HARALD NAHRSTEDT

# Excel + VBA

## Ergänzungen

**Kapitel**

Einführung in VBA

### Kommentare mit VBA verwalten

Erstellt am 15.10.2011

Beschreibung Excel stellt für jede Zelle einen Kommentarbereich zur Verfügung. Darin kann Hinweistext gespeichert werden, der jedes Mal bei der Aktivierung der Zelle erscheint oder dauerhaft angezeigt werden kann.

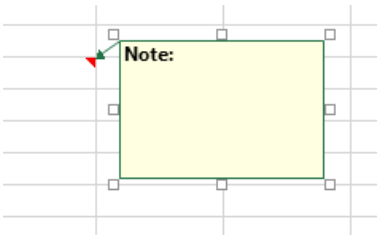
Excel stellt für jede Zelle einen Kommentarbereich zur Verfügung. Darin kann Hinweistext gespeichert werden, der jedes Mal bei der Aktivierung der Zelle erscheint oder dauerhaft angezeigt werden kann.

## 1 Kommentarvorgabe

Mit dem Erstellen eines Kommentars erscheint im Kommentarfeld der Benutzername. Um diesen zu ändern, oder um einen anderen Text einzustellen, genügt die nachfolgende kleine Prozedur.

```
Sub CommentNote()
   Application.UserName = "Note"
End Sub
```

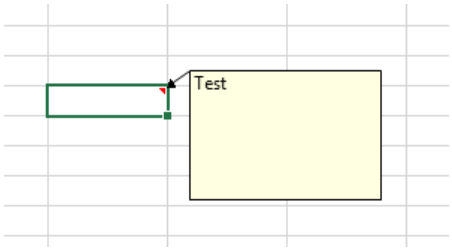
Das Ergebnis sieht dann wie folgt aus. Natürlich kann jeder beliebige Text gewählt werden.




## 2 Einen Kommentar erstellen

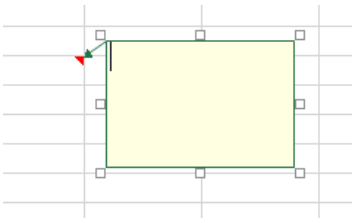
Ein Kommentar ist ein Objekt der Klasse Comments und kann mit dem Set-Befehl instanziiert werden. Der Kommentar wird der markierten Zelle zugeordnet.

```
Sub CommentAddOrNot()
   Dim comTest As Comment

   Set comTest = ActiveCell.Comment
   If comTest Is Nothing Then
      ActiveCell.AddComment Text:="Test"
   End If
End Sub
```



Etwas komfortabler ist die nachfolgende Prozedur, bei der man sofort den neuen Kommentar schreiben kann.

```
Sub CommentAddOrExit()
   Dim comTest As Comment

   Set comTest = ActiveCell.Comment
   If comTest Is Nothing Then
      Set comTest = ActiveCell.AddComment
   End If
   comTest.Visible = True
   comTest.Text Text:=""
   comTest.Shape.Select
End Sub
```



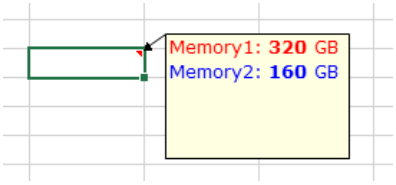
Die nachfolgende Prozedur formatiert zusätzliche den Kommentar in der Schriftart Verdana. Stellt die Schriftgröße auf 12, fett und die Schriftfarbe auf Rot.

```
Sub CommentAddFormat()
   Dim comTest As Comment

   Set comTest = ActiveCell.Comment
   If comTest Is Nothing Then
      Set comTest = ActiveCell.AddComment
      With comTest.Shape.TextFrame.Characters.Font
         .Name = "Verdana"
         .Size = 12
         .Bold = True
         .ColorIndex = 3
      End With
   End If
   comTest.Visible = True
   comTest.Text Text:=""
   comTest.Shape.Select
End Sub
```

Es können aber auch einzelne Textteile unterschiedlich formatiert werden. Die nachfolgende Prozedur erzeugt eine erste Zeile in Rot und eine zweite Zeile in Blau. Die Werte werden fett dargestellt.

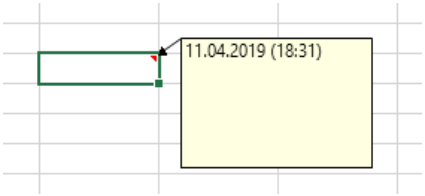
```
Sub CommentFormatColor()
   Dim comTest As Comment
   Dim sTx1    As String
   Dim sTx2    As String
   Dim lStop   As Long
   Dim lNo1    As Long
   Dim lNo2    As Long
   Dim lNoLg   As Long
   Dim sFind   As String

   'Settings
   On Error Resume Next
   sTx1 = "Memory1: 320 GB"
   sTx2 = "Memory2: 160 GB"
   sFind = ":"
   lNoLg = 4

   'erzeuge Comment
   Set comTest = ActiveCell.Comment
   If comTest Is Nothing Then
      ActiveCell.AddComment Text:=sTx1 & vbLf & sTx2
      Set comTest = ActiveCell.Comment
      With comTest.Shape.TextFrame.Characters.Font
         .Name = "Verdana"
         .Size = 10
      End With
   End If

   'finde LineFeed und Marken
   lStop = InStr(1, comTest.Text, vbLf)
   lNo1 = InStr(1, comTest.Text, sFind) + 1
   lNo2 = InStr(lStop + 1, comTest.Text, sFind) + 1

   'formatiere Textzeile
   With comTest.Shape.TextFrame
      .Characters(1, lStop).Font.ColorIndex = 3
      .Characters(lStop + 1, Len(comTest.Text)).Font.ColorIndex = 5
   End With

   'Werte fett darstellen
   If lNo1 > 0 Then
      With comTest.Shape.TextFrame
         .Characters.Font.Bold = False
         .Characters(lNo1, lNoLg).Font.Bold = True
         .Characters(lNo2, lNoLg).Font.Bold = True
      End With
   End If
End Sub
```

Das Ergebnis ist nachfolgend dargestellt.




Oft möchte man das Datum und die Uhrzeit mit in den Kommentar einbinden, damit man weiß, wann der Hinweis geschrieben wurde. Die nachfolgende Prozedur erstellt aktuell Datum und Uhrzeit in einen neuen oder bereits vorhandenen Kommentar. Auch hier muss der Zellbereich natürlich vorher markiert sein.

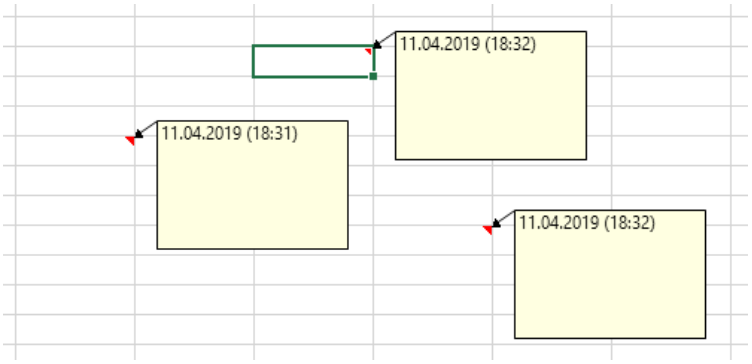
```
Sub CommentAddNow()
   Dim comTest As Comment
   Dim sDate   As String

   sDate = "dd.mm.yyyy (hh:mm)"
   Set comTest = ActiveCell.Comment
   If comTest Is Nothing Then
      Set comTest = ActiveCell.AddComment
      comTest.Text Text:=Format(Now, sDate) & vbLf
   Else
      comTest.Text Text:=Format(Now, sDate) & vbLf & _
         comTest.Text
   End If
End Sub
```




### 3 Kommentare zeigen und kopieren

Die nachfolgende Prozedur zeigt alle Kommentare der aktiven Tabelle an, auch wenn sie ausgeblendet waren.

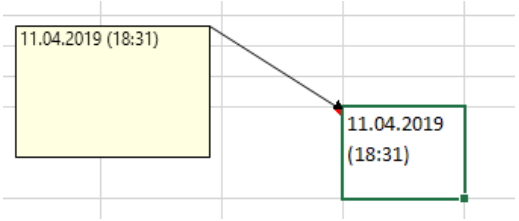
```
Sub CommentsOnSheet()
   Dim comTest As Comment

   For Each comTest In ActiveSheet.Comments
      comTest.Visible = True
   Next
End Sub
```




Es kommt schon einmal vor, dass der Kommentartext in eine Zelle kopiert werden muss. Natürlich geht das auch einfach manuell. Die nächste Prozedur kopiert den Kommentartext in die Zelle rechts daneben. Vorausgesetzt, sie ist leer.

```
Sub CommentsCopyCell()
   Dim rngTest    As Range
   Dim rngCell    As Range
   Dim wshTest    As Worksheet

   Set wshTest = ActiveSheet
   On Error Resume Next
   Set rngTest = wshTest.Cells.SpecialCells(xlCellTypeComments)
   On Error GoTo 0
   If rngTest Is Nothing Then
      MsgBox "Keine Kommentare gefunden!"
      Exit Sub
   End If
   For Each rngCell In rngTest
      If rngCell.Offset(0, 1).Value = "" Then
         rngCell.Offset(0, 1).Value = rngCell.Comment.Text
      End If
   Next rngCell
End Sub
```

Das Ergebnis unseres Anwendungsbeispiels sieht dann so aus.




Eine regelrechte Verwaltung von Daten erlaubt die nachfolgende Prozedur. Sie sammelt alle wichtigen Daten zu vorhandenen Kommentaren in einem neuen Arbeitsblatt.

```
Sub CommentsCopySheet()
   Dim rngTest    As Range
   Dim rngCell    As Range
   Dim wshTest    As Worksheet
   Dim wshNew     As Worksheet
   Dim lCount     As Long

   Set wshTest = ActiveSheet
   On Error Resume Next
   Set rngTest = wshTest.Cells.SpecialCells(xlCellTypeComments)
   On Error GoTo 0
   If rngTest Is Nothing Then
      MsgBox "Keine Kommentare gefunden!"
      Exit Sub
   End If

   Set wshNew = Worksheets.Add
   wshNew.Range("A1:D1").Value = _
      Array("Addresse", "Name", "Inhalt", "Kommentar")

   lCount = 1
   For Each rngCell In rngTest
      With wshNew
         lCount = lCount + 1
         On Error Resume Next
         .Cells(lCount, 1) = rngCell.Address
         .Cells(lCount, 2) = rngCell.Name.Name
         .Cells(lCount, 3) = rngCell.Value
         .Cells(lCount, 4) = rngCell.Comment.Text
      End With
   Next rngCell
End Sub
```

Das Ergebnis sieht dann so aus:

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Addresse | Name | Inhalt | Kommentar |
| 2 | $E$5 | | | 11.04.2019 (18:37) |
| 3 | $C$8 | | | 11.04.2019 (18:37) |
| 4 | $F$9 | | | 11.04.2019 (18:37) |
| 5 | | | | |

Tabelle2 | Tabelle1

Die nachfolgende Prozedur sammelt alle Kommentare einer Mappe in einem neuen Tabellenblatt.

```
Sub CommentsAllToSheet()
   Dim rngTest    As Range
   Dim rngCell    As Range
   Dim wshTest    As Worksheet
   Dim wshNew     As Worksheet
   Dim lCount     As Long

   Set wshNew = Worksheets.Add
   wshNew.Range("A1:D1").Value = _
      Array("Sheet", "Addresse", "Name", "Inhalt", "Kommentar")

   lCount = 1
   For Each wshTest In ActiveWorkbook.Worksheets
      On Error Resume Next
      Set rngTest = wshTest.Cells.SpecialCells(xlCellTypeComments)
      On Error GoTo 0
      If rngTest Is Nothing Or _
         wshTest.Name = wshNew.Name Then
         'do nothing
      Else
         For Each rngCell In rngTest
            With wshNew
               lCount = lCount + 1
               On Error Resume Next
               .Cells(lCount, 1) = wshTest.Name
               .Cells(lCount, 2) = rngCell.Address
               .Cells(lCount, 3) = rngCell.Name.Name
               .Cells(lCount, 4) = rngCell.Value
               .Cells(lCount, 5) = rngCell.Comment.Text
            End With
         Next rngCell
      End If
      Set rngTest = Nothing 'wichtig!
   Next wshTest
End Sub
```

Auch dazu ein Ergebnis:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Sheet | Addresse | Name | Inhalt | |
| 2 | Tabelle2 | $D$2 | | 11.04.2019 (18:37) | 11.04.2019 (18:40) |
| 3 | Tabelle2 | $D$3 | | 11.04.2019 (18:37) | 11.04.2019 (18:41) |
| 4 | Tabelle2 | $D$4 | | 11.04.2019 (18:37) | 11.04.2019 (18:41) |
| 5 | Tabelle1 | $E$5 | | | 11.04.2019 (18:37) |
| 6 | Tabelle1 | $C$8 | | | 11.04.2019 (18:37) |
| 7 | Tabelle1 | $F$9 | | | 11.04.2019 (18:37) |

Tabelle3 Tabelle2 Tabelle1

Die nachfolgende Prozedur überträgt die Kommentare der aktiven Tabelle in ein Worddokument.

```
Sub CommentsToWord()
   Dim comTest    As Comment
   Dim objWord    As Object

   'Word öffnen
   On Error Resume Next
   Set objWord = GetObject(, "Word.Application")
   If Err.numer <> 0 Then
      Err.Clear
      Set objWord = CreateObject("Word.application")
   End If

   With objWord
      .Visible = True
      .documents.Add DocumentType:=0
      For Each comTest In ActiveSheet.Comments
         .Selection.typetext comTest.Parent.Address & _
            vbTab & comTest.Text
         .Selection.typeparagraph
      Next
   End With
   Set objWord = Nothing
End Sub
```

Das neue Worddokument hat in unserem Beispiel folgenden Inhalt.

$E$5 11.04.2019 (18:37)

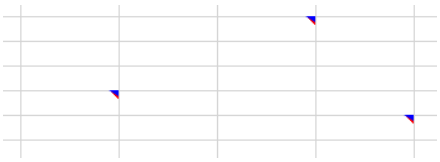
$C$8 11.04.2019 (18:37)

$F$9 11.04.2019 (18:37)

Der Ausdruck eines Tabellenblatts, das Kommentare enthält, zeigt die Kommentare nicht. Es gibt auch keine Möglichkeit, dieses Verhalten zu ändern. Als Workaround können Sie Dreiecke aus den Autoformen über die Kommentar-Indikatoren zeichnen.

```
Sub CoverCommentIndicator()
   Dim wshTest    As Worksheet
   Dim comTest    As Comment
   Dim rngCom     As Range
   Dim shpCom     As Shape
   Dim dShapeW    As Double 'shape width
   Dim dShapeH    As Double 'shape height
```

```
   Set wshTest = ActiveSheet
   dShapeW = 6
   dShapeH = 4
   For Each comTest In wshTest.Comments
      Set rngCom = comTest.Parent
      With rngCom
         Set shpCom = wshTest.Shapes. _
         AddShape(msoShapeRightTriangle, _
         rngCom.Offset(0, 1).Left - dShapeW, _
         .Top, dShapeW, dShapeH)
      End With
      With shpCom
         .Flip msoFlipVertical
         .Flip msoFlipHorizontal
         '10 = Rot, 12=Blau, 57=Grün
         .Fill.ForeColor.SchemeColor = 12
         .Fill.Visible = msoTrue
         .Fill.Solid
         .Line.Visible = msoFalse
      End With
   Next comTest
End Sub
```

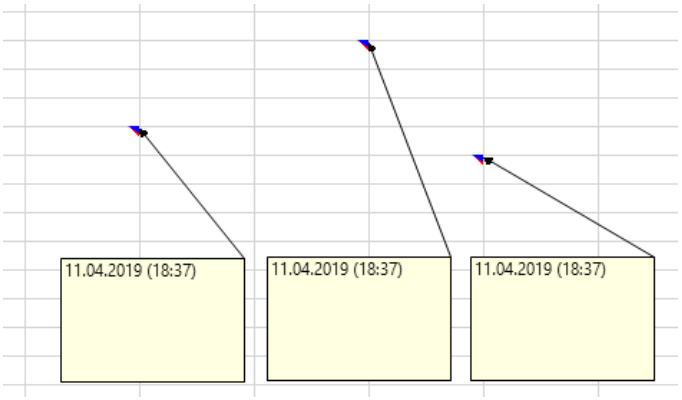


Mit der nachfolgenden Prozedur lassen sich die AutoFormen wieder entfernen.

```
Sub RemoveIndicatorShapes()
   Dim wshTest    As Worksheet
   Dim shpCom     As Shape

   Set wshTest = ActiveSheet
   For Each shpCom In wshTest.Shapes
      If Not shpCom.TopLeftCell.Comment Is Nothing Then
         If shpCom.AutoShapeType = msoShapeRightTriangle Then
            shpCom.Delete
         End If
      End If
   Next shpCom
End Sub
```

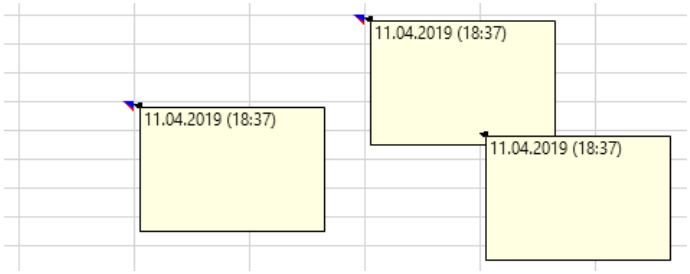
### 4 Positionen und Design

Kommentarrahmen lassen sich durch anfassen und verschieben mit der linken Maustaste anders positionieren.

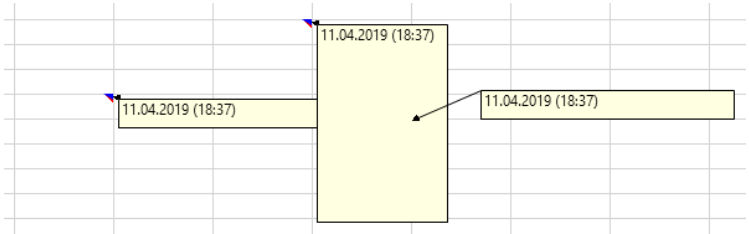



Die nachfolgende Prozedur positioniert die Kommentare wieder direkt neben die Zellen.

```
Sub CommentPosition()
   Dim comTest As Comment

   For Each comTest In ActiveSheet.Comments
      comTest.Shape.Top = comTest.Parent.Top + 3
      comTest.Shape.Left = comTest.Parent.Offset(0, 1).Left + 3
   Next
End Sub
```

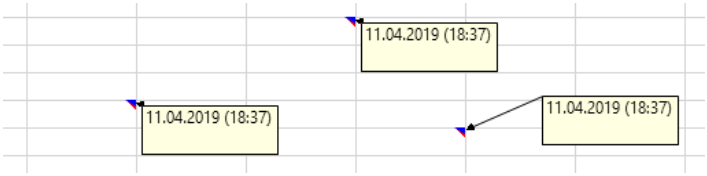



Auch die Größe eines Kommentarfeldes lässt sich durch Ziehen verändern.




Die nachfolgende Prozedur setzt die Feldbreite wieder auf den Ausgangswert zurück. Dabei muss notfalls auch die Höhe angepasst werden.

```
Sub CommentAutoWidth()
   Dim comTest As Comment
   Dim lArea   As Long

   For Each comTest In ActiveSheet.Comments
      With comTest
         .Shape.TextFrame.AutoSize = True
         If .Shape.Width > 300 Then
            lArea = .Shape.Width * .Shape.Height
            .Shape.Width = 200
            'der Faktor dient zur Anpassung
            .Shape.Height = (lArea / 200) * 1.1
         End If
      End With
   Next
End Sub
```




Nicht immer will man aber die Anpassung für alle Kommentare durchführen. Die nachfolgende Prozedur führt die Änderung nur bei den Zellen im markierten Bereich durch.

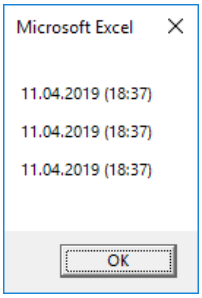
```
Sub CommentAutoArea()
   Dim rngTest As Range
   Dim rngSoll As Range
   Dim lArea   As Long

   Set rngSoll = Selection

   For Each rngTest In rngSoll.Cells
      If Not (rngTest.Comment Is Nothing) Then
         With rngTest.Comment
            .Shape.TextFrame.AutoSize = True
            If .Shape.Width > 300 Then
               lArea = .Shape.Width * .Shape.Height
               .Shape.Width = 200
               'der Faktor dient zur Anpassung
               .Shape.Height = (lArea / 200) * 1.1
            End If
         End With
      End If
   Next
End Sub
```

## 5 Kommentarlisten

Die Kommentare einer Tabelle lassen sich aus dem Listenobjekt mit einer *For Each Schleife* lesen.

```
Sub SearchComments()
   Dim sCom       As Object
   Dim sText      As String

   sText = ""
   For Each sCom In Worksheets("Tabelle1").Comments
      sText = sText & sCom.Text & vbLf
   Next
   MsgBox sText
End Sub
```

Das Ergebnis sieht wie folgt aus.



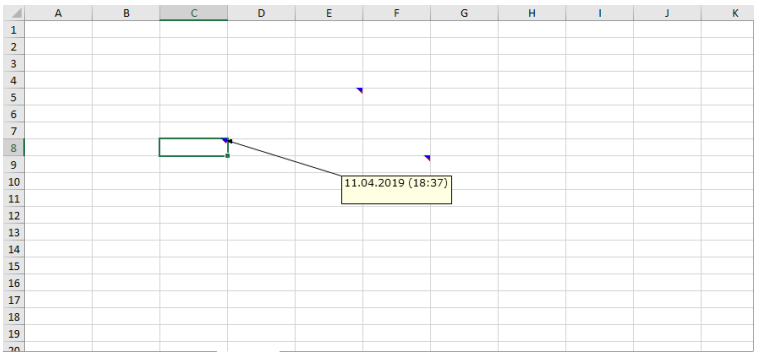

Über die Kommentarliste lassen sich auch alle Kommentare einheitlich formatieren. Die nachfolgende Prozedur stellt für alle Kommentare den Schrifttyp Verdana in der Größe 9 ein.

```
Sub CommentsFormatAll()
   Dim wshTab     As Worksheet
   Dim comTest    As Comment

   For Each wshTab In ActiveWorkbook.Worksheets
      For Each comTest In wshTab.Comments
         With comTest.Shape.TextFrame.Characters.Font
            .Name = "Verdana"
            .Size = 9
         End With
      Next comTest
   Next wshTab
End Sub
```

## 6 Kommentare und Ereignisse

Die nachfolgenden Anweisungen sollten Sie ausschließlich nur in die Ereignissprozedur *Worksheet_SelectionChange* eines Tabellenblattes setzen. *SelectionChange* wird immer dann aufgerufen, wenn eine Zelle oder ein Zellbereich markiert wird. Verfügt die Zelle über einen Kommentar, oder die linke obere Zelle eines Zellbereichs, dann wird der Kommentar mittig in das sichtbare Fenster der Anwendung gestellt.

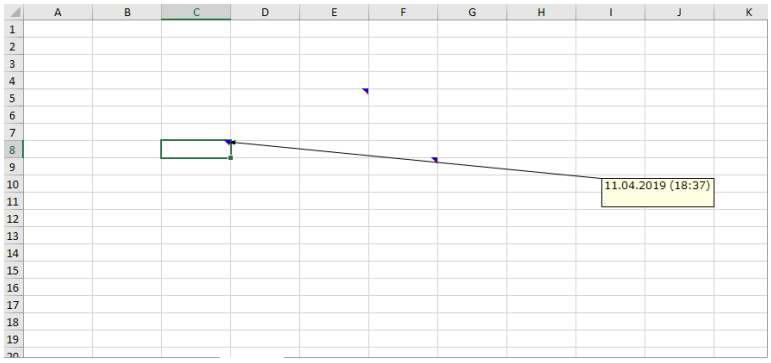
```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
   Dim rngCell    As Range
   Dim lTop       As Long
   Dim lWidth     As Long
   Dim comTest    As Comment
   Dim shpTest    As Shape

   Application.DisplayCommentIndicator = xlCommentIndicatorOnly
   Set rngCell = ActiveWindow.VisibleRange
   lTop = rngCell.Top + rngCell.Height / 2
   lWidth = rngCell.Left + rngCell.Width / 2
   If ActiveCell.Comment Is Nothing Then
      'do nothing
   Else
      Set comTest = ActiveCell.Comment
      Set shpTest = comTest.Shape
      shpTest.Top = lTop - shpTest.Height / 2
      shpTest.Left = lWidth - shpTest.Width / 2
      comTest.Visible = True
   End If
End Sub
```




Es kann aber auch eine andere Position für die Kommentare gewählt werden, so wie in der nachfolgenden Prozedur der rechte Rand der aktiven Seite.

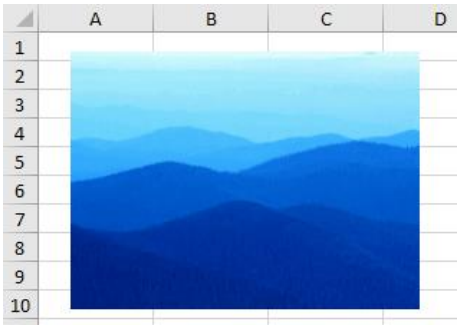
```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
   Dim rngCell    As Range
   Dim lTop       As Long
   Dim lGap       As Long
```

```
    Dim comTest    As Comment
    Dim shpTest    As Shape

    Application.DisplayCommentIndicator = xlCommentIndicatorOnly
    Set rngCell = ActiveWindow.VisibleRange
    lTop = rngCell.Top + rngCell.Height / 2
    lGap = 60
    If ActiveCell.Comment Is Nothing Then
       'do nothing
    Else
       Set comTest = ActiveCell.Comment
       Set shpTest = comTest.Shape
       shpTest.Top = lTop - shpTest.Height / 2
       shpTest.Left = rngCell.Width - shpTest.Width - lGap
       comTest.Visible = True
    End If
End Sub
```




## 7 Bilder in Kommentarrahmen

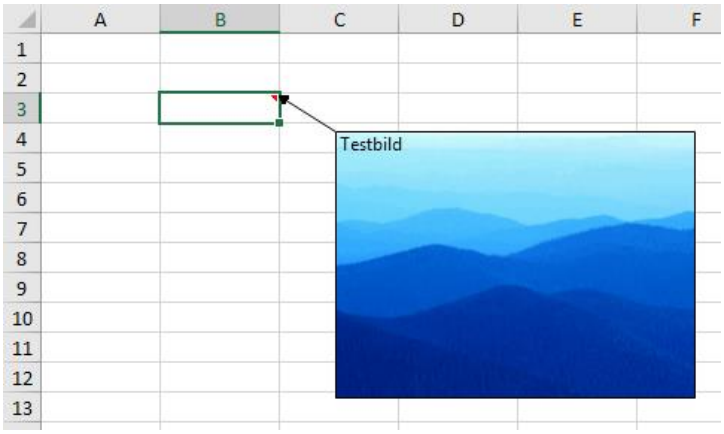
Kommentarrahmen können nicht nur Texte sondern auch Bilder aufnehmen. Zur Nutzung der nachfolgenden Prozedur gehen sie wie folgt vor. Fügen Sie ein Bild in ihr aktuelles Tabellenblatt ein. Passen sie es der gewünschten Größe an. Markieren Sie dann nacheinander die Zelle für das Kommentarbild und das Bild. Dadurch wird die Zelle für den Kommentar aktiv und dann das Bild.



Dann rufen Sie die nachfolgende Prozedur auf.

```
Sub PictureIntoComment()
   Dim chtTest    As ChartObject
   Dim dWidth     As Double
   Dim dHeight    As Double
   Dim wshTest    As Worksheet
   Dim sName      As String
   Dim comTest    As Comment
   Dim sPath      As String
   Dim sFile      As String
   Dim rngTest    As Range

   Set wshTest = ActiveSheet
   Set rngTest = ActiveCell
   sPath = ThisWorkbook.Path & "\"
   sName = "E_11_TestBild.png"
   sFile = sPath & sName
   dWidth = Selection.Width
   dHeight = Selection.Height
   Selection.Cut 'Bild in die Zwischenablage

'Bildobjekt erstellen
   Set chtTest = wshTest.ChartObjects.Add( _
      Left:=rngTest.Left, _
      Top:=rngTest.Top, _
      Width:=dWidth, _
      Height:=dHeight)
   chtTest.Chart.Paste 'Bild laden
   rngTest.Activate
   chtTest.Chart.Export sFile 'Bild exportieren
   chtTest.Delete 'Darstellung löschen

'Kommentar erstellen
   Set comTest = rngTest.AddComment
   comTest.Text Text:="Testbild"
   With comTest.Shape
      .Fill.UserPicture sFile
      .Width = dWidth
      .Height = dHeight
   End With

   Set comTest = Nothing
   Set chtTest = Nothing
   Set rngTest = Nothing
   Set wshTest = Nothing
End Sub
```

Nach der Zuordnung kann die Größe wie bei jedem anderen Kommentar angepasst werden.


A
B
C
D
E
F
1
2
3
4
5
6
7
8
9
10
11
12
13
Testbild