1 Then
        Select Case Target.Value
            Case 1 To 5 'ROT
                Target.Interior.ColorIndex = 3
            Case 6 To 10 'ORANGE

```

Bei der Eingabe von Werten in Spalte A werden die Zellen entsprechend der hinterlegten Regel gefärbt.

Bereich kopieren, Werte einfügen (ID2869)

In diesem Beispiel wird ein Bereich aus einer Tabelle in eine andere Tabelle kopiert. Dabei werden als Ergebnis nur Festwerte übertragen. Im Code-Beispiel mit der ID 2869 wird Ihnen exemplarisch die Umsetzung gezeigt.

```

Sub BereichNurWerteKopieren()
    Tabelle1.Range("A1:D14").Copy
    Tabelle2.Range("A1").PasteSpecial
xlPasteValues
    Application.CutCopyMode = False
End Sub

```

Der Bereich A1:D14 der Tabelle1 wird kopiert und in Tabelle2 beginnend bei Zelle A1 eingefügt. Dabei werden nur die Werte, also

keine Formeln und Formate übertragen. Soll ein Bereich hingegen 1:1 in eine andere Tabelle überführt werden, dann reicht dafür ein einziger Befehl:

```

Sub BereichKopieren()
    Tabelle1.Range("A1:D14").Copy
    Destination:=Tabelle2.Range("A1")
End Sub

```

Leere Zeilen löschen (ID 297)

In diesem Beispiel werden alle leeren Zeilen einer Tabelle gelöscht. Dabei wird geprüft, ob diese Zeilen wirklich leer sind.

```

Sub LeereZeilenLöschen()
    Dim Zeile As Long
    Dim ZeileMax As Long
    With Tabelle1
        ZeileMax = .UsedRange.Rows.Count
        For Zeile = ZeileMax To 2 Step -1
            If Application.Worksheet _
Function.CountA(.Rows(Zeile)) = 0 Then
                .Rows(Zeile).Delete
            End If
        Next Zeile
    End With
End Sub

```

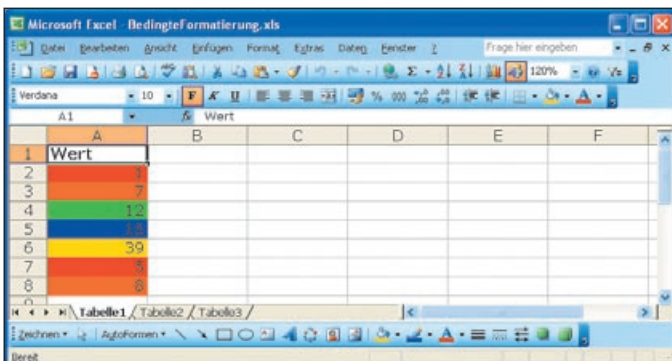
Die Integration einer Tabellenfunktion (ANZAHL2 = CountA) im VBA-Code bringt Ihnen erheblichen Zeitvorteil und Performance.

Tabellen automatisch sortieren (ID 440)

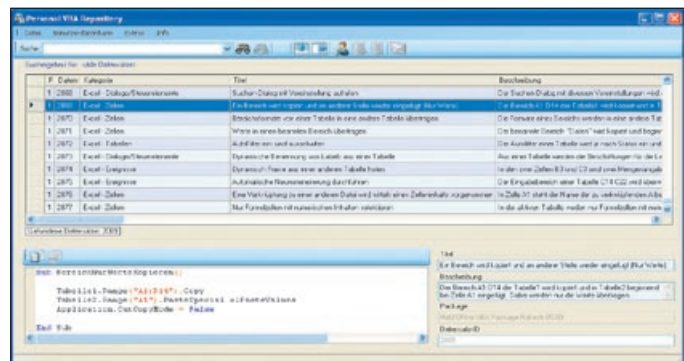
In diesem letzten Beispiel sehen Sie eine Lösung, mit der Sie die Tabellen in Ihrer Arbeitsmappe alphabetisch anordnen können. Damit Sie die Arbeit mit dem Personal-VBA-Repository üben, haben wir diesmal den Code nicht abgedruckt. Starten Sie die Software unter Start/Programm und holen Sie sich den Code aus der Datenbank. Viel Spaß beim Stöbern.

tr

magnus.de powered by PC Magazin
 Weitere Informationen finden Sie unter <http://software.magnus.de/software>



Schneller, speicheroptimiert und ausbaufähig – die bedingte Formatierung via Ereignis-Makro.



Prozeduren direkt aus der Datenbank herauskopieren und in eigene Projekte einfügen.