HARALD NAHRSTEDT

Excel + VBA

Ergänzungen

Kapitel

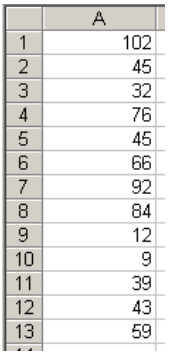
Einführung in VBA

Bedingte Formate mit VBA erstellen

Erstellt am	15.10.2011
Beschreibung	Zellen und Zellbereiche lassen sich durch Formate gestalten. Dazu liefert Excel einige vorgegebene Formate. Aber Formate lassen sich auch mit Bedingungen koppeln, so dass sie immer nur bei erfüllter Bedingung zur Anwendung kommen. Wie sich solche bedingten Formate per VBA erstellen lassen, ist Gegenstand dieses Kapitels.

Zellen und Zellbereiche lassen sich durch Formate gestalten. Dazu liefert Excel einige vorgegebene Formate. Aber Formate lassen sich auch mit Bedingungen koppeln, so dass sie immer nur bei erfüllter Bedingung zur Anwendung kommen. Wie sich solche bedingten Formate per VBA erstellen lassen, ist Gegenstand dieses Kapitels. Es ist so allgemein gehalten, dass es in den Versionen von 2003 bis 2010 angewendet werden kann.

1 Bereiche vorbereiten



	A
1	102
2	45
3	32
4	76
5	45
6	66
7	92
8	84
9	12
10	9
11	39
12	43
13	59

Eine bedingte Formatierung wird immer auf den Bereich angewendet, der zuvor markiert ist. Das ändert sich auch unter VBA nicht. Erstellen wir daher eine einfache Zahlenliste wie nachfolgend dargestellt.

Für den Bereich A1:A13 erstellen wir in einem Modul ein Range-Objekt und anschließend selektieren wir dieses Objekt.

```
Sub SetFormatConditions()
   Dim Bereich1 As Range

   Set Bereich1 = Range("A1:A13")
   Bereich1.Select

End Sub
```

Jeder Bereich besitzt die Unterobjektliste *FormatConditions*. In ihr lassen sich also bedingte Formate instanziieren und auch wieder entfernen. Das Löschen aller vorhandenen bedingten Formate kann über die Methode Delete erfolgen.

```
   Selection.FormatConditions.Delete
```

Die Anweisung sollte immer am Anfang einer Prozedur stehen, damit eventuell noch vorhandene Formate gelöscht werden.

2 Die Methode Add

Immer wenn wir es mit Objektlisten zu tun haben und ein weiteres Objekt instanziieren wollen, kommt die Methode *Add* ins Spiel. Sie hat in diesem Fall die allgemeine Syntax

```
Add(Type As XlFormatConditionType, _
  [Operator], _
  [Formula1], _
  [Formula2]) As FormatCondition
```

Es gibt zwei Formen von FormatConditionType's. Je nachdem, ob man die Bedingung vom Inhalt der Zellen abhängig macht, oder über eine Funktionskonstruktion bestimmt.

```
   xlCellValue  (Bedingung für Zellinhalte)
   xlExpression (Bedingung laut Formel)
```

Um den Zellinhalt mit einem Wert vergleichen zu können, benötigt man einen Operator. Es ist einer der nachfolgend aufgeführten.

```
   xlBetween
   xlEqual
   xlGreater
   xlGreaterEqual
   xlLess
   xlLessEqual
   xlNotBetween
   xlNotEqual
```

Damit ist auch schon das erste sinnvolle Beispiel möglich. Darin werden die Werte im Bereich1 gesucht, die gleich oder größer sind als 50, und mit lichtem Gelb (ColorIndex=36) als Hintergrund gekennzeichnet.

```
Sub Beispiel1()
   Dim Bereich1 As Range

   Set Bereich1 = Range("A1:A13")
   Bereich1.Select
   With Selection
      .FormatConditions.Add _
        Type:=xlCellValue, _
        Operator:=xlGreaterEqual, _
        Formula1:="50"
      .FormatConditions(1).Interior.ColorIndex = 36
   End With
End Sub
```

Da wir hier nur einen Wert (nämlich 50) haben, wird auch nur der Parameter Formula1 benötigt. Anders sieht das im nächsten Beispiel aus. Da werden alle Werte zwischen 50 und 90 markiert.

```
Sub Beispiel2()
   Dim Bereich1 As Range

   Set Bereich1 = Range("A1:A13")
   Bereich1.Select
   With Selection
      .FormatConditions.Add _
        Type:=xlCellValue, _
        Operator:=xlBetween, _
        Formula1:="50", _
        Formula1:="90"
      .FormatConditions(1).Interior.ColorIndex = 36
   End With
End Sub
```

Das Beispiel1 lässt sich auch über den Formeltyp definieren.

```
Sub SetFormatConditions()
   Dim Bereich1 As Range

   Set Bereich1 = Range("A1:A13")
   Bereich1.Select
   With Selection
      .FormatConditions.Add _
         Type:=xlExpression, _
         Formula1:="=A1>=50"
      .FormatConditions(1).Interior.ColorIndex = 36
   End With
End Sub
```

Diesmal nimmt der Parameter Formula1 die als Bedingung definierte Formel auf.

3 Die Liste bedingter Formate

In der Version 2003 ließen sich nur bis zu drei bedingten Formaten für einen Zellbereich festlegen. Ab der Version 2007 können es auch mehr sein. Bereits in den Beispielen haben wir den Listenindex benutzt, um die Hintergrundfarbe einzustellen.

```
      .FormatConditions(1)
```

Mit jedem Add wird also ein weiteres Listenelement instanziiert. Das nachfolgende Beispiel erzeugt drei bedingte Formatierungen für den Bereich. Die erste Bedingung sucht wieder alle Zahlen, die größer oder gleich 50 sind. Die zweite Bedingung sucht alle Zahlen, die kleiner oder gleich 20 sind. Die dritte Bedingung markiert die Zahl 32.

Die erste Bedingung formatiert den Hintergrund wieder in lichtem Gelb. Die zweite Bedingung erstellt einen blauen Rahmen um die Zelle, mit unterschiedlichen Linientypen. Die dritte Bedingung verändert Farbe und Typ der Schrift.

```
Sub SetFormatConditions2()
   Dim Bereich1 As Range

   Set Bereich1 = Range("A1:A13")
   Bereich1.Select
   With Selection
   '1. Bedingung
      .FormatConditions.Add _
         Type:=xlExpression, _
         Formula1:="=A1>=50"
   '2. Bedingung
      .FormatConditions.Add _
         Type:=xlExpression, _
         Formula1:="=A1<20"
   '3. Bedingung
      .FormatConditions.Add _
         Type:=xlExpression, _
         Formula1:="=A1=32"
   'Zellformate setzen
      'Hintergrundfarbe
      .FormatConditions(1).Interior.ColorIndex = 36
   'Zellrahmen zur bedingten Formatierung 2
      With .FormatConditions(2).Borders(xlLeft) 'Linie links
         .LineStyle = xlContinuous
         .Weight = xlHairline
         .ColorIndex = 5
      End With
      With .FormatConditions(2).Borders(xlRight) 'Linie rechts
         .LineStyle = xlContinuous
         .Weight = xlHairline
         .ColorIndex = 5
      End With
      With .FormatConditions(2).Borders(xlTop) 'Linie oben
         .LineStyle = xlContinuous
         .Weight = xlHairline
         .ColorIndex = 5
      End With
      With .FormatConditions(2).Borders(xlBottom) 'Linie unten
         .LineStyle = xlContinuous
         .Weight = xlThin
         .ColorIndex = 5
      End With
   'Schrift
      With .FormatConditions(3).Font
         .Bold = True
         .Italic = True
         .ColorIndex = 3
      End With
   End With
End Sub
```