Workshop 8: VBA-Programmierung mit MS Excel

Benutzerformular aufrufen

(**Bitte erst in der 8. Sitzung aufrufen**)

1 VBA als Automatisierungssprache

In diesem Workshop wird gezeigt, wie mit *Visual Basic for Applications* (VBA) drei Office-Anwendungen von Microsoft integriert werden können, wobei *Excel* im Mittelpunkt steht. Im Einzelnen geht es um folgende Automatisierungsaufgaben (s. dazu Abb. 1).

Aus Word heraus …

- per Knopfdruck eine Arbeitsmappe in *Excel* zu erzeugen,
- *Word*-Tabellen in *Excel*-Tabellenblätter umzuwandeln,
- den im *Word*-Dokument enthaltenen VBA-Code nach *Excel* zu exportieren.

In Excel …

- viele verschiedene Diagrammtypen automatisch zu erzeugen und
- die erzeugten Diagramme auf Knopfdruck nach *Powerpoint* zu exportieren.

Abb. 1 veranschaulicht das Vorhaben.

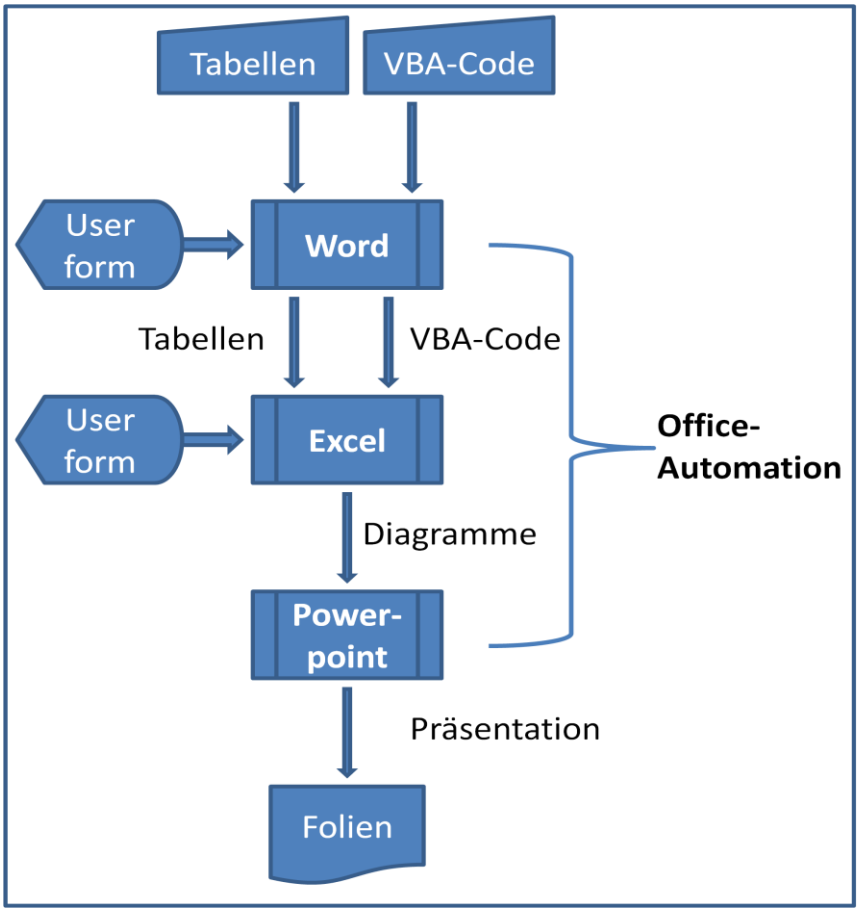

Abb. 1: Office-Automatisierung mit VBA

2 Diagrammerstellung automatisieren

2.1 Datenanordnung auf Tabellenblatt

Zur Diagrammerstellung auf Knopfdruck müssen die Basisdaten im zugrunde liegenden Tabellenblatt richtig angeordnet sein, sodass sie für den gewünschten Diagrammtyp verwendet werden können. Im Artikel *Erstellen eines Diagramm*[1] 2s können Tipps von Microsoft nachgelesen werden, welche Datenanordnung für welchen Grund-Diagrammtyp

- Säulen-, Balken-, Linien-, Oberflächen- oder Netzdiagramm
- Kreis- oder Ringdiagramm
- Punkt(X/Y)- oder Blasendiagramm
- Kursdiagramm

erforderlich ist.

2.2 Diagrammelemente

- Beschriftungen der Einheiten und Rubriken der X-Achse (Unterteilung und Beschriftung werden i. d. R. aus Zeilen oder Spaltenbeschriftungen der zugrundliegenden Basisdaten erzeugt.)
- Einheiten der Y-Achse (Skalierung und Beschriftung erstellt Excel automatisch aus den zugeordneten Tabellenwerten)

[1] http://office.microsoft.com/de-de/excel-help/erstellen-eines-diagramms-HP005199491.aspx
[2] http://office.microsoft.com/de-de/help/beispiele-fur-diagrammtypen-HP005262319.aspx

- Legende (erklärt die Zuordnung der einzelnen Y-Datenreihen)
- Diagrammtitel (kann individuell erfasst werden)
- Titel der Achsentitel (neben der Skalierungsbeschriftung kann ein zusätzlicher beschreibender Titel erfasst werden.)
- Titel der X-Achse (dient zur Beschriftung der Rubriken)

2.3 Diagrammblätter und eingebettete Diagramme

Excel-Arbeitsmappen können Diagramme enthalten und zwar

- entweder als eingebettetes Diagramm (**`ChartObject`**)
- oder als separates Diagrammblatt (`Chart`)

Ein eingebettetes Diagramm ist ein Grafikobjekt, das als Bestandteil des Tabellenblattes gespeichert wird, in dem es erstellt wurde. Eingebettete Diagramme können nachträglich in Diagrammblätter umgewandelt werden.

Ein Diagrammblatt ist ein separates Blatt mit einem eigenen Blattnamen. in einer Arbeitsmappe.

3 Mit Diagrammblättern arbeiten

Mit den in Tabelle 1 enthaltenen Daten soll in Excel ein sogen. Punkt(XY)-Diagramm automatisch erzeugt werden. Ein solches Diagramm zeigt die Beziehungen zwischen den numerischen Werten in mehreren Datenreihen oder stellt zwei Zahlengruppen als eine Reihe von XY-Koordinaten dar.

	Q1	Q2	Q3	Q4
Westen	500	555	555	600
Osten	600	625	674	700
Norden	451	471	491	510
Süden	800	751	776	790

Tab. 1: Wertetabelle für eingebettetes **Punktdiagramm**

3.1 Benutzerformular in Word

Die Überführung dieser Tabelle und der noch folgenden Word-Tabellen in die entsprechenden Excel-Tabellenblätter soll mit einem VBA-Programm erfolgen. Dieses Programm wird mittels Benutzerformular (`Userform`) gestartet (s. Abb. 2), indem die entsprechende Schaltfläche betätigt wird. Abb. 2 gibt nicht nur das Aussehen des Benutzerformulars wieder, sondern auch die Benennung der zugehörigen Steuerelemente. Die mit dem Präfix `cmd` versehenen Steuerelemente, nämlich `cmdCloseForm` und `cmdExportToExcel`, gehören jeweils zu einer Ereignisprozedur. Diese werden jeweils durch das sogen. Click-Ereignis ausgelöst werden. Der VBA-Code, der zu diesen beiden Ereignisprozeduren gehört, wird im Folgenden wiedergegeben:

```
Private Sub cmdCloseForm_Click()
   ' Aktuelles Fenster wieder anzeigen
   ActiveWindow.Visible = True
   Me.Hide      ' Benutzerformular ausblenden
   Unload Me    ' Benutzerformular schließen
End Sub
```

```
Private Sub cmdExportToExcel_Click()
   Call CreateExcelApp         ' Excel-Anwnedung starten
   Call WordTablesToExcel      ' Word-Tabllen nach Excel exportieren
   Call WordCodeLinesToExcel   ' Word-Code nach Excel exportieren
   Call QuitExcelApp           ' Excel-Anwendung schließen
```

```
    If Dir(strFullPath) <> vbNullString Then
       Me!txtFullPath.Value = strFullPath
    End If
    Me!cmdCloseForm.SetFocus
End Sub
```

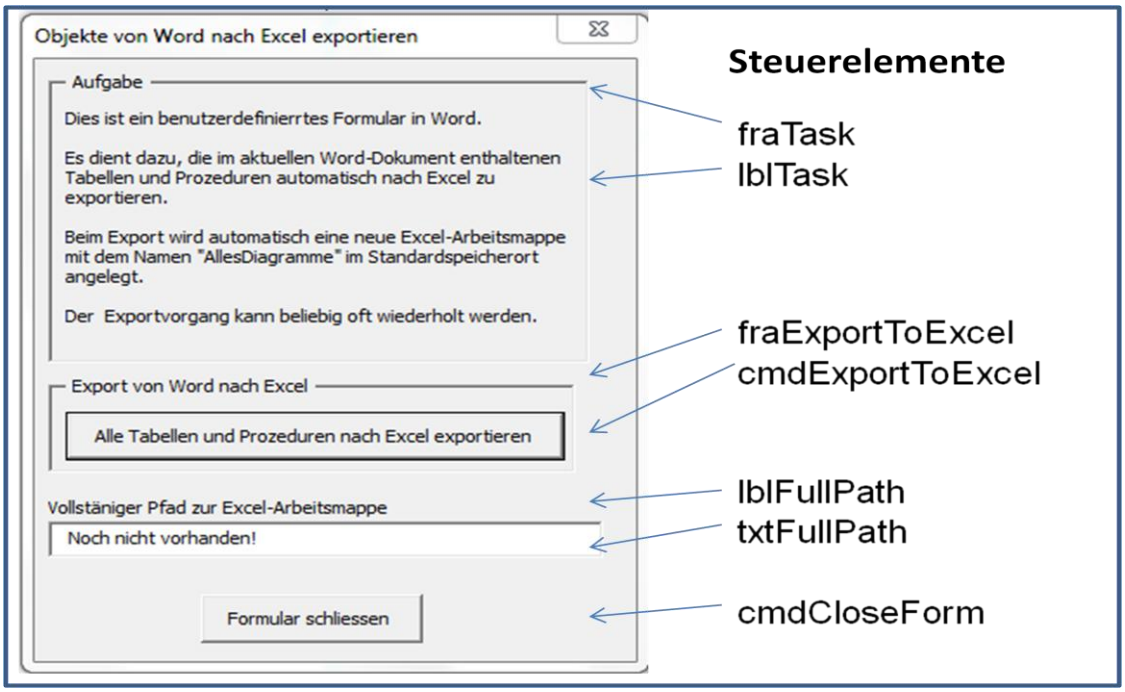

Abb. 2: Benutzerdefiniertes Formular (`UserForm`) im Word-Dokument

Das benutzerdefinierte Formular (s. Abb. 2) wird gemäß Namenskonventionen `frmBenutzer` genannt (s. Abb. 3).

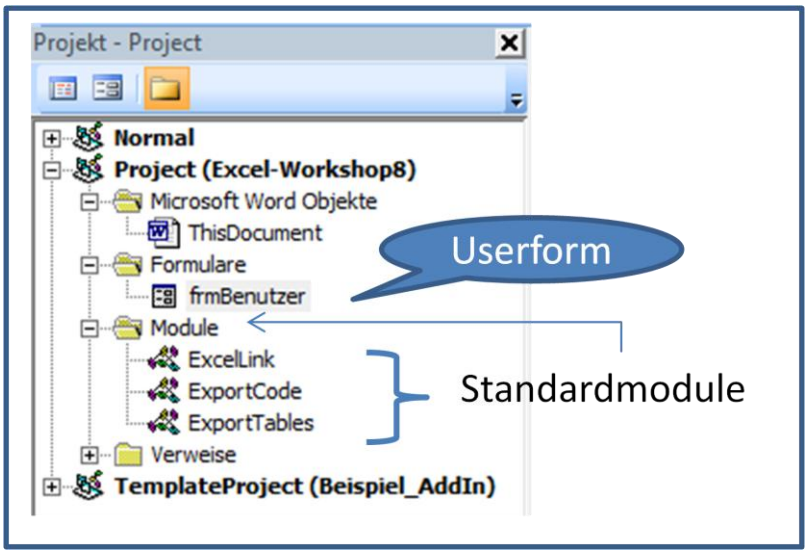

Abb. 3: `Userform` und Standardmodule in Projekt-Explorer von Word

Im Standardmodul `ExportTables` (s. Abb. 3) werden *offene* Textmarken (`Bookmarks`) verwendet, die für jede Wertetabelle in diesem Dokument exakt gesetzt sein müssen (s. Abb. 4 in Verbindung mit Tab. 3)

Im Standardmodul `ExportCode` (s. Abb. 3) werden die CodeZeilen der in diesem Dokument enthalten Prozeduren zur Diagrammerstellung nach Excel exportiert. Dafür werden *geschlossene* Textmarken benutzt, die den Code der jeweiligen Prozedur einschließen. Die geschlossenen Textmarken werden dynamisch eingefügt und auch wieder gelöscht.

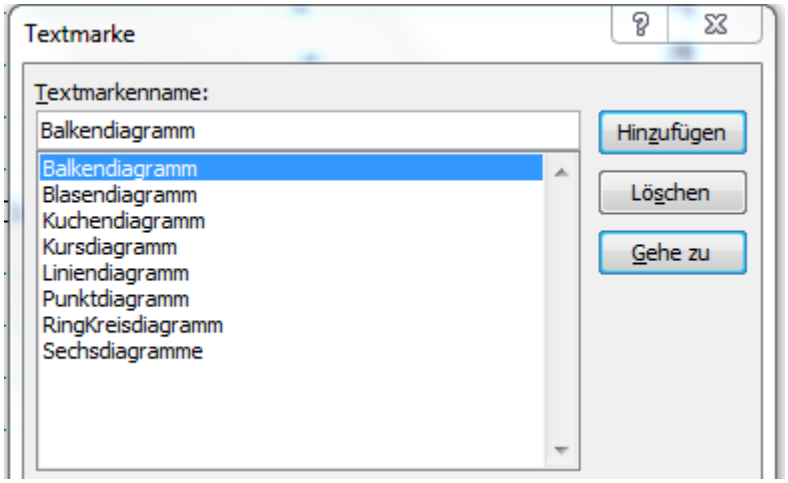

Abb. 4: Textmarken für Wertetabellen in diesem Dokument

Die Betätigung der Schaltfläche cmdExportToExcel im Formular frmBenutzer bewirkt, dass in der aktuellen Excel-Arbeitsmappe das Tabellenblatt Punktdiagramm erstellt wird. Die Word-Tabelle sieht als Excel-Tabellenblatt wie folgt aus (s.. Abb. 5):

	A	B	C	D	E
1		Q1	Q2	Q3	Q4
2	Westen	500	555	555	600
3	Osten	600	625	674	700
4	Norden	451	471	491	510
5	Süden	800	751	776	790

Abb. 5: Nach *Excel* exportierte Datentabelle aus *Word*

Mit der Prozedur `Punktdiagramm` kann automatisch ein eingebettetes Punkt(XY)-Diagramm erzeugt werden. Dazu muss es vorher als Standardmodul in die aktuelle Excel-Arbeitsmappe kopiert werden. Diese Aufgabe erfolgt automatisch per VBA-Programm zusammen mit der dazugehörigen Wertetabelle (s. Tab. 1)

```
Sub Punktdiagramm()
   Dim objCht As ChartObject ' Eingebettetes Diagramm
   ' Eingebettetes Diagramm hinzufügen
   Set objCht = Worksheets("Punktdiagramm").ChartObjects.Add(0, 100, 400, 250)
   With objCht.Chart
      .SetSourceData Source:=Sheets("Punktdiagramm").Range("A1:E5")
      ' Diagrammtyp: Punkt(X/Y)-Diagramm
      .ChartType = xlXYScatterLines
      .ChartArea.Select
      .HasTitle = True
      .ChartTitle.Characters.Text = "Absatzmengen"
      .Axes(xlCategory, xlPrimary).HasTitle = True
      .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Quartal"
      .Axes(xlValue, xlPrimary).HasTitle = True
      .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Menge"
   End With
End Sub
```

Das mit der Prozedur `Punktdiagramm` erzeugte eingebettete Diagramm ist in Abb. 6 dargestellt.

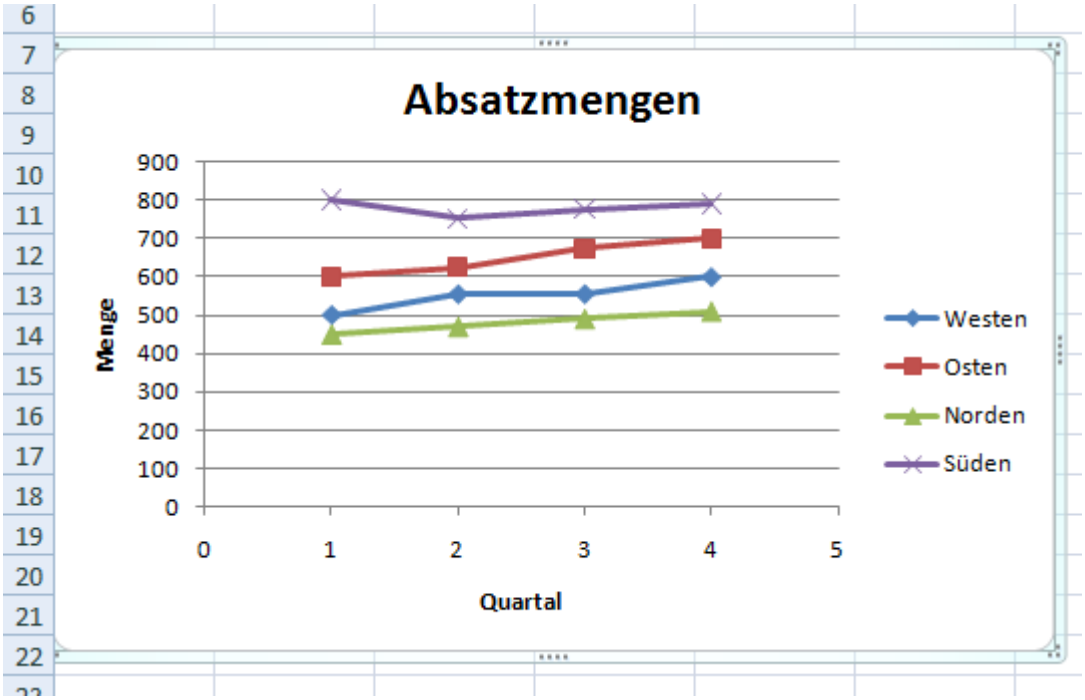

Abb. 6: Eingebettetes Punkt(XY)-Diagramm in Excel

3.2 Benutzerformular in Excel

Die Excel-Arbeitsmappe `Benutzerformular.xlsm` enthält ein benutzerdefiniertes Formular namens `frmUsr`. (s. Abb. 8). Dieses Formular enthält vier 4 Befehlsschaltflächen (s. Abb. 7): Ihre jeweilige Aufgabe kann der folgenden Tabelle entnommen werden.

Befehlsschaltfläche	Aufgabe
cmdRunProcs	Erstellt Diagramme in Excel mit den importierten Tabellen aus Word und jeweils dazugehörige Prozeduren zur Diagramm-Erstellung.
cmdDelete	Löscht in Excel per Optionsgruppe bestimmte Diagrammtypen, die vorher mit `cmdRunProcs` erstellt wurden.
cmdExortToPpt	Exportiert auf Knopfdruck Excel-Diagramme als Bilder in eine neu erstellte Powerpoint-Präsentation
cmdCloseForm	Schließt das benutzerdefinierte Formular `frmUsr`

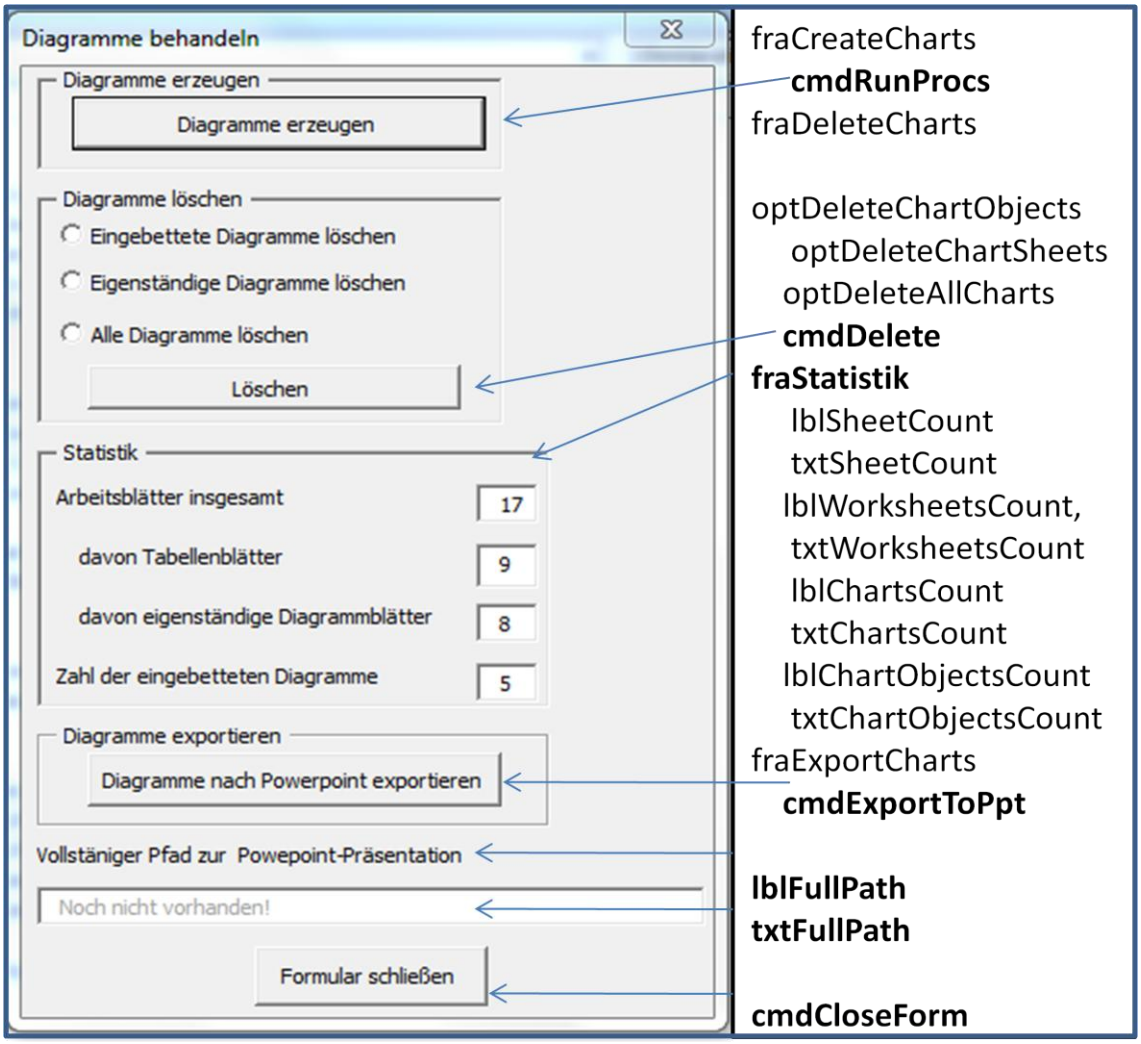

Abb. 7: Benutzerdefiniertes Formular (`UserForm`) in Excel-Arbeitsmappe

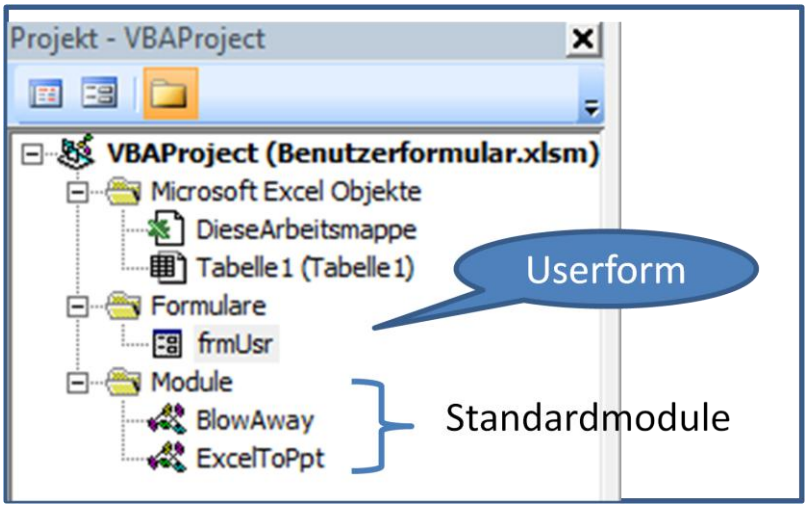

Abb. 8: `Userform` und Standardmodule in Projekt-Explorer von Excel

Das Standardmodul `BlowAway` (s. Abb. 8) enthält Prozeduren zum Löschen von Excel-Diagrammen, die nicht in eine neu zu erstellende Powerpoint-Präsentation exportiert werden sollen . Das Standardmodul `ExcelToPpt` erstellt eine neue Präsentation und exportiert Excel-Diagramme als Bilder.

4 Eingebettete Diagramme erstellen

Die folgenden Word-Tabellen und Prozeduren werden auf Knopfdruck automatisch nach Excel exportiert (vgl. Abb. 2).

4.1 Punktdiagramm

Bereits weiter oben beschrieben.

4.2 Liniendiagramm

Monat	Vorjahr	lfd. Jahr
Jan	50	60
Feb	55	62
Mrz	61	63
Apr	70	70
Mai	45	75
Jun	30	20
Jul	50	25
Aug	62	30
Sep	64	40
Okt	70	45
Nov	80	50
Dez	90	35

Tab. 2: Wertetabelle für **Liniendiagramm**

```
Sub Liniendiagramm()
   Dim objCht     As ChartObject
   Dim strTitel   As String
   Dim intRows    As Integer
   With ActiveWorkbook.Worksheets("Liniendiagramm")
      .Activate
      strTitel = .Name
      If .ChartObjects.Count > 0 Then
         Application.DisplayAlerts = False
            .ChartObjects(1).Delete
         Application.DisplayAlerts = True
      End If
      intRows = .Range("B1").End(xlDown).Row
      .Range("A2:C" & intRows).Copy
      Set objCht = .ChartObjects.Add(185, 0, 300, 250)
      objCht.Name = "Tagesumsätze"
      .ChartObjects("Tagesumsätze").Activate
      With ActiveChart
         .SeriesCollection.Paste _
               RowCol:=xlColumns, _
               SeriesLabels:=False, _
               CategoryLabels:=True, _
               Replace:=True, _
               NewSeries:=True
```

```
         Application.CutCopyMode = False
         .ChartType = xlLineMarkers
         .HasLegend = False
         .HasTitle = True
         .ChartTitle.Text = strTitel
      End With
      .Range("A1").Select
   End With
End Sub
```

4.3 Balkendiagramm

	Jan 11	Feb 11	Mrz 11	Apr 11	Mai 11	Jun 11
Umsatz	99	120	135	132	145	130

Tab. 3: Wertetabelle für **Balkendiagramm**

```
Sub Balkendiagramm()
   Const conShtName  As String = "Balkendiagramm"
   Dim objCht        As ChartObject ' Eingebettetes Diagramm
   Dim rngBereich    As Range
   Dim strTitel      As String
   With ActiveWorkbook.Worksheets(conShtName)
      .Select
      strTitel = .Name
      If .ChartObjects.Count > 0 Then
         Application.DisplayAlerts = False
            .ChartObjects(1).Delete
         Application.DisplayAlerts = True
      End If
      Set rngBereich = .Range("A1:G2")
      Set objCht = .ChartObjects.Add(0, 60, 400, 250)
   End With
   With objCht.Chart
      .SetSourceData Source:=rngBereich, PlotBy:=xlRows
      .ChartType = xlColumnClustered
      .ChartArea.Select
      .HasTitle = True
      .ChartTitle.Characters.Text = strTitel
      .Axes(xlCategory, xlPrimary).HasTitle = True
      .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Monat"
      .Axes(xlValue, xlPrimary).HasTitle = True
      .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Euro"
   End With
   Set rngBereich = Nothing
   Set objCht = Nothing
End Sub
```

4.4 Blasendiagramm

X	Y	BubbleSize	Artikel
1	5	4	A
5	7	10	B
6	10	15	C
8	20	20	D

Tab. 4: Wertetabelle für **Blasendiagramm**

```
Sub Blasendiagramm()
   ' Eingebettetes Blasendiagramm erzeugen
```

```
    Dim objCht     As ChartObject          ' Eingebettetes Diagramm
    Dim objSeries  As Series               ' Datenreihe
    Dim intSeries  As Integer              ' Nummer der Datenreihe
    With ActiveWorkbook.Worksheets("Blasendiagramm")
       .Activate ' Tabellenblatt aktivieren
       If .ChartObjects.Count > 0 Then
          Application.DisplayAlerts = False
             .ChartObjects(1).Delete
          Application.DisplayAlerts = True
       End If
       Set objCht = .ChartObjects.Add(0, 100, 400, 250)
    End With
    With objCht.Chart
       ' Diagrammtyp: Blase
       .ChartType = xlBubble
       ' Diagrammdaten zuordnen
       For intSeries = 1 To ActiveSheet.Cells.SpecialCells(xlCellTypeLastCell).Row
          Set objSeries = .SeriesCollection.NewSeries
          With objSeries
             .XValues = ActiveSheet.Range("A" & intSeries)      ' X-Wert
             .Values = ActiveSheet.Range("B" & intSeries)       ' Y-Wert
             .BubbleSizes = ActiveSheet.Range("C" & intSeries) ' Blasengröße
             .Name = ActiveSheet.Range("D" & intSeries)         ' Datenreihe
          End With
       Next intSeries
       ' Spaltenkoepfe in der 1. Tabellenzeile löschen
       .SeriesCollection(1).Delete
       ' Diagrammtyp: Blase mit 3D-Effekten
       .ChartType = xlBubble3DEffect
       ' Diagrammüberschrift
       .HasTitle = True
       With .ChartTitle
          .Text = "Blasendiagramm"
          .Font.Size = 14
       End With
       ' Diagrammfläche einfärben
       .ChartArea.Interior.ColorIndex = 15 ' grau 25%
       ' Zeichnungsfläche einfärben
       .PlotArea.Interior.ColorIndex = 36  ' hell-gelb
    End With
End Sub
```

4.5 Kursdiagramm

Datum	Eröffnungskurs	Höchstkurs	Tiefstkurs	Schlusskurs
01.03.2011	111	115	100	115
02.03.2011	105	116	95	110
03.03.2011	100	117	85	105
04.03.2011	95	118	106	100
05.03.2011	90	119	81	95

Tab. 5: Wertetabelle für **Kursdiagramm**

```
Sub Kursdiagramm()
    Dim strChtName As String
    Dim shtNew     As Worksheet   ' Neues Tabelleblatt
    Dim rngDia     As Range       ' Datentabelle
    Dim lngLastRow As Long        ' Letzte Zeile
    Dim objCht     As Chart       ' eingebettes Diagramm
    strChtName = "Kursschaubild"
```

```
    ' Prüfen, ob die Tabelle 'Diagramm' bereits existiert
    On Error Resume Next
       Application.DisplayAlerts = False
          Sheets(strChtName).Delete
       Application.DisplayAlerts = True
    On Error GoTo 0
    ' Quelldatenbereich bestimmen
    Set rngDia = Sheets("Kursdiagramm").UsedRange
    ' Neues Tabellenblatt einfügen
    Set shtNew = Worksheets.Add(After:=Sheets(Sheets.Count))
    With shtNew
       .Name = strChtName
       Set objCht = .ChartObjects.Add(Left:=20, Top:=20, _
           Width:=400, Height:=200).Chart
    End With
    With objCht
       ' Quelldatenbereich festlegen
       .SetSourceData Source:=rngDia, PlotBy:=xlColumns
       ' Diagrammtyp Open, Hight, Low, Close
       .ChartType = xlStockOHLC
       ' Sichtbaren  Titel ausgeben
       .HasTitle = True
       .ChartTitle.Characters.Text = "Kurs-Chart"
       ' Rubrikenachse beschriften
       With .Axes(xlCategory)
          .HasTitle = True
          .AxisTitle.Caption = "Datum"
       End With
       ' Werteachse beschriften
       With .Axes(xlValue)
          .HasTitle = True
          .AxisTitle.Caption = "Kurs"
       End With
       ' Diagramm bessitzt Legende
       .HasLegend = True
       ' Diagramm besitzt keine Datentabelle
       .HasDataTable = False
    End With
End Sub
```

5 Eigenständige Diagrammblätter erstellen

5.1 Säulen-, Balken- und Netzdiagramm

	Q1	Q2	Q3
A	13	12	43
B	32	54	12
C	34	34	53
D	43	23	23
E	23	43	54

Tab. 6: Wertetabelle für **SechsDiagramme**

```
Sub SechsDiagramme()
    ' Sechs Diagrammtypen mit Quartalswerten erstellen
    Const conShtName           As String = "SechsDiagramme"
    Dim rngData                As Range
    Dim intLoop                As Integer
    Dim strChartName(1 To 6)   As String
    Dim lngChartType(1 To 6)   As Long
    Dim strChtName             As String
    Dim lngChtType             As Long
    strChartName(1) = "Säulendiagramm-3D"
```

```
   strChartName(2) = "Säulendiagramm-gruppiert"
   strChartName(3) = "Säulendiagramm-gestapelt"
   strChartName(4) = "Säulendiagramm-3D"
   strChartName(5) = "Balkendiagramm-gruppiert"
   strChartName(6) = "Netzdiagramm"
   lngChartType(1) = xl3DColumn
   lngChartType(2) = xlColumnClustered
   lngChartType(3) = xlColumnStacked
   lngChartType(4) = xl3DColumn
   lngChartType(5) = xlBarClustered
   lngChartType(6) = xlRadar
   For intLoop = 1 To UBound(strChartName)
      strChtName = strChartName(intLoop)
      lngChtType = lngChartType(intLoop)
      On Error Resume Next
      Application.DisplayAlerts = False
         Charts(strChtName).Delete
      Application.DisplayAlerts = True
      Err.Clear
      On Error GoTo 0
      ' Diagramm hinzufügen
      Charts.Add After:=Worksheets(Worksheets.Count)
      With ActiveChart
         ' Diagrammtyp
         .ChartType = lngChtType
         ' Datenbereich setzen
         Set rngData = Sheets(conShtName).UsedRange
         .SetSourceData Source:=rngData, PlotBy:=xlColumns
         'Gitternetzlinien für beide Achsen setzen
         .Axes(xlCategory).HasMajorGridlines = True
         '.Axes(xlValue).HasMajorGridlines = True
         ' Legende anzeigen: oben
         .HasLegend = True
         .Legend.Position = xlTop
         ' Überschrift einsetzen
         .HasTitle = True
         .ChartTitle.Text = strChtName
         ' Werte über den Säulen anzeigen
         .ApplyDataLabels Type:=xlDataLabelsShowValue
         ' Diagramm verschieben
         .Location Where:=xlLocationAsNewSheet, Name:=strChtName
      End With
      Set rngData = Nothing
   Next intLoop
End Sub
```

5.2 Ring-und Kreisdiagramm

Kontinente	Werte
Asien	256
Europa	330
Amerika	440

Tab. 7: WerteTabelle für **RingKreisdiagramm**

```
Sub RingKreisDiagramm()
   ' Ring- bzw. Kreisdiagramm als eigenständige Diagrammblätter einfügen
   Const conShtName           As String = "RingKreisdiagramm"
   Dim rngData                As Range
   Dim intLoop                As Integer
   Dim strChartName(1 To 2)   As String
   Dim lngChartType(1 To 2)   As Long
   Dim strChtName             As String
   Dim lngChtType             As Long
   strChartName(1) = "Kreisdiagramm"
   strChartName(2) = "Ringdiagramm"
   lngChartType(1) = xlPie
   lngChartType(2) = xlDoughnut
   For intLoop = 1 To 2
      strChtName = strChartName(intLoop)
      lngChtType = lngChartType(intLoop)
      On Error Resume Next
      Application.DisplayAlerts = False
         Charts(strChtName).Delete
      Application.DisplayAlerts = True
      Err.Clear
      On Error GoTo 0
      ' Diagramm hinzufügen
      Charts.Add After:=Worksheets(Worksheets.Count)
      With ActiveChart
         ' Diagrammtyp
         .ChartType = lngChtType
         ' Datenbereich setzen
         Set rngData = Sheets(conShtName).UsedRange
         .SetSourceData Source:=rngData, PlotBy:=xlColumns
         ' Legende anzeigen: oben
         .HasLegend = True
         .Legend.Position = xlTop
         ' Überschrift einsetzen
         .HasTitle = True
         .ChartTitle.Text = strChtName
         ' Werte über den Säulen anzeigen
         .ApplyDataLabels Type:=xlDataLabelsShowValue
         ' Diagramm verschieben
         .Location Where:=xlLocationAsNewSheet, Name:=strChtName
      End With
      Set rngData = Nothing
   Next intLoop
End Sub
```

5.3 Kuchendiagramm (3D-Kreisdiagramm)

Landtagswahlen	Stimmanteile
CDU	32,6
Linke	23,7
SPD	21,5
Grüne	6,8
FDP	3,8
NDP	7,7
Sonstige	3,9

Tab. 8: Wertetabelle für **Kuchendiagramm**

```
Sub Kuchendiagramm()
   ' Kreisdiagramm als neues Diagrammblatt einfügen
   ' Konstanten deklarieren
   Const conChartName   As String = "Kuchenschaubild"
   Const conChartTitle  As String = "Stimmverteilung Sachsen-Anhalt"
   Const conDarkTeal    As Long = 49      ' dunkelblau
   Const conBlueGray    As Long = 23      ' hellblau
   Const conElevation   As Long = 30      ' Betrachtungshöhe
   Const conRotation    As Variant = 80   ' Drehung der Zeichenfläche
   ' Prüfen, ob die Tabelle 'Diagramm' bereits existiert
   On Error Resume Next
   Application.DisplayAlerts = False
      Charts(conChartName).Delete
   Application.DisplayAlerts = True
   Err.Clear
   On Error GoTo 0
   ' Diagramm hinzufühen
   Charts.Add
   With ActiveChart
      ' Diagrammtyp festlegen: Kreisdiagramm
      .ChartType = xl3DPieExploded
      .Elevation = conElevation     ' Betrachtungshöhe
      .Rotation = conRotation       ' Drehung der Zeichenfläche
      .ApplyDataLabels Type:=xlDataLabelsShowPercent
      ' Diagrammfläche füllen
      With .ChartArea.Fill
         .Visible = True
         .ForeColor.SchemeColor = conDarkTeal
         .BackColor.SchemeColor = conBlueGray
         .TwoColorGradient msoGradientHorizontal, 1
      End With
      ' Datenherkunft bestimmen
      .SetSourceData _
         Source:=Sheets("Kuchendiagramm").UsedRange, _
         PlotBy:=xlColumns
      ' Datenbeschriftung
      With .SeriesCollection(1)
         .HasDataLabels = True
         .ApplyDataLabels Type:=xlDataLabelsShowValue
         With .DataLabels
            .ShowLegendKey = True
            .Type = xlValue
            With .Font
               .Size = 12
               .ColorIndex = 2
            End With
         End With
      End With
      ' Kreisdiagramm als neues Diagrammblatt einfügen
      .Location Where:=xlLocationAsNewSheet, Name:=conChartName
      ' Überschrift setzen
      .HasTitle = True
      With .ChartTitle
         .Characters.Text = conChartTitle
         With .Font
            .Size = 24              ' Schriftgrad
            .ColorIndex = 2         ' weiß
         End With
      End With
```

```
      ' Legende setzen
      .HasLegend = True
      With .Legend
         .Position = xlLegendPositionRight
         .Shadow = True
         .AutoScaleFont = False
         .Font.Size = 14            ' Schriftgrad
      End With
   End With
End Sub
```

6. Diagrammautomatisierung

Für die in Tab. 9 enthaltenen Daten sollen zwei eigenständige Diagrammblätter in Excel erstellt werden, eins für die Region „Ost“ und ein weiteres für die Region „*West*“. In diesen beiden Diagrammblättern sollen jeweils so viele Balkendiagramme enthalten sein, wie Personen in der jeweiligen Region vorkommen, also 2 in „*Ost*“ und 3 in „*West*“.

Das Excel-Tabellenblatt, das die Werte aus Tab. 9 aufnimmt, muss „**Data**“ heißen. Wird dafür ein anderer Name gewählt, muss die Konstante **`conShtDaten`** entsprechend geändert werden. Benötigt wird außerdem ein eigenständiges Excel-Diagrammblatt, das nach dem Muster in Abb. 9 zu erstellen ist. Dieses manuell erstellte Diagrammblatt muss einmalig erstellt und „**Vorlage**“ genannt werden. Wird dafür ein anderer Name gewählt, muss die Konstante `conShtChart` entsprechend geändert werden.

Region	Person	Kalenderwoche	Offen	in Bearbeitung	geschlossen
Ost	Heinz	11	37	41	68
Ost	Heinz	12	35	52	72
Ost	Heinz	13	36	51	72
Ost	Heinz	14	36	49	63
Ost	Peter	11	57	71	68
Ost	Peter	12	42	62	72
Ost	Peter	13	45	55	72
Ost	Peter	14	26	51	63
West	Walter	11	76	81	101
West	Walter	12	76	84	100
West	Walter	13	76	88	95
West	Walter	14	78	84	96
West	Franz	11	57	65	92
West	Franz	12	59	81	94
West	Franz	13	62	87	95
West	Franz	14	82	99	107
West	Xaver	11	71	110	331
West	Xaver	12	101	157	229
West	Xaver	13	96	151	224
West	Xaver	14	110	148	224

Tab. 9: Wertetabelle für die Diagrammautomatisierung
(zu speichern im Tabellenblatt „**Data**“

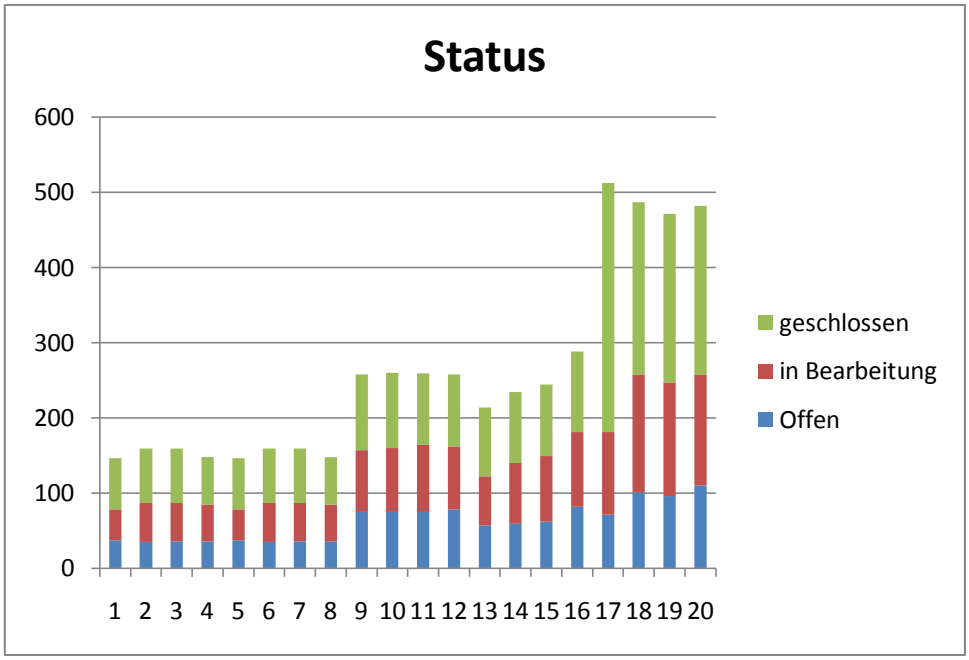

Abb. 9: Eigenständiges Diagrammblatt „**Vorlage**“
(basierend auf Bereich '**Data**'!$D$1:$F$21)

```
' Automatisiertes Erstellen von Excel-Diagrammen
' Konstanten deklarieren
Const conShtChart = "Vorlage" ' Diagrammblatt mit Vorlage
Const conShtDaten = "Data"    ' Tabellenblat mit Daten
Const conShtOst = "Ost"       ' Ziel-Diagrammblatt
Const conShtWest = "West"     ' Zieldiagrammblatt
Const conRowStart = 2
Const conDataInterval = 4
Const conColNbr = 1
Const conColNames = 2
Sub CreateCharts()
   Dim strShtName As String                  ' Name des Tabellebblatts {Ost, Wet}
   Dim lngLoop As Long                       ' Schleifenzähler

   ThisWorkbook.Activate                     ' Arbeitsmappe aktivieren
   Application.ScreenUpdating = False        ' Bildschirmaktualisierung ausschalten
   ' Diagramme entfernen, falls vorhanden
   On Error Resume Next
   Sheets(conShtOst).ChartObjects.Delete     ' Alle Diagramme in Blatt "Ost" entfernen
   Sheets(conShtWest).ChartObjects.Delete    ' Alle Diagramme in Blatt "West" entfernen
   On Error GoTo 0
   With Sheets(conShtDaten)                  ' Datentabelle "Data" auswerten
      For lngLoop = conRowStart To .Cells(.Rows.Count, conColNames).End(xlUp).Row _
            Step conDataInterval
         ' If .Cells(lngLoop, conColNbr) Like Left(conShtOst, 1) Then
         If .Cells(lngLoop, conColNbr) = conShtOst Then
            strShtName = conShtOst           ' Tabellenbatt "Ost"
         Else
            strShtName = conShtWest          ' Tabellenbaltt "West"
         End If
         ' Neues Diagramm einfügen
         Call MakeNewChart(strShtName, .Cells(lngLoop, conColNames))
      Next lngLoop
   End With
   Application.ScreenUpdating = True         'Bildschirmaktualisierung einschalten
End Sub
```

```
Private Sub MakeNewChart(ByRef strShtName As String, ByRef rngName As Range)
   Dim dblChartTop   As Double               ' Diagramm-Position, oben
   Dim lngLoop       As Long                 ' Schleifenzähler
   With Sheets(strShtName).ChartObjects
      If .Count = 0 Then                     ' Wenn Zähler für Diagramm = 0, ...
         dblChartTop = 0                     ' dann Diagramm Position, oben = 0,
      Else                                   ' sonst, ...
         With .Item(.Count).BottomRightCell  ' Diagramm-Positionen ermitteln
            dblChartTop = .Top + .Height     ' Diagramm-Position, oben=oben + Höhe
         End With
      End If
      ' Diagramm-Vorlage zum Ziel-Tabellenblatt kopieren
      Sheets(conShtChart).ChartObjects(1).Copy
      ' Diagramm-Vorage einfügen
      .Parent.Paste
      With .Item(.Count).Chart               ' Mit dem neuen Diagramm-Objekt ...
         .Parent.Top = dblChartTop           ' Position, oben
         .Parent.Left = 0                    ' Position, links
         ' Überschrift setzen
         .ChartTitle.Characters.Text = rngName.Text
         For lngLoop = 1 To 3                ' Datenreihen setzen
            .SeriesCollection(lngLoop).Values = _
               rngName.Offset(0, lngLoop + 1).Resize(conDataInterval, 1)
         Next
         ' Diagramm-X-Achse (Kalenderwochen) setzen
         .SeriesCollection(1).XValues = _
            rngName.Offset(ColumnOffset:=1).Resize(conDataInterval, 1)
      End With
   End With
End Sub
```

7.Übungen

Bitte studieren Sie die Prozedur `AddNewChartObject`:

```
Sub AddNewChartObject()
   ' Objektvariable für eingebettetes Diagramm deklarieren
   Dim objCht As ChartObject
   ' Eingebettetes Diagramm links oben im aktiven Tabellenblatt hinzufügen
   Set objCht = ActiveSheet.ChartObjects.Add _
      (Left:=0, Top:=0, Width:=375, Height:=225)
   With objCht
      With .Chart
         ' Diagrammtyp festlegen
         .ChartType = xlXYScatterLines
         ' Überschrift bestimmen
         .HasTitle = True
         .ChartTitle.Text = "Punkt(XY)-Diagramm"
         ' Datenreihe hinzufügen
         With .SeriesCollection.NewSeries
            .Name = "Meine neue Datenreihe"  ' Name der neuen Datenreihe
            .Values = Array(19, 15, 12, 10)  ' Y-Werte der neuen Datenreihe
            .XValues = Array(1, 2, 3, 4)     ' X-Werte der neuen Datenreihe
         End With
      End With
   End With
   ' Objektvariable freigeben
   Set objCht = Nothing
End Sub
```

1. Was bewirkt die Prozedur?
2. Woher bezieht die Prozedur ihre Daten?
3. Was passiert, wenn die Codezeile `.HasTitle =True` weggelassen wird oder wenn `.HasTitle = False` gesetzt wird?
4. Eignen sich die benutzen Daten zur Erzeugung eines (**a**) Kreisdiagramms bzw. (**b**) Kursdiagramms vom Typ *Höchst-Tiefst-Geschlossen* (`xlStockHLC`) ?

8. Lösungen

1. Sie erstellt ein Punkt(XY)-Diagramm im aktiven Tabellenblatt.
2. Aus Datenfeldern, also *nicht* aus Tabellenwerten.
3. Die Prozedur erzeugt in beiden Fällen einen Laufzeitfehler.
4. (**a**) Ja (**b**) Nein