Notizen

Autor & Copyright: Dipl.-Ing. Harald Nahrstedt

Version: 2016 / 2019 / 2021 / 365

Erstellungsdatum: 15.10.2011

Überarbeitung: 17.01.2024

Beschreibung:

Excel stellt für jede Zelle einen Notizbereich zur Verfügung. In älteren Versionen waren es noch Kommentare. Darin kann Hinweistext gespeichert werden, der jedes Mal bei der Aktivierung der Zelle erscheint oder dauerhaft angezeigt werden kann.

Anwendungs-Datei: AE-011_Notizen.xlsm

1 Notizvorgabe

Mit dem Erstellen einer Notiz erscheint im Notizfeld der Benutzername. Um diesen zu ändern, oder um einen anderen Text einzustellen, genügt die folgende kleine Prozedur.

Codeliste 1. Notiz erzeugen

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

Das Ergebnis sieht dann wie folgt aus (Bild 1). Natürlich kann jeder beliebige Text gewählt werden.

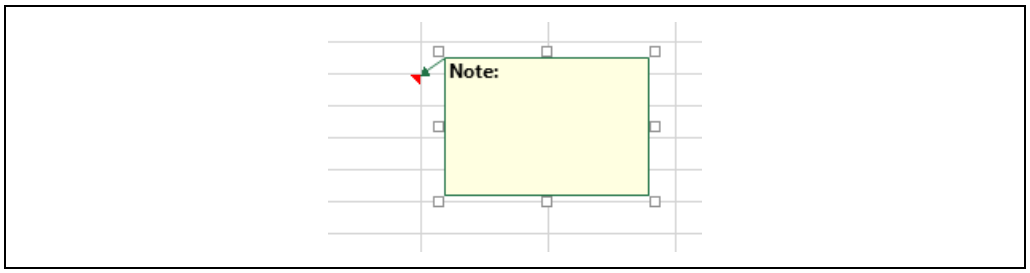


Bild 1. Einem Notizfeld Text zuordnen

2 Eine Notiz erstellen

Eine Notiz ist ein Objekt der Klasse *Comments* und kann mit dem *Set*-Befehl instanziiert werden.

Codeliste 2. Notiz zuordnen

```
Sub CommentAddOrNot()
   Dim comTest As Comment

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

Die Notiz wird der markierten Zelle zugeordnet (Bild 2).

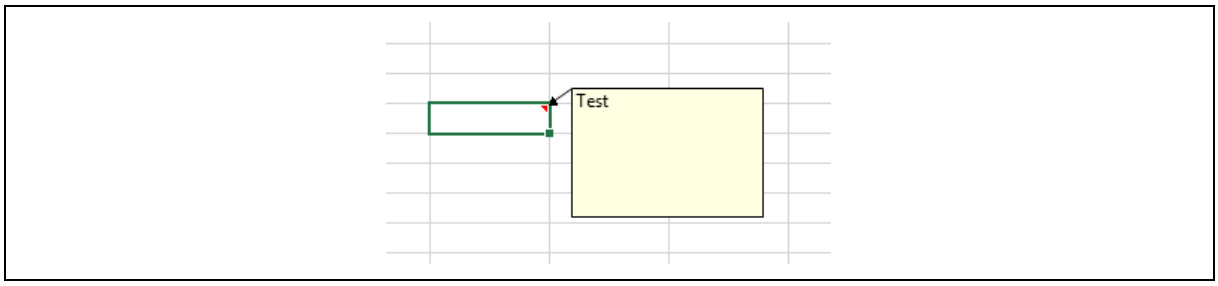


Bild 2. Zelle mit Notiz

Etwas komfortabler ist die folgende Prozedur, bei der man sofort die neue Notiz schreiben kann (Bild 3).

Codeliste 3. Notiz erstellen

```
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
```

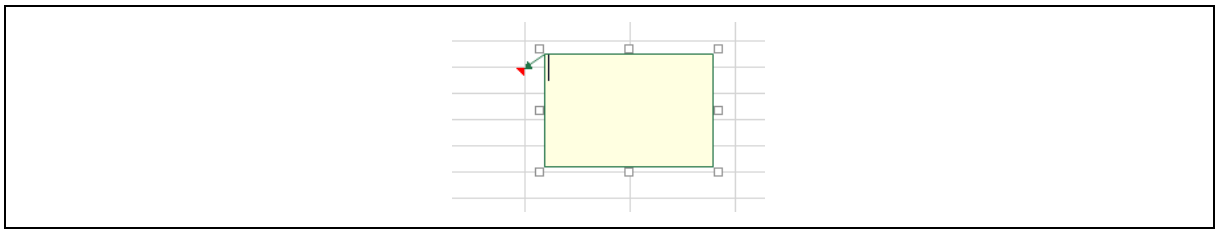


Bild 3. Notizfeld zur Texteingabe

Die nachfolgende Prozedur formatiert zusätzliche die Notiz in der Schriftart Verdana.

Codeliste 4. Notizdesign erstellen

```
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
```

Neben der Schriftgröße von 12 pt, wird die Schrift fett und in der Schriftfarbe Rot dargestellt (Bild 4).

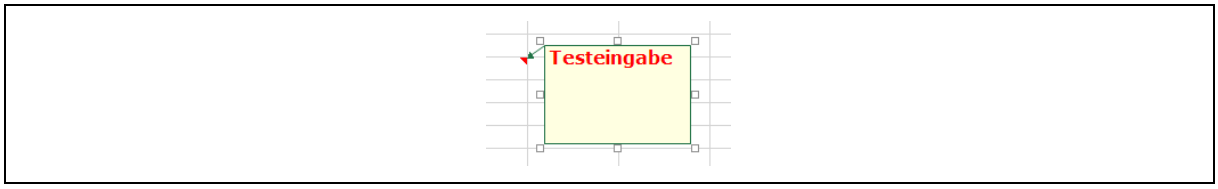


Bild 4. Notizdesign

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

Codeliste 5. Detailliertes Notizdesign erstellen

```
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
```

Außerdem werden die Werte fett dargestellt (Bild 5).

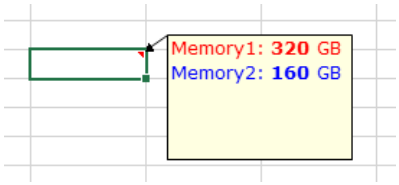


Bild 5. Detaillierter Notiztext

Oft möchte man das Datum und die Uhrzeit mit in die Notiz einbinden, damit man weiß, wann der Hinweis geschrieben wurde. Die nachfolgende Prozedur erstellt aktuell Datum und Uhrzeit in eine neue oder bereits vorhandene Notiz.

Codeliste 6. Notiztext mit Datum und Uhrzeit

```
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
```

Auch hier muss der Zellbereich natürlich vorher markiert sein (Bild 6).

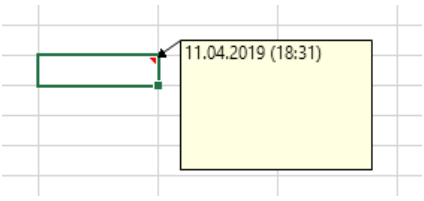


Bild 6. Datum und Uhrzeit in der Notiz

3 Notizen zeigen und kopieren

Die folgende Prozedur zeigt alle Notizen der aktiven Tabelle an, auch wenn sie ausgeblendet waren.

Codeliste 7. Die Prozedur CommentAddNow zeigt alle Notizen

```
Sub CommentsOnSheet()
   Dim comTest As Comment

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

Angezeigt werden sichtbare und unsichtbare Notizen (Bild 7).

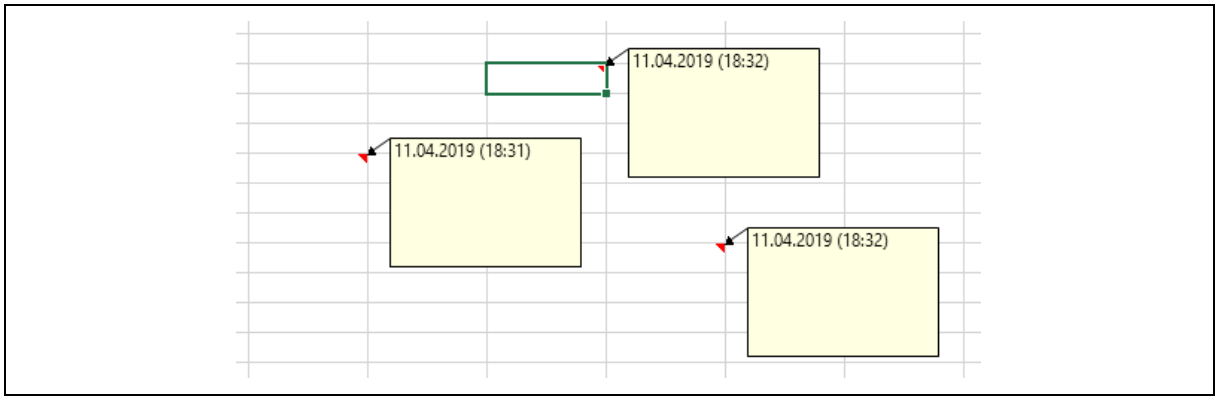


Bild 7. Anzeige aller Notizen

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

Codeliste 8. Die Prozedur CommentsCopyCell kopiert den Inhalt

```
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 Notiz 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
   Set wshTest = Nothing
   Set rngTest = Nothing
End Sub
```

Der kopierte Text wird angepasst (Bild 8).

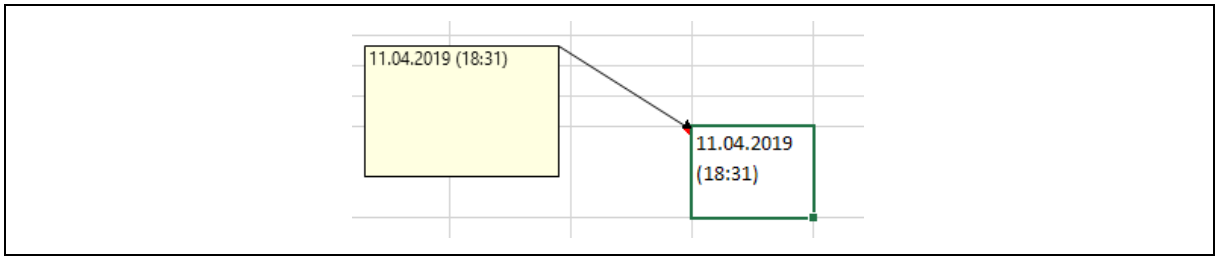


Bild 8. Kopierter Notiztext

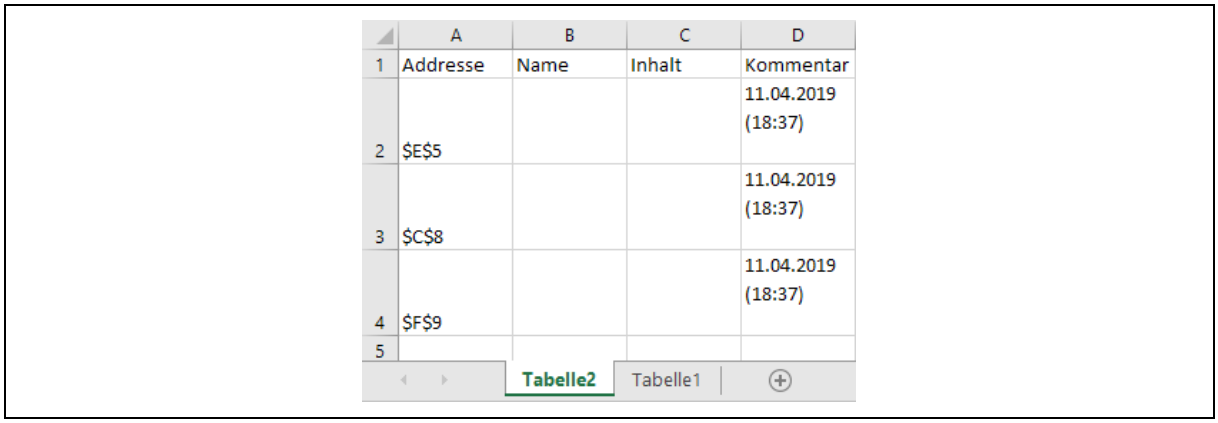
Eine regelrechte Verwaltung von Daten erlaubt die folgende Prozedur.

Codeliste 9. Die Prozedur CommentsCopySheet kopiert Notizen in eine Tabelle

```
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 Notizen gefunden!"
      Exit Sub
   End If
   Set wshNew = Worksheets.Add
   wshNew.Range("A1:D1").Value = _
      Array("Addresse", "Name", "Inhalt", "Notiz")
   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
```

Sie sammelt alle wichtigen Daten zu vorhandenen Notizen in einem neuen Arbeitsblatt (Bild 9).



	A	B	C	D
1	Addresse	Name	Inhalt	Kommentar
2	E5			11.04.2019 (18:37)
3	C8			11.04.2019 (18:37)
4	F9			11.04.2019 (18:37)
5				

Tabelle2 Tabelle1

Bild 9. Sammlung der Notizen in einer Tabelle

Die folgende Prozedur sammelt alle Notizen einer Mappe in einem neuen Tabellenblatt.

Codeliste 10. Die Prozedur CommentsCopySheet kopiert Notizen der Arbeitsmappe in eine Tabelle

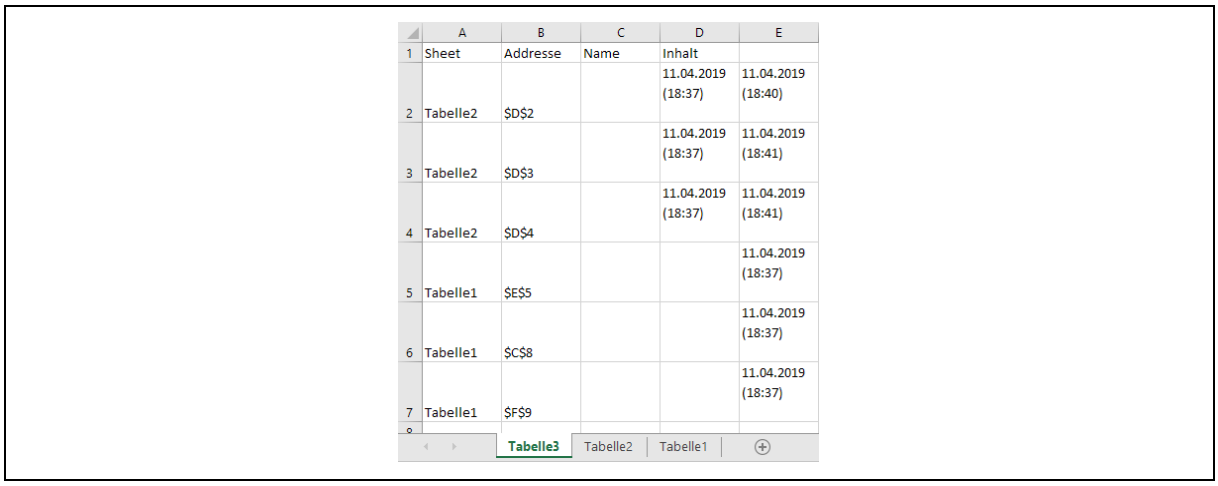
```
Sub CommentsAllToSheet()
   Dim rngTest    As Range
   Dim rngCell    As Range
   Dim wshTest    As Worksheet
   Dim wshNew     As Worksheet
   Dim lCount     As Long

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

   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
```

Das Ergebnis ist folgend dargestellt (Bild 10).



	A	B	C	D	E
1	Sheet	Addresse	Name	Inhalt	
2	Tabelle2	D2		11.04.2019 (18:37)	11.04.2019 (18:40)
3	Tabelle2	D3		11.04.2019 (18:37)	11.04.2019 (18:41)
4	Tabelle2	D4		11.04.2019 (18:37)	11.04.2019 (18:41)
5	Tabelle1	E5			11.04.2019 (18:37)
6	Tabelle1	C8			11.04.2019 (18:37)
7	Tabelle1	F9			11.04.2019 (18:37)

Tabelle3 | Tabelle2 | Tabelle1

Bild 10. Gesammelte Notizen

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

Codeliste 11. Die Prozedur CommentsToWord kopiert die gesammelten Notizen in ein Word-Dokument

```
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 Word-Dokument hat in unserem Beispiel folgenden Inhalt (Bild 11).

E5 11.04.2019 (18:37)

C8 11.04.2019 (18:37)

F9 11.04.2019 (18:37)

Bild 11. Einträge im Word-Dokument

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

Codeliste 12. Die Prozedur CoverCommentIndicator verwendet Notiz-Indikatoren

```
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 Prozedur werden die Indikatoren (Dreiecke) sichtbar (Bild 12).

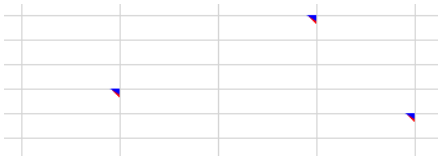


Bild 12. Anzeige der Notiz-Indikatoren

Mit der folgenden Prozedur lassen sich die AutoFormen wieder entfernen.

Codeliste 13. Die Prozedur RemoveIndicatorShapes entfernt Notiz-Indikatoren

```
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

Notizrahmen lassen sich durch Anfassen und Verschieben mit der linken Maustaste anders positionieren (Bild 13).

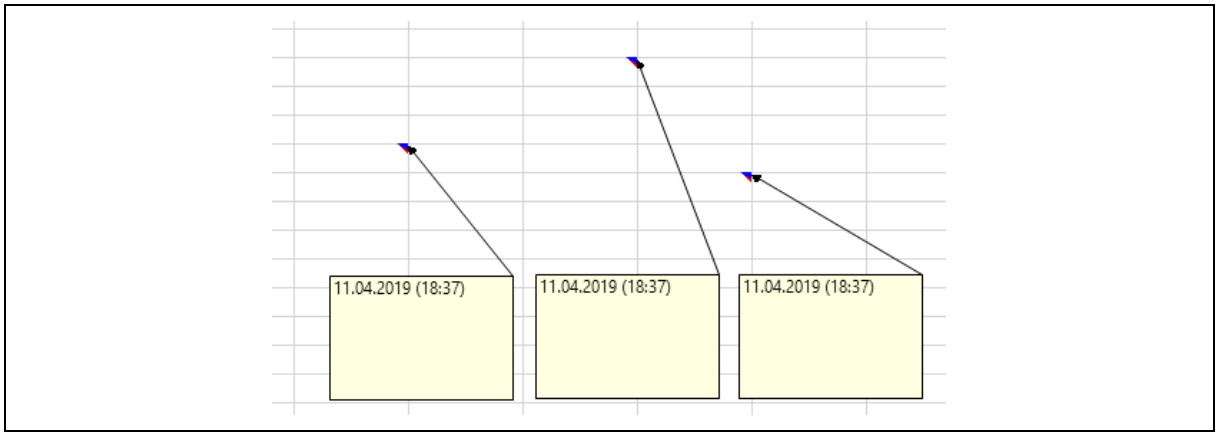


Bild 13. Manuell verschobene Notizen

Die folgende Prozedur positioniert die Notizen wieder direkt neben die Zellen.

Codeliste 14. Die Prozedur CommentPosition setzt die Notizen zurück auf ihre Position

```
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
```

Das Ergebnis zeigt die Notizen direkt neben die Zelle (Bild 14).

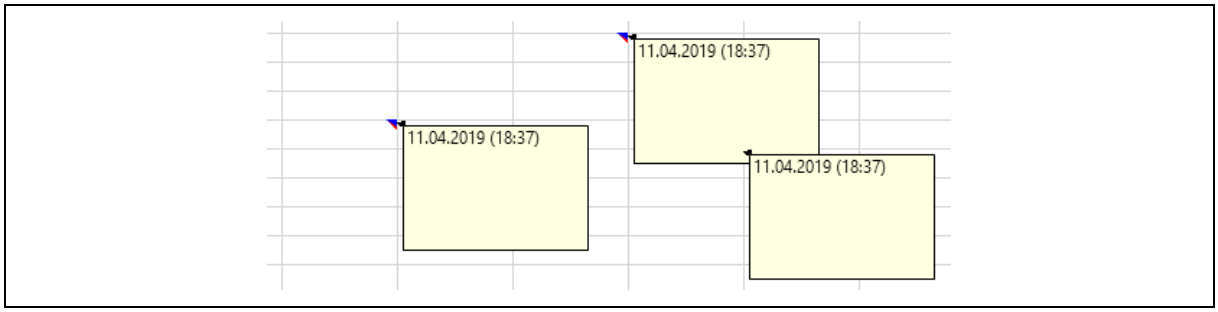


Bild 14. Zurückgesetzte Notiz-Positionen

Auch die Größe eines Notizfeldes lässt sich durch Ziehen verändern (Bild 15).

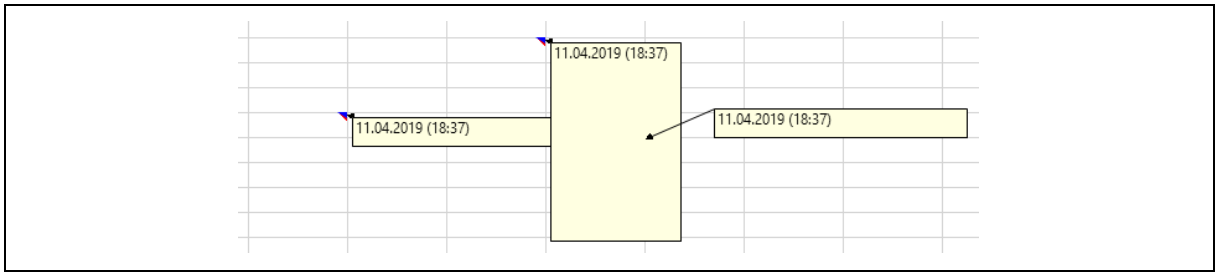


Bild 15. Manuell veränderte Notizrahmen

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

Codeliste 15. Die Prozedur CommentAutoWidth setzt die Rahmenbreite der Notiz zurück

```
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
```

Dadurch werden einheitliche Notizrahmenbreiten erzeugt (Bild 16).

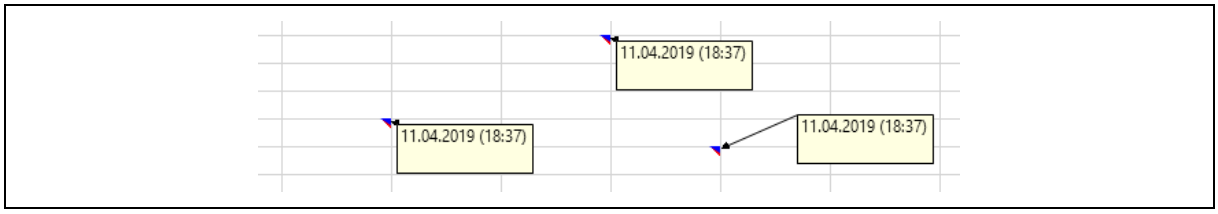


Bild 16. Gesetzte Notizrahmenbreiten

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

Codeliste 16. Die Prozedur CommentAutoArea setzt die Rahmenbreite der Notizen im Bereich zurück

```
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 Notizlisten

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

Codeliste 17. Die Prozedur SearchComments findet alle Notizen einer Tabelle

```
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 wird mit einer MsgBox ausgegeben (Bild 17).

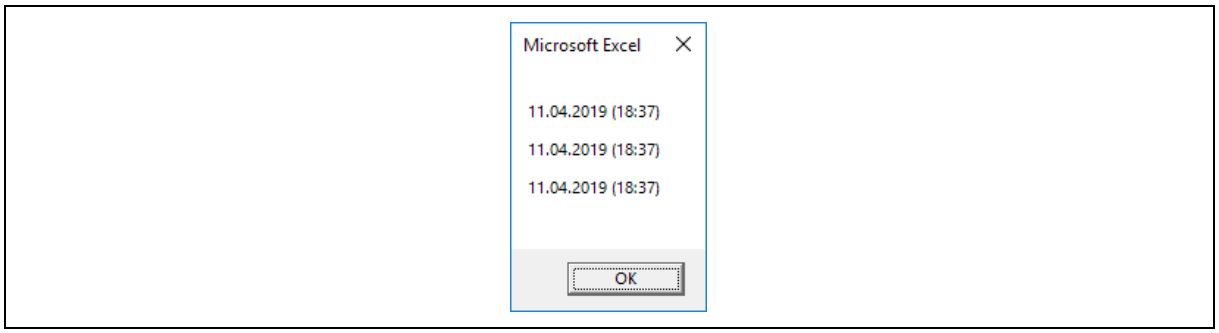


Bild 17. Ausgabebeispiel

Über die Notizenliste lassen sich auch alle Notizen einheitlich formatieren. Die folgende Prozedur stellt für alle Notizen den Schrifttyp Verdana in der Größe 9 ein.

Codeliste 18. Die Prozedur CommentsFormatAll formatiert alle gefundenen Notizen

```
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 Notizen und Ereignisse

Die folgenden 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 eine Notiz, oder die linke obere Zelle eines Zellbereichs, dann wird die Notiz mittig in das sichtbare Fenster der Anwendung gestellt.

Codeliste 19. Die Ereignis-Prozedur Worksheet_SelectionChange verschiebt Notizen mittig

```
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
```

Darstellung einer Notiz im Beispiel (Bild 18).

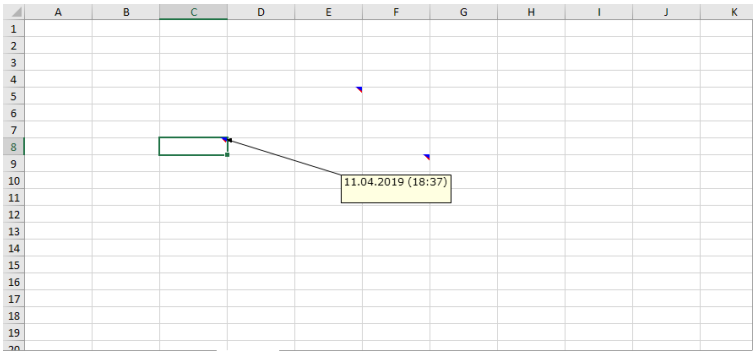


Bild 18. Mittig gesetzte Notiz

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

Codeliste 20. Die Ereignis-Prozedur Worksheet_SelectionChange verschiebt Notizen an den Rand

```
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
```

Darstellung einer Notiz im Beispiel (Bild 19).

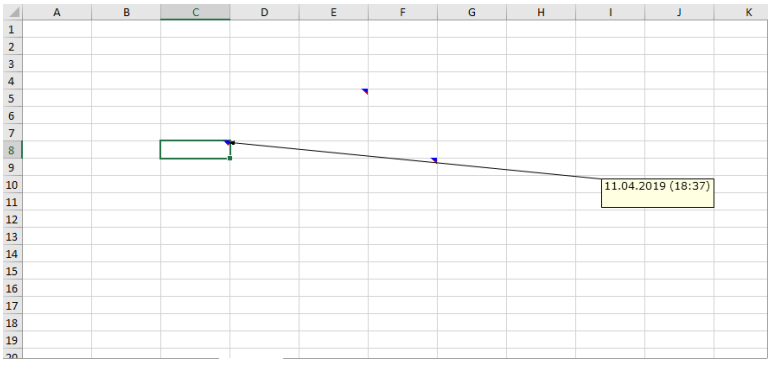


Bild 19. An den Rand gesetzte Notiz

7 Bilder in Notizrahmen

Notizrahmen können nicht nur Texte sondern auch Bilder aufnehmen. Zur Nutzung der folgenden 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 Notizbild und das Bild. Dadurch wird die Zelle für die Notiz aktiv und dann das Bild (Bild 20).

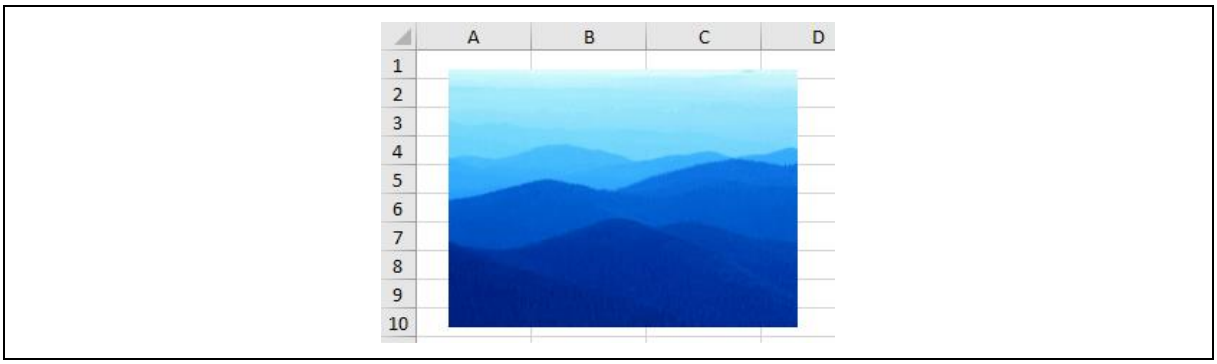


Bild 20. Beispielbild

Dann rufen Sie die nachfolgende Prozedur auf.

Codeliste 21. Die Prozedur PictureIntoComment setzt ein Bild in den Notizrahmen

```
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

'Notiz 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 jeder anderen Notiz angepasst werden (Bild 21).

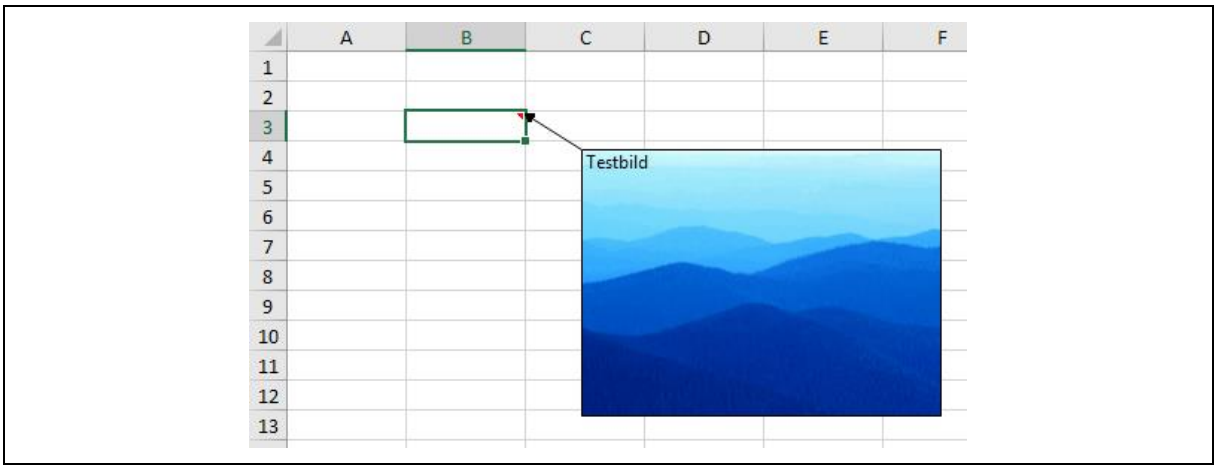


Bild 21. Bild im Notizrahmen