Excel Master

MAKRO CHEAT SHEET

v.3.01

Übersicht Datentypen:
Die meist verwendeten Datentypen für Variablen sind hier grün markiert.

Typ:	Wertebereich:	Beispiel:
Boolean	Ja/Nein-Werte (True oder False)	Erwachsen = True / False
Byte	Ganzzahlen 0 bis 255	Anzahl = 57
Integer	Ganzzahlen -32.768 bis 32.767	Jahr = 2021
Long	Ganzzahlen -2.147.483.648 bis 2.147.483.647	Zahl = 100975
LongLong	Ganzzahlen ca. ± 9 Trillionen	Menschen = 8000000000
Currency	skalierte Ganzzahlen ca. ± 9 Billionen, 4 Nachkommastellen	Euro = -22.21
Single	Gleitkommazahl, einfache Genauigkeit	gerundet = 3.5
Double	Gleitkommazahl, doppelte Genauigkeit	Ergebnis = 7.75
Date	Datum (1. Jan 100 bis 31. Dez 9999) und Zeit	Datum = 01.01.2021 12:45
String	Texte	Überschrift = "Hallo Welt"
Object	abhängig vom Objekt	Set ARD = Sender
Variant	abhängig vom gerade aktuellen Inhalt	diverses = -1

Infos und Hinweise:
Auf den nächsten Seiten wird fortlaufend ein Makro dargestellt, das sämtliche der wichtigsten VBA-Codes abdeckt. Einige Code-Zeilen sind eingerückt, das dient der Übersichtlichkeit.

Den auf den folgenden Seiten abgebildeten VBA-Code kannst du 1zu1 aus diesem Dokument kopieren, in ein Modul eingefügen und die VBA-Codes anhand deines eigenen Beispiels testen.

Die einzelnen Programmierzeilen werden hier meist vor oder rechts vom eigentlichen Programmier-Code mit einem Kommentar erläutert. Die Position der Erläuterung hängt davon ab wie viel Platz rechts vom VBA-Code vorhanden ist und stellt den Code sowohl übersichtlich für einen selbst als auch für Arbeitskollegen dar. Navigiere am besten mit Strg + F durch dieses Dokument falls du spezielle VBA-Codes suchst. Ich hoffe ich kann dir hiermit einiges an Zeit ersparen.

ACHTUNG: Änderungen, die durch Makros ausgeführt wurden können nicht rückgängig gemacht werden, daher achtet immer darauf eure Dokumente regelmäßig zwischen zu speichern.

Farblegende:
Grün wird ein Kommentar markiert
Blau sind Schlüsselworttexte, die in VBA vorreserviert sind (hier nicht weiter relevant)
Schwarz werden alle restlichen Bestandteile des VBA-Codes dargestellt
Mit '---Text---' wird jeweils ein neues Thema begonnen

```
'Eine Kommentarzeile beginnt mit einem Hochkomma

'Mit "Sub" beginnen wir ein neues Makro und mit End Sub endet dieses.
'Der MakroName sollte zusammenhängend sein und darf nicht mit einer Zahl beginnen
'Tipp: Es können beliebig viele Makros in einem Modul untereinander geschrieben werden.
Sub MakroName()

'Schaltet die Aktualisierung des Bildschirms während des Makrodurchlaufs aus
Application.ScreenUpdating = False

'Schaltet Pop-Up Fenster von Excel während des Makrodurchlaufs aus
'Pop-Up Fester können Makros unterbrechen
Application.DisplayAlerts = False
```

'---Deklaration der Variablen in VBA---'

```
'Variablen werden immer ohne Leerzeichen geschrieben
Dim varTextVariable As String 'String = Text-Variable
Dim varZahlenVariableCounter As Integer 'Integer = Ganzzahlen
Dim varZahlenVariableMitNachkommastellen As Double
Dim varJaNeinVariable As Boolean 'Boolean = Ja/Nein-Variable (True oder False)
```

'---Andere Dateien öffnen/speichern/schließen---'

```
'Öffnet eine weitere Datei
Workbooks.Open "D:\Dateipfad\Dateiname.xlsx"
'Speichert eine die aktive Exceldatei
ActiveWorkbook.Save
'Speichert die aktive Exceldatei unter einem bestimmten Namen
ActiveWorkbook.SaveAs FileName:="E:\User\JohnDoe\Dateiname.xlsx"
'Speichert eine Datei unter einem bestimmten Namen und verwendet hierbei Werte aus Variablen
ActiveWorkbook.SaveAs FileName:=varPfad & varDateinamen & ".xlsx"

'Schließt die aktive Exceldatei ohne sie zu speichern
ActiveWorkbook.Close savechanges:=False

'Speichert alle geöffneten Workbooks/Exceldateien
For Each w In Application.Workbooks 'Schleife geht jedes Workbook durch
    w.Save 'speichert jedes Workbook
Next w 'wählt das nächste Workbook aus
Application.Quit 'Schließt das Excel Programm komplett
```

'---Tabellenblätter/Tabs---'

```
'Tabellenblätter auswählen
Sheets("Tabelle1").Select 'Tabellenblatt anhand des Tab-Namens auswählen
Sheets(2).Select 'Tabellenblatt anhand der Position auswählen
'Tabellenblatt anhand des im VBA Editor hinterlegten Namens auswählen
Tabelle2.Select 'Vorteil hiervon: Funktioniert auch nach Änderung des Tab-Namens im Frontend

'Tabs hinzufügen
Sheets.Add After:=ActiveSheet 'Fügt neuen Tab hinter dem selektierten ein
Sheets.Add Before:=Sheets(1) 'Fügt neuen Tab an erster Stelle ein
Sheets("Tabelle1").Name = "Neuer Name" 'Benennt den Tab "Tabelle1" um in "Neuer Name"
ActiveWindow.SelectedSheets.Delete 'Löscht die aktuell selektierten Tabellenblätter
```

'---Selektieren---'

```
Range("A1").Select 'Wählt Zelle A1 aus
Range("A1:C3").Select 'Wählt den Bereich A1:C3 aus
Rows("8:8").Select 'Wählt die komplette Zeile 8 aus
Rows("3:5").Select 'Wählt die kompletten Zeilen 3-5 aus
Columns("C:C").Select 'Wählt die komplette Spalte C aus
Columns("A:C").Select 'Wählt die kompletten Spalten A-C aus
Cells.Select 'Wählt alle Zellen im Dokument aus
ActiveSheet.UsedRange.Select 'Wählt den Bereich aus, der bisher im Tabellenblatt benutzt wurde
'Ausgehend von A1 springt die Selektion zur letzten befüllten Zelle vor einer Leerzelle nach unten
Range("A1").End(xlDown).Select
```

'---Von der aktiven Zelle ausgehend - relative Bezüge---'

```
'Versetzt die Auswahl eine Zeile nach unten
ActiveCell.Offset(1, 0).Select
'Versetzt die Auswahl zwei Spalten nach rechts
ActiveCell.Offset(0, 2).Select
'Wählt ausgehende von den aktuell aktiven Zelle die komplette Zeile/n aus
ActiveCell.EntireRow.Select
'Wählt ausgehende von den aktuell aktiven Spalten die komplette Zeile/n aus
ActiveCell.EntireColumn.Select

'Zum Ende eines befüllten Bereichs springen
'Folgende 4 Zeilen springen nach links, rechts, oben oder unten die zur letzten befüllten Zelle
Selection.End(xlToLeft).Select
Selection.End(xlToRight).Select
Selection.End(xlUp).Select
Selection.End(xlDown).Select
```

'---Kopieren und einfügen---'

```
'Kopiert Zellen oder Bereiche
Range("C5").Copy
Range("C5:E20").Copy
'Kopiert die aktuelle Auswahl
Selection.Copy

'Einfügen: nur Werte
Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
    :=False, Transpose:=False
'Einfügen: nur Formate
Selection.PasteSpecial Paste:=xlPasteFormats, Operation:=xlNone, SkipBlanks _
    :=False, Transpose:=False
'Kopiermodus aufheben
Application.CutCopyMode = False

'Fügt den selektierten Bereich als leere Zellen neu ein
Selection.Insert Shift:=xlDown 'xlDown verschiebt die bestehenden Zellen nach unten
Selection.Insert Shift:=xlToRight 'xlToRight verschiebt die bestehenden Zellen nach rechts
```

'---Werte und Formeln eintragen---'

```
'Werte in Zellen eintragen
Range("A1").Value = "Dieser Text steht nun in A1" 'Fügt Text in eine Zelle ein
Range("B2").Value = 5 'Fügt eine Zahl in eine Zelle ein
Range("A1:B3").Value = 5 'Fügt eine Zahl in einen Bereich ein
```

'---Formeln in Zellen einfügen und erweitern---'

```
'Fügt eine Summen-Formel in eine Zelle ein
ActiveCell.FormulaR1C1 = "=SUM(R[-2]C:R[-1]C)"
'Formel erweitern auf andere Zellen (funktioniert nach unten, rechts, links und oben)
Selection.AutoFill Destination:=Range("E11:E18"), Type:=xlFillDefault
```

'---Formatierungen---'

```
'Mit With kann hier einiges an Programmiertext gespart werden, da sonst vor
'jeder Zeile Selection.Font eingesetzt werden müsste Selection.Font.Bold... , Selection.Font.Italic...
With Selection.Font
    .Bold = True
    .Italic = True
    .Underline = xlUnderlineStyleSingle
End With

Selection.Interior.Color = RGB(255, 0, 0) 'Farbe der Zelle ändern
```

'---Arbeiten mit Variablen---'

```
'Liest den Wert aus Zelle E2 aus und übergibt ihn an eine Variable
varTextVariable = Range("E2").Value
'Liest den exakt angezeigten Text aus Zelle E7 aus und übergibt ihn an eine Variable
varTextVariable = Range("E7").Text
```

'---Löschen---'

```
'Löscht den Inhalt einer Zelle oder eines Bereichs
Selection.ClearContents
Range("A1:C3").ClearContents
ActiveCell.ClearContents
'Löscht die Formatierung einer Zelle oder eines Berechts
Selection.ClearFormats
Range("A1:C3").ClearFormats
ActiveCell.ClearFormats

'Löscht alle Formate und Inhalte einer Zelle/eines Bereichs
Range("C3").Clear

'Löscht die gesamt Spalte/n ausgehend von der/den aktiven Zelle/n
ActiveCell.EntireColumn.Delete Shift:=xlLeft
'Löscht die gesamt Zeile/n ausgehend von der/den aktiven Zelle/n
ActiveCell.EntireRow.Delete Shift:=xlUp

'Löscht einen bestimmten Bereich
Range("C3").Delete Shift:=xlToLeft 'xlToLeft verschiebt die bestehenden Zellen nach links
Range("B5:B20"). Shift:=xlToLeft
Selection.Delete Shift:=xlUp 'xlUp verschiebt die bestehenden Zellen nach oben
```

'---Auslesen aus Zellen---'

```
'Übergibt den eingetragenen Wert einer Zelle (z.B. "01.01.2021")
varTextVariable = Range("C10").Value
MsgBox varTextVariable 'Als Check eingebaut um zu sehen was wirklich in der Variablen gelandet ist
'Übergibt exakt den angezeigten und formatierten Text-Wert der Zelle (z.B. "1. Jan 2021")
varTextVariable = Range("C10").Text
MsgBox varTextVariable 'Als Check eingebaut um zu sehen was wirklich in der Variablen gelandet ist
```

'---If Else Bedingungen---'

```
If Range("A1").Value = 1 Then
    'Wenn die Bedingung zutrifft. Durch das Einrücken der Zeile entsteht mehr Übersicht.
    MsgBox "Bedingung ist wahr"
Else
    'Wenn die Bedingung nicht zutrifft.
    Exit Sub 'Beendet das aktuelle Makro
End If 'Beendet einen If-Block
```

'---Message Boxen---'

```
'Text in einer Pop-Up MessageBox
MsgBox "Dies ist eine MessageBox mit Text"

'Berechnetes Ergebnis in der MessageBox
MsgBox varZahlenVariableCounter + varZahlenVariableMitNachkommastellen

'Text + Variable in der MessageBox
MsgBox "Das Ergebnis der Berechnung lautet: " & Variable3

'Ein Absatz in einer MessageBox wird mit zwei aufeinanderfolgenden & vbCrLf & vbCrLf
'Anweisungen angezeigt
'Ein Absatz in der VBA Programmierung zur besseren Übersicht wird mit einem Unterstrich
'ausgewiesen
MsgBox ("Hier kommt ein Absatz in der MessageBox" & _
    vbCrLf & "" & _
    vbCrLf & "Text nach dem Absatz")

'Mit einer InputBox können beim Durchlauf eines Makros Inputs eingegeben werden
'Diese Inputs können dann in Variablen gespeichert und später weiterverwendet werden
varTextVariable = InputBox("Nachricht", "Titel", Default) 'Default gibt Nachrichtentyp an

'Eine Yes/No Box stellt eine Frage und führt z.B. das Makro nur weiter aus wenn die Antwort Ja ist
varTextVariable = MsgBox("Text", vbQuestion + vbYesNo + vbDefaultButton2, "Titel")

'Auswertung der YesNo Box mit einer If-Abfrage
'Überprüft ob die Antwort "Ja" war
If answer = vbYes Then
  'Wenn die Antwort "Ja" war, kann ein beliebiger Code abgespielt werden. Hier eine MsgBox.
  MsgBox "Yes"
'Wenn die Antwort "nein" war springt das Makro direkt auf die Else-Anweisung.
Else
  'Beliebiger Code kann abgespielt werden
  'Exit Sub beendet das Makro
  Exit Sub
'End If beendet eine If-Verzweigung
End If
```

'---Loops---'

```
'Do While führt eine Schleife aus so lange die Bedingung erfüllt ist
Do While varZahlenVariableCounter <= 5 'Führt die Schleife aus bis die Variable größer als 5 ist
  'Setzt die Auswahl Zeile für Zeile eine Zelle weiter nach unten
  ActiveCell.Offset(1, 0).Select
  'Zählt die Variable bei jedem Schleifendurchlauf 1 hoch, sodass sie irgendwann bei 5 ankommt
  varZahlenVariableCounter = varZahlenVariableCounter + 1
'Loop beendet die Schleife
Loop

'Die "Do Until-Schleife" wird so lange ausgeführt BIS eine bestimmte Bedingung eintritt.
'Führt die Schleife so lange aus bis die Variable den Wert 10 wiedergibt
Do Until varZahlenVariableCounter = 10
ActiveCell.Offset(0, 1).Select
'Zählt die Variable bei jedem Schleifendurchlauf 1 hoch
varZahlenVariableCounter = varZahlenVariableCounter + 1
'Loop beendet die Schleife
Loop

'Die For Each Schleife geht in diesem Fall jedes Tabellenbaltt durch
For Each ws In ActiveWorkbook.Worksheets
'Auf jedem Tabellenblatt wird Zelle A1 selektiert
ws.Range("A1").Select
'Next springt zum nächsten Tabellenblatt bis alle Tabellenblätter durchgearbeitet wurden
Next ws

'Formula Counter
'Erstellt eine Variable für die Zellen als Range
Dim cell As Range
'Erstellt eine Variable um alle Formeln zu zählen
Dim CountallFormula As Long
    'Jede Zelle des im aktuellen Tabellenblatt benutzten Bereichs wird durchgegangen
    For Each cell In ActiveSheet.UsedRange
      'Prüft ob die aktuelle Zelle eine Formel beinhaltet
      If cell.HasFormula Then
        'Wenn die aktuell geprüfte Zelle ein Formel beinhaltet wird Variable hochgezählt
        CountallFormula = CountallFormula + 1
      'Eine Else Anweisung wird hier gar nicht benötigt - End If beendet die If-Verzweigung wieder
      End If
    'Next cell springt zur nächsten Zelle und führt den Schleifeninhalt erneut aus
    Next cell
```

'---Sonstiges---'

```
  'Ein weiteres Makro im aktuellen Makro abspielen lassen
  Call Makro2

  'Rot wird in VBA eine fehlerhaft programmierte Zeile dargestellt.
  'Im folgenden Beispiel ist ein Schreibfehler bei "value" und ein Geichzeichen ohne Wert = Fehler
  Range("A1").valu =

  'Aktiviert am Ende des Makros wieder das aktualisieren des Displays
  Application.ScreenUpdating = True
  'Aktiviert am Ende des Makros wieder die Display Pop-Ups
  Application.DisplayAlerts = True

  'Ananweisung für das Sub - beendet das aktuelle Makro
End Sub
```

Feedback, Fehler und Kommentare:
Auch wenn ich dieses Dokument mehrfach Korrektur lesen lassen und alle enthaltenen Makros ebenfalls mehrfach getestet wurden kann es dennoch immer mal wieder vorkommen, dass
Sollten dir also Fehler in diesem Dokument auffallen oder hast du vielleicht Vorschläge zur Verbesserung, dann lasses mich gerne wissen. Ich würde mich sehr freuen wenn dieses Dokument über die Zeit und mit dem Input von dir und anderen aus der Excel VBA Community zu einem umfangreichen Nachschlagwerk für jeden VBA-Nutzer werden würde.

Schreib mir einfach an **info@excelmaster.de**

Viel Erfolg bei der Arbeit mit Makros

Daniel