

6 Grafiken: die *InlineShape*- und *Shape*-Objekte

Da Word ein Textverarbeitungsprogramm ist, bekundet es manchmal etwas Mühe mit Grafiken. Vor allem ist die VBA-Schnittstelle dafür weder vollständig noch besonders intuitiv. Ziel dieses Abschnitts ist, den allgemeinen Umgang mit Grafiken zu erklären, sowie einige Besonderheiten hervorzuheben.

Hinweis Beginn



2007

Für Office 2007 wurde die Grafik-Funktionalität neu entwickelt, um den erhöhten Ansprüchen der heutigen Welt zu entsprechen. Die neuen Möglichkeiten wurden voll in PowerPoint, aber auch in Excel integriert. Da in Word die Zusammenarbeit mit Grafiken komplexer ist, wurde hier die neue Funktionalität nur teilweise eingeführt. Gewisse Funktionalität, die der Anwender aus einer früheren Version erwartet, steht ihm nicht mehr zur Verfügung, während sie im Objektmodell noch vorhanden ist. Andersherum kann der Programmierer auf einige der neuen Eigenschaften zugreifen, die dem Benutzer noch nicht zur Verfügung stehen. Eine verwirrende Situation, die sich hoffentlich bis zur nächsten Version klärt... Wir werden in diesem Kapitel aus Platzgründen diesen Eigenarten nicht nachgehen, sondern uns hauptsächlich auf den Umgang mit dem Objektmodell befassen.

Hinweis Ende

InlineShape vs. *Shape*

Es gibt zwei verschiedene Methoden, eine Grafik in ein Dokument einzubetten: in der Zeile mit dem Text oder mit einer Textflussformatierung. Steht eine Grafik in der Zeile mit dem Text, ist sie Teil der *InlineShapes*-Auflistung, Word behandelt sie grundsätzlich wie ein Textzeichen, und die VBA-Befehle gehören dem Word-Objektmodell an. Wurde sie jedoch mit Textfluss formatiert, steht sie in der Zeichnungsebene des Dokuments, ist Teil der *Shapes*-Auflistung, und die passenden Befehle befinden sich im Office-Objektmodell, da Zeichnungsobjekte Office-übergreifend sind.

Jede Word-Version fügt Grafiken standardmäßig anders ein. Ab Word 2002 darf der Benutzer sogar die Methode selber bestimmen. (Bis Word 2007 über den Menübefehl *Extras/Optionen*, Registerkarte *Bearbeiten*, Option *Bild einfügen als*. Ab Word 2007 befindet sich die Option in der Registerkarte *Erweitert*, im Abschnitt *Ausschneiden, Kopieren und Einfügen* der Word-Optionen.) Folglich muss der Code, wollen Sie alle Grafiken eines Dokuments bearbeiten, beide Auflistungen berücksichtigen.

Grafiken einfügen

Bis und mit Word 2003 sind Einsichten in das Objektmodell für beide Auflistungen durch den Makrorekorder möglich. Es gibt aber einige Tricks, um eine Grafik nach deren Einfügung zu markieren, um sie weiter zu manipulieren. In Listing 6.1 befinden sich zwei in Word 2003 aufgezeichnete Prozeduren für das Einfügen und die Größenänderung von Grafiken. Bei der ersten wurde die Grafik in die Zeile mit dem Text eingefügt, bei der zweiten mit Textflussformatierung.



Word 2007 zeichnet das Einfügen der Grafik korrekt auf. Im Gegensatz zu früheren Versionen sind die eingefügten Objekte markiert. Das nutzt uns herzlich wenig, da über die Multifunktionsleiste vorgenommene Formatierungen werden vom Makrorekorder nicht aufgezeichnet. Dies liegt wohl an der oben beschriebenen, teilweisen Integration der neuen Grafik-Funktionalität. Wem keine ältere Version von Word zur Verfügung, um ungewisse Punkte im Objektmodell nachzuforschen, wird es hart haben.

Während in früheren Word-Versionen ein InlineShape von einem Shape durch Farbe der Anfasser schnell erkennbar war, ist dies in Word 2007 nur im Kompatibilitäts-Format möglich. Grafiken, die in ein Word 2007 Dokument (*docx*) eingefügt wurden, sind visuell nur durch Prüfen der Textumbruch-Formatierung erkennbar.

Aus Datei mit AddPicture

Beim Vergleich der zwei Routinen fällt auf, dass sowohl die InlineShapes- wie die Shapes-Auflistungen eine AddPicture-Methode besitzen, um eine Grafikdatei einzufügen. Die beiden haben aber zum Teil unterschiedliche Argumente (zweite Codezeile).

In der nächsten Codezeile sehen Sie, wie Word 2003 und früher ein InlineShape bzw. ein Shape markiert wird. Im ersten Fall steht nach der Einfügung die Einfügemarke links neben der Grafik. Halten Sie also die (Umschalt)-Taste fest und drücken Sie dann (Pfeil rechts). Wurde ein Shape eingefügt, müssen Sie es mit der Maus anklicken.

Letztlich machen wir darauf aufmerksam, dass der Makrorekorder in Word 2003 und früher nicht nur die geänderten Optionen im Dialogfeld *Grafik formatieren* aufzeichnet, sondern mehrere Optionen aus allen Registerkarten. Sie müssen also den Code anpassen, so dass er nur die gewünschte Änderung vornimmt, wenn er später ausgeführt wird. Auffallend ist, dass gerade die Größenänderung für das Shape nicht aufgezeichnet wurde. Die Codezeilen

```
Selection.InlineShapes(1).LockAspectRatio = msoTrue
Selection.InlineShapes(1).Height = 113.4
Selection.InlineShapes(1).Width = 113.4
```

sind für die Größenänderung des InlineShapes zuständig.

Listing 6.1: Dieser aufgezeichnete Code gibt Aufschluss über die InlineShape- bzw. Shape-Objekte

```
Sub InlineShapeImportierenUndBearbeiten()

    Selection.InlineShapes.AddPicture FileName:="C:\WINDOWS\Seifenblase.bmp", _
        LinkToFile:=False, SaveWithDocument:=True
    Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
```

```

Selection.InlineShapes(1).Fill.Visible = msoFalse
Selection.InlineShapes(1).Fill.Solid
Selection.InlineShapes(1).Fill.Transparency = 0#
Selection.InlineShapes(1).Line.Weight = 0.75
Selection.InlineShapes(1).Line.Transparency = 0#
Selection.InlineShapes(1).Line.Visible = msoFalse
Selection.InlineShapes(1).LockAspectRatio = msoTrue
Selection.InlineShapes(1).Height = 113.4
Selection.InlineShapes(1).Width = 113.4
Selection.InlineShapes(1).PictureFormat.Brightness = 0.5
Selection.InlineShapes(1).PictureFormat.Contrast = 0.5
Selection.InlineShapes(1).PictureFormat.ColorType = msoPictureAutomatic
Selection.InlineShapes(1).PictureFormat.CropLeft = 0#
Selection.InlineShapes(1).PictureFormat.CropRight = 0#
Selection.InlineShapes(1).PictureFormat.CropTop = 0#
Selection.InlineShapes(1).PictureFormat.CropBottom = 0#
End Sub

```

```

Sub ShapeImportierenUndBearbeiten()
ActiveDocument.Shapes.AddPicture(Anchor:=Selection.Range, FileName:= _
"C:\WINDOWS\Zapotek.bmp", LinkToFile:=False, SaveWithDocument:=True). _
WrapFormat.Type = wdWrapSquare
ActiveDocument.Shapes("Picture 2").Select
Selection.ShapeRange.Fill.Visible = msoFalse
Selection.ShapeRange.Fill.Solid
Selection.ShapeRange.Fill.Transparency = 0#
Selection.ShapeRange.Line.Weight = 0.75
Selection.ShapeRange.Line.DashStyle = msoLineSolid
Selection.ShapeRange.Line.Style = msoLineSingle
Selection.ShapeRange.Line.Transparency = 0#
Selection.ShapeRange.Line.Visible = msoFalse
Selection.ShapeRange.LockAspectRatio = msoTrue
Selection.ShapeRange.Rotation = 0#
Selection.ShapeRange.PictureFormat.Brightness = 0.5
Selection.ShapeRange.PictureFormat.Contrast = 0.5
Selection.ShapeRange.PictureFormat.ColorType = msoPictureAutomatic
Selection.ShapeRange.PictureFormat.CropLeft = 0#
Selection.ShapeRange.PictureFormat.CropRight = 0#
Selection.ShapeRange.PictureFormat.CropTop = 0#
Selection.ShapeRange.PictureFormat.CropBottom = 0#
Selection.ShapeRange.Left = 70.85
Selection.ShapeRange.Top = 198.1
Selection.ShapeRange.RelativeHorizontalPosition = _
wdRelativeHorizontalPositionColumn
Selection.ShapeRange.RelativeVerticalPosition = _
wdRelativeVerticalPositionParagraph
Selection.ShapeRange.Left = CentimetersToPoints(0)
Selection.ShapeRange.Top = CentimetersToPoints(0)
Selection.ShapeRange.LockAnchor = False
Selection.ShapeRange.LayoutInCell = True
Selection.ShapeRange.WrapFormat.AllowOverlap = True
Selection.ShapeRange.WrapFormat.Side = wdWrapBoth
Selection.ShapeRange.WrapFormat.DistanceTop = CentimetersToPoints(0)

```

```

Selection.ShapeRange.WrapFormat.DistanceBottom = CentimetersToPoints(0)
Selection.ShapeRange.WrapFormat.DistanceLeft = CentimetersToPoints(0.32)
Selection.ShapeRange.WrapFormat.DistanceRight = CentimetersToPoints(0.32)
Selection.ShapeRange.WrapFormat.Type = wdWrapSquare
End Sub

```

Wie in Kapitel 5 erwähnt, soll Code mit dem Selection-Objekt möglichst vermieden werden. Listing 6.2 bzw. Listing 6.3 zeigt den bereinigten Code. Da die AddPicture-Methode immer ein Objekt zurückgibt, ist es einfach, eine entsprechende Objekt-Variable zu deklarieren und ihr das Grafikobjekt gleich beim Importieren zuzuweisen.

Beachten Sie, wie die Positionierung des Shapes nochmals nach der Anpassung der Größe durchgeführt wird. In diesem Fall wird die Grafik relativ zum verankernden Absatz positioniert. (Entspricht der Einstellung *Objekt mit Text verschieben* der Registerkarte *Bildposition*, die in Word 2003 über die Schaltfläche *Weitere* der Registerkarte *Layout* im Dialogfeld *Grafik formatieren* erreicht wird (Abbildung 6.2) und in Word 2007 über den Menübefehl *Weitere Layout-Optionen* der Schaltflächen *Position* und *Textumbruch* der Registerkarte *Bildtools/Format*.)

Listing 6.2: Der bearbeitete Code führt nur die gewünschten Handlungen aus: Grafik einfügen und die Größe anpassen

```

Sub InlineShapeImportierenUndBearbeiten2()
    Dim ils As Word.InlineShape

    Set ils = Selection.InlineShapes.AddPicture(FileName:="C:\Veispiele\Soap Bubbles.bmp",
    -
        LinkToFile:=False, SaveWithDocument:=True)
    ils.LockAspectRatio = msoTrue
    ils.Height = 113.4
    ils.Width = 113.4
End Sub

Sub ShapeImportierenUndBearbeiten2()
    Dim shp As Word.Shape

    Set shp = ActiveDocument.Shapes.AddPicture(Anchor:=Selection.Range, FileName:= _
        "C:\Beispiele\Zapotec.bmp", LinkToFile:=False, SaveWithDocument:=True)
    shp.WrapFormat.Type = wdWrapSquare
    shp.LockAspectRatio = msoTrue
    shp.Height = 113.4
    shp.RelativeHorizontalPosition = _
        wdRelativeHorizontalPositionColumn
    shp.RelativeVerticalPosition = _
        wdRelativeVerticalPositionParagraph
    shp.Left = CentimetersToPoints(0)
    shp.Top = CentimetersToPoints(0)
End Sub

```

Listing 6.3 (.NET): Die C#-Version

```

private void InlineShapeImportierenUndBearbeiten_CS()
{
    wd.Selection sel = wdApp.Selection;
    object objRange = (object) sel.Range;
    wd.InlineShape ils = sel.InlineShapes.AddPicture(@"C:\Beispiele\Soap Bubbles.bmp",

```

```

        ref objFalse, ref objTrue, ref objRange);
    ils.LockAspectRatio = Microsoft.Office.Core.MsoTriState.msoTrue;
    ils.Height = 113.4f;
    ils.Width = 113.4f;
}

private void ShapeImportierenUndBearbeiten_CS(wd.Document doc)
{
    wd.Selection sel = wdApp.Selection;
    object objRange = (object) sel.Range;
    wd.Shape shp = doc.Shapes.AddPicture(@"C:\Beispiele\Zapotec.bmp", ref objFalse,
        ref objTrue, ref objMissing, ref objMissing, ref objMissing,
        ref objRange);
    shp.WrapFormat.Type = wd.WrapType.wdWrapSquare;
    shp.LockAspectRatio = Microsoft.Office.Core.MsoTriState.msoTrue;
    shp.Height = 113.4f;
    shp.RelativeHorizontalPosition =
        wd.RelativeHorizontalPosition.wdRelativeHorizontalPositionColumn;
    shp.RelativeVerticalPosition =
        wd.RelativeVerticalPosition.wdRelativeVerticalPositionParagraph;
    shp.Left = wdApp.CentimetersToPoints(0);
    shp.Top = wdApp.CentimetersToPoints(0);
}

```

CD-ROM Beginn

Die Beispieldatei *Bsp06_01_Graf.doc* finden Sie auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

CD-ROM Ende

Über die Zwischenablage

Grafiken werden nicht nur aus Dateien importiert, sie werden auch aus der Zwischenablage eingefügt. Wie beim aufgezeichneten Code besteht auch hier das Problem: Wie wird das eingefügte Objekt angesprochen, so dass mit der Aufzeichnung weitergefahren werden kann? Die Methoden `Paste` und `PasteSpecial` geben, im Gegensatz zu `AddPicture`, kein Objekt zurück.

Prozeduren für diese Aufgabe finden Sie in Listing 6.4. Im Fall eines `InlineShape`-Objekts steht nach dem Einfügen aus der Zwischenablage die Einfügemarke rechts neben der Grafik. Dies funktioniert also ähnlich wie bei der Aufzeichnung beim Einfügen aus einer Datei: die Markierung wird mit der Tastenfolge **(Umschalt)+(Pfeil links)** um ein Zeichen – diesmal nach links – erweitert.

Nach Ausführung der Methode `Paste` ist ein Shape markiert, also kein Problem. Wollen Sie aber `PasteSpecial` (Menübefehl *Bearbeiten/Inhalte einfügen*; ab Word 2007 im Menü der Schaltfläche *Einfügen* in der Registerkarte *Start*) einsetzen, um eine Verknüpfung herzustellen oder den Datentyp festzulegen, braucht es ein »Workaround«, da die Markierung im Text bleibt. Der Code muss das eingefügte Objekt irgendwie erkennen können. Die Frage ist nur, wie.

Im Abschnitt »Die Positionierung von Shape-Objekten« in diesem Kapitel wird beschrieben, wie jedes grafische Objekt mit einem bestimmten Absatz verbunden (verankert) sein muss. Folglich müsste das `Paragraph`-Objekt der aktuellen Markierung oder des Zielbereichs (Range) das eingefügte Objekt liefern. Problematisch wird es, jedoch, wenn mehrere Objekte im gleichen Absatz verankert

sind.

Im Beispielcode der Prozedur *InhalteEingefuegtesShapeErfassen* wurde das Problem mit Hilfe der `AlternativeText`-Eigenschaft eines `Shape`-Objekts gelöst. Standardmäßig wird die Eigenschaft neu eingefügter Grafiken nicht belegt. Bevor die Grafik eingefügt wird, wird diese Eigenschaft für alle im Absatz verankerten Objekte mit einem eindeutigen Wert belegt, wenn sie noch keine Angabe enthält. Nach dem Einfügen wird nochmals durch den `ShapeRange` geschleift, um das einzige Objekt ohne einen `AlternativeText`-Eintrag zu finden – dies ist die eingefügte Grafik – und es einer Objekt-Variablen zugewiesen. Danach werden die vom Code generierten `AlternativeText`-Angaben entfernt.

Hinweis Beginn

Ein Fehler wird ausgelöst, falls sich bei der Ausführung der Prozedur *InhalteEingefuegtesShapeErfassen* keine Grafik, sondern ein anderer Dateityp in der Zwischenablage befindet. Die Prozedur fängt den Fehler auf und blendet eine entsprechende Meldung ein. Das Office-Objektmodell bietet keine Schnittstelle an, um den Datentyp der Zwischenablage zu ermitteln.

Hinweis Ende

Listing 6.4: Prozeduren, um das über die Zwischenablage eingefügte Objekt im Code zu erfassen

```
Sub EingefuegtesInlineShapeErfassen()  
    Dim rng As Word.Range  
    Dim ils As Word.InlineShape  
  
    Set rng = Selection.Range  
    'rng.Paste  
    rng.PasteSpecial Placement:=wdInLine, DataType:=wdPasteMetafilePicture  
    rng.MoveStart wdCharacter, -1  
    Set ils = rng.InlineShapes(1)  
End Sub  
  
Sub EingefuegtesShapeErfassen ()  
    Dim rng As Word.Range  
    shp As Word.Shape  
  
    Set rng = Selection.Range  
    rng.PasteSpecial Placement:=wdFloatOverText, DataType:=wdPasteMetaPicture  
    Set shp = rng.ShapeRange(1)  
    Debug.Print shp.Name  
End Sub  
  
Sub InhalteEingefuegtesShapeErfassen()  
    Dim rng As Word.Range  
    Dim shpNeu As Word.Shape  
    Dim shp As Word.Shape  
    Dim counter As Long  
  
    On Error GoTo Fehlerbehandlung  
    Set rng = Selection.Range  
    counter = 0  
    'Es ist nur möglich, einen ShapeRange durchzuschleifen,  
    'wenn mindestens ein Shape vorhanden ist, also testen
```

```

If rng.Paragraphs(1).Range.ShapeRange.Count > 0 Then
    'Sicherstellen, dass die Eigenschaft AlternativeText aller
    'im Absatz verankerten Shapes belegt ist
    'mit eindeutigem Inhalt (Datum + Zeit des Kontrollgangs)
    For Each shp In rng.Paragraphs(1).Range.ShapeRange
        If Len(shp.AlternativeText) = 0 Then
            shp.AlternativeText = "Autogeneriert" & Format(Now, "yyymmdd_hhnnss_") & counter
            counter = counter + 1
        End If
    Next
End If

rng.PasteSpecial Placement:=wdFloatOverText, DataType:=wdPasteMetafilePicture
'Die neue Grafik finden
For Each shp In rng.Paragraphs(1).Range.ShapeRange
    If Len(shp.AlternativeText) = 0 Then
        Set shpNeu = shp
    End If
Next
'Da AlternativeText bei eingeblendeten nicht druckbaren Zeichen
'über die Grafik angezeigt wird, alle "Autogeneriert" löschen
For Each shp In rng.Paragraphs(1).Range.ShapeRange
    If Left(shp.AlternativeText, 13) = "Autogeneriert" Then
        shp.AlternativeText = ""
    End If
Next
Debug.Print shpNeu.Name

Fertigstellen:
Exit Sub

Fehlerbehandlung:
Select Case Err.Number
    Case 5342 'Unbekannter Datentyp beim Einfügen
        MsgBox "Keine Grafik in der Zwischenablage vorhanden", _
            vbOKOnly + vbCritical, "Kapitel 6 - Grafiken"
    Case Else
        MsgBox "Es gab den folgenden unerwarteten Fehler in der Prozedur " & _
            & "InhalteEingefuegtesShapeErfassen:" & vbCrLf & vbCrLf & _
            Err.Number & vbCrLf & vbCrLf & Err.Description, vbCritical + vbOKOnly, _
            "Kapitel 6 - Grafiken"
End Select
End Sub

```

Dialogfeld Grafik einfügen

Oft müssen Anwendungen interaktiv mit dem Benutzer arbeiten. Wir wollen es ihm beispielsweise ermöglichen, eine Grafik auszuwählen, aber der Code soll diese einfügen und weiter bearbeiten. Am einfachsten geht es, wenn wir das Word-eigene Dialogfeld *Grafik einfügen* im Code verwenden können. Ein Beispiel hierfür finden Sie in Kapitel IV des Bonusteils auf der CD.

Es ist auch möglich, ab Word 2002, das `FileDialog`-Objekt einzusetzen, falls Sie über die Benutzerschnittstelle mehr Kontrolle brauchen (beispielsweise nur bestimmte Dateiendungen anbieten wollen). Das `FileDialog`-Objekt wird im Kapitel 15 vorgestellt.

Die Beispieldatei *Bsp06_01_Graf.doc* finden Sie auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

Verknüpfungen erstellen und verwalten

Jede der diversen Methoden, um Grafiken einzufügen, erlaubt eine Verknüpfung mit der Ursprungsdatei. Wird eine Grafik mit der `PasteSpecial`-Methode eingefügt, bedient sich der Entwickler der `Link`-Eigenschaft, um die Verknüpfung herzustellen.

Wie aus Listing 6.2 hervorgeht, hat die `AddPicture`-Methode eine optionale `LinkToFile`-Eigenschaft, die für eine Dateiverknüpfung sorgt. Zudem hat die Methode die optionale Eigenschaft `SaveInDocument`. Diese wird nur berücksichtigt, wenn `LinkToFile` auf `True` gesetzt wird. Ist `SaveInDocument` ebenfalls `True`, wird die Grafik in der Dokumentstruktur gespeichert. Sonst enthält das Dokument nur die Verknüpfungsinformationen, was in einer kleineren Dateigröße resultiert. Standardmäßig werden beide auf `False` gesetzt.

Verknüpfungen eingebetteter Grafiken werden über die `LinkFormat`-Eigenschaft gesteuert. Die wichtigsten Eigenschaften und Methoden sind in Tabelle 6.1 aufgelistet.

Tabelle 6.1: Eigenschaften für die Verwaltung von Verknüpfungen

LinkFormat-Eigenschaft oder -Methode	Datentyp	Beschreibung
BreakLink		Löst die Verknüpfung auf.
SavePictureWithDocument	Boolean	Wenn False, wird nur die Verknüpfung zur Grafik im Dokument gespeichert. (*)
SourceFullName	Zeichenkette	Geben die vollen Pfadangaben, den Dateinamen sowie den Dateipfad zurück.
SourceName		
SourcePath		
Update		Aktualisiert die Grafik. (*)
<i>Gilt nur für InlineShapes</i>		
Locked	Boolean	Wenn True, kann die Grafik nicht aktualisiert werden.

(*) Fügt ab Word 2002 automatisch den Schalter `* Mergeformat` hinzu, auch wenn er bereits vorhanden ist, was zu Formatierungsverlusten führen kann.

Kasten Beginn

SourceFullName-Eigenschaft in Word 97 und Word 2000

Zwar stellt das Word-Objektmodell die `LinkFormat.SourceFullName`-Eigenschaft für `InlineShape`- sowie `Shape`-Objekte zur Verfügung. Wird sie jedoch benutzt, um die Pfadangabe einer verknüpften Grafik zu ändern, stürzen frühere Word-Versionen (z.B. 97 und 2000) ab:

```
ActiveDocument.Shapes("MeinBild").LinkFormat.SourceFullName = "C:\Grafiken\MeinBild.bmp"
```

Das Problem wurde in Word 2002 behoben, was aber nicht hilft, wenn Sie frühere Versionen automatisieren müssen. Ein gutes »Workaround« für `Shape`-Objekte in diesen Versionen gibt es nicht. In diesem Fall muss ein `Shape` in ein `InlineShape`

konvertiert werden, um auf den Feldcode zuzugreifen:

```
Set ils = ActiveDocument.Shapes("MeinBild").ConvertToInlineShape
```

Wir raten davon ab, eine in ein `InlineShape` konvertierte Grafik mit der `ConvertToShape`-Methode zurück in ein `Shape` zu wandeln. Es geht, aber eine nochmalige Konvertierung in ein `InlineShape` beschädigt den Feldcode, was einen Dokumentabsturz verursachen kann. Stattdessen sollen Sie den Textfluss um die Grafik anders hin bekommen, indem Sie die Grafik als ein `InlineShape` in einen Positionsrahmen oder in ein Textfeld einfügen (siehe den Abschnitt »Textfluss-Formatierung« in diesem Kapitel).

Kasten Ende

Im Hintergrund verwaltet Word verknüpfte Grafiken über die Feldfunktion *IncludePicture* (in Abbildung 6.1 abgebildet). Neben dem Feldnamen besteht der Feldcode aus der Pfadangabe zur Ursprungsdatei; der Pfad darf relativ zum Speicherort des Dokuments sein, oder absolut. Zusätzlich stehen der Feldfunktion zwei wichtige Schalter zur Verfügung:

- `\d`: Weist Word an, dass die Grafikdaten nicht im Dokument gespeichert werden sollen (gleich `SavePictureInDocument = False`.)
- `* Mergeformat`: Behält bei der Aktualisierung in Word vorgenommene Formatierungen bei.

Durch Drücken der Tastenkombination `(Alt)+(F9)` werden alle Feldcodes im *Dokumenttext* ein- und wieder ausgeblendet. Sie werden also nur diejenige für `InlineShape`-Objekte sehen, da `Shapes` außerhalb des Textes, in der Zeichnungsebene stehen.

Abbildung 6.1: In Wirklichkeit werden in Word verknüpfte Grafiken durch *IncludePicture*-Feldcodes verwaltet

```
{ INCLUDEPICTURE "C:\\Windows\\Feder.bmp" \* MERGEFORMAT }
```



Wichtig Beginn

Die Backslashes in Pfadangaben einer Feldfunktion müssen immer verdoppelt werden. Dies weil ein einzelnes Backslash-Zeichen in einem Feldcode einen Schalter identifiziert.

Relative Pfadangaben werden genauso wie in DOS geschrieben. Befindet sich die verknüpfte Datei im gleichen Ordner mit dem Dokument, braucht es nur den Dateinamen. Um auf die nächst höhere Ordnersebene zu weisen, wird `..\` vor den Pfadnamen gesetzt; mehrere `..\` dürfen aufeinander folgen. Bitte beachten Sie, dass sich in diesem Fall Word auf das aktuelle Verzeichnis für *die Anwendung* bezieht

(meistens dort, wo das letzte Dokument geöffnet wurde), und nicht auf den Ordner, worin sich das Dokument mit der Verknüpfung befindet. Diese könnten die gleiche sein, es ist aber nicht zwingend. Relative Pfadangaben sind also mit einem gewissen Risiko behaftet.

Wichtig Ende

Bestimmt fragen Sie sich, warum wir hier die Feldfunktion behandeln. Die Antworten darauf sind folgende:

- Nur so ist es möglich, den * *Mergeformat*-Schalter im Dokument vorhandener Grafiken gezielt hinzuzufügen oder zu entfernen.
- Wegen des Problems ab Word 2002 mit dem automatischen Hinzufügen des Schalters * *Mergeformat* bei Verwendung der Eigenschaften und Methoden des Objektmodells ist es vorteilhaft, auch die Änderungen der Pfadangabe, das Hinzufügen oder Entfernen des Schalters \d, die Sperrung sowie die Aktualisierung der Verknüpfung über die Feldcodes vorzunehmen.
- Der Verknüpfungspfad kann direkt im Feldcode bearbeitet werden, um in Word 2000 das Problem mit `SourceFullName` zu umgehen.

Das Beispiel in Listing 6.5 zeigt, wie Sie einen Feldcode benutzen, um eine verknüpfte Grafik einzufügen und deren Größe zu ändern. Dann wird die Prozedur *GrafikFeldcodeBearbeiten* aufgerufen, um den Feldcode zu ändern. Es ist insbesondere zu beachten, dass als Folge der Aktualisierung des Feldcodes das damit verbundene VBA-Objekt gelöscht wird. Da es im Code weiterhin gebraucht wird, muss das Objekt wieder hergestellt werden. Am Ende der Prozedur *GrafikAlsFeldEinfuegen* wird die Verknüpfung aufgelöst, so dass nur eine gewöhnliche eingebettete Grafik im Dokument zurückbleibt.

Listing 6.5: Eine Grafik als Feldfunktion einfügen und den Feldcode anschließend bearbeiten

```
Private Const strGRAFIKPFAD1 As String = "C:\\Windows\\Seifenblase.bmp"
Private Const strGRAFIKPFAD2 As String = "C:\\Windows\\Feder.bmp"

Sub GrafikAlsFeldEinfuegen()
    Dim ils As Word.InlineShape
    Dim rng As Word.Range

    Set rng = Selection.Range
    'Grafik in die Zeile mit dem Text einfügen
    Set ils = rng.Fields.Add(Range:=rng, Type:=wdFieldEmpty, _
        Text:="IncludePicture "" & strGRAFIKPFAD1 & "" \* Mergeformat \d ",
        PreserveFormatting:=False).Result.InlineShapes(1)
    'Grafikgröße um 50% reduzieren
    ils.ScaleHeight = 50
    ils.ScaleWidth = 50
    'Feldcode ändern
    GrafikFeldcodeBearbeiten ils, strGRAFIKPFAD2, True, True
    'Die Verknüpfung auflösen
    ils.Fields.Unlink
End Sub

Sub GrafikFeldcodeBearbeiten(ByRef ils As Word.InlineShape, _
    ByRef strPfad As String, ByVal bMergeFormat As Boolean, _
    ByVal bSaveInDocument As Boolean)
```

```

Dim strMergeFormat As String
Dim strSaveInDocument As String
Dim rng As Word.Range

Set rng = ils.Range
strMergeFormat = ""
strSaveInDocument = ""
If bMergeFormat Then
    strMergeFormat = " \* MergeFormat"
End If
If Not bSaveInDocument Then
    strSaveInDocument = " \d"
End If
With rng.Fields(1)
    .Code.Text = "IncludePicture " & Chr$(34) & strPfad & Chr$(34) _
        & strMergeFormat & strSaveInDocument & " "
    .Update
End With
'Das InlineShape-Objekt wird durch die Aktualisierung zerstört
'Wir stellen es wieder her
Set ils = rng.InlineShapes(1)
End Sub

```

CD-ROM Beginn

Die Beispieldatei *Bsp06_02_Graf.doc* finden Sie auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

CD-ROM Ende

Hinweis Beginn

Das Dialogfeld für die Verwaltung verknüpfter Objekte befand sich in früheren Word-Versionen in der Menüfolge *Bearbeiten/Verknüpfungen*. In Word 2007 müssen Sie unter der »Office«-Schaltfläche *Vorbereiten*, dann *Verknüpfungen mit Dateien bearbeiten* anwählen.

Hinweis Ende

Layout-Optionen

Allgemein sind *InlineShape*-Objekte *Shape*-Objekten vorzuziehen. Einige der Gründe wurden im Abschnitt »Verknüpfungen erstellen und verwalten« in diesem Kapitel aufgelistet. Es gesellen sich zu den Problemen mit der Link-Verwaltung:

- Nur *InlineShape*-Objekte sind in der Gliederungs- und Normalen- (in Word 2007 Entwurfs-) Ansicht sichtbar.
- *InlineShape*-Objekte können problemlos als verborgener Text formatiert werden. (Eine oft gestellte Frage ist, wie man den Ausdruck von gewissen Grafiken unterbindet.)
- *InlineShape*-Objekte sind berechenbarer. Bei leichter Dokumentbeschädigung wird die Positionierung von *Shape*-Objekten instabil.
- Beschriftungen in der Zeichnungsebene werden vor Word 2007 bei der Bildung von Inhaltsverzeichnissen und Querverweisen ignoriert.

Manchmal geht es auch tatsächlich ohne Textflussformatierung, aber gelegentlich ist sie ein »notwendiges Übel«. Glücklicherweise stehen Alternativen bereit. Sie können beispielsweise eine Grafik als `InlineShape` in eine Tabellenzelle einfügen. Da Word 2000 und später den Textfluss um Tabellen unterstützt, aber die Tabelle sowie ihr Inhalt weiterhin als Teil des Dokumenttextes behandelt, entfallen die aufgelisteten Probleme.

Beschriftungen

Eine Tabelle als Behälter für Grafiken ist auch nützlich für Beschriftungen in wissenschaftlichen Dokumenten, die rechts ausgerichtet neben der Grafik stehen sollen.

Eine weitere, oft benutzte Möglichkeit ist, die Grafik als `InlineShape` in einen Positionsrahmen einzufügen. Beim Verschieben bleiben Grafik und Beschriftung zusammen.

Die beiden oben vorgestellten Lösungen finden Sie mit Code-Beispielen im Kapitel IV des Bonusteils auf der CD.

Ein weiterer Vorteil des Einfügens einer Grafik in einen Positionsrahmen ist, wenn dieser eine feste Breite oder Höhe hat, passt sich die Grafik dieser Dimension proportional an und die Größe nicht weiter bearbeitet werden muss. Beispielcode hierfür finden Sie in Listing 6.6.

Listing 6.6: Grafik in einen Positionsrahmen mit fester Breite einfügen

```
Private Const strMSGTITEL As String = "Grafik einfügen"

Sub GrafikInPositionsRahmenEinfuegen()
    Dim rng As Word.Range
    Dim fram As Word.Frame
    Dim ils As Word.InlineShape

    On Error GoTo Fehlerbehandlung
    Set rng = Selection.Range.Paragraphs(1).Range
    'Der Positionsrahmen wird die gesamte Markierung oder den Absatz, worin sich die
    'Einfügemarke befindet, beinhalten. Es soll aber nur eine Absatzmarke beinhalten.
    If Len(rng.Text) > 1 Then
        'Der Positionsrahmen soll neben dem Absatz stehen, worin sich die Einfügemarke
        'befindet. Ein leerer Absatz muss also VOR dem aktuellen Absatz eingefügt werden,
        ' falls dieser nicht leer ist
        rng.Collapse wdCollapseStart
        rng.Text = vbCr
    End If
    'Positionsrahmen einfügen und formatieren
    Set fram = rng.Frames.Add(rng)
    fram.Borders.Enable = False
    fram.Width = CentimetersToPoints(3.6)
    'Den Bereich in den Positionsrahmen setzen
    Set rng = fram.Range
    rng.Collapse Direction:=wdCollapseEnd
    Set ils = rng.InlineShapes.AddPicture(FileName:="C:\Windows\Feder.bmp", _
        LinkToFile:=False, Range:=rng)
    Set rng = ils.Range
    rng.Collapse Direction:=wdCollapseEnd
    'Ein Leerzeichen nach dem Positionsrahmen
```

```

rng.InsertAfter " "
rng.Font.Size = 1

Exit Sub
Fehlerbehandlung:
Select Case Err.Number
Case 4605
    MsgBox "Die Einfügemarke muss im Dokumenttext sein.", _
        vbOKOnly + vbCritical, strMSGTITEL
Case Else
    MsgBox "Unerwarteter Fehler: " & CStr(Err.Number) & vbCr & vbCr & _
        Err.Description, vbOKOnly + vbCritical, strMSGTITEL
End Select
End Sub

```

CD-ROM Beginn

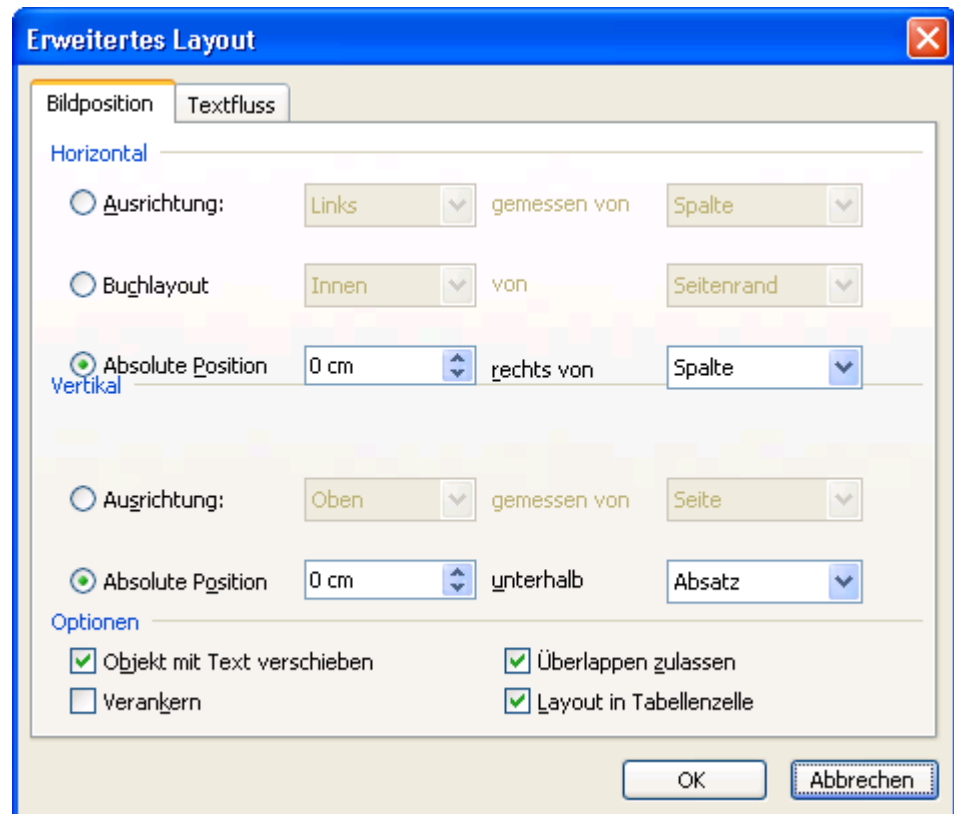
Die Beispieldatei *Bsp06_03_Graf.doc* finden Sie auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

CD-ROM Ende

Die Positionierung von Shape-Objekten

Da Word InlineShape-Objekte wie andere Textzeichen im Dokument behandelt, bedarf das Thema Positionierung keiner eingehenden Diskussion mehr. Shape-Objekte (und Positionsrahmen) hingegen unterliegen einer komplexen Regelung von Optionen. In der Benutzerschnittstelle befinden sich diese in der Menüfolge *Format/Grafik/Layout/Weitere* (in Word 2007 in der Registerkarte *Bildtools/Format*, im Menüpunkt *Weitere Layout Optionen* der Schaltflächen *Position* oder *Textumbruch*), in der Registerkarte *Bildposition* (Abbildung 6.2) bzw. unter *Format/Positionsrahmen*. (Nicht alle der hier vorgestellten Optionen gelten für Positionsrahmen, aber die Verhaltensregel sind die gleichen.) Es ist empfehlenswert, sich mit der Wirkung der verschiedenen Optionen vertraut zu machen, bevor Sie versuchen, grafische Objekte programmtechnisch zu positionieren.

Abbildung 6.2: Die Optionen, die die Positionierung von Shape-Objekten regeln



Die erste und wichtigste Regel für die Positionierung ist, dass ein grafisches Objekt (Shape oder Positionsrahmen (Frame)) *immer* mit einem Absatz verankert ist, und sich *immer* auf der gleichen Seite befindet wie dieser Absatz. In Abbildung 6.3 sehen Sie einen Objektanker. Falls das entsprechende Kontrollkästchen über *Extras/Optionen* auf der Registerkarte *Ansicht* aktiviert ist, soll er sichtbar sein, wenn sein grafisches Objekt markiert ist. (In Word 2007 befindet sich diese Option in der Registerkarte *Anzeigen* der Word-Optionen.)

Der Anker kann mit der Maus verschoben werden, eine programmtechnische Schnittstelle dafür gibt es jedoch nicht. Der verankernde Bereich kann lediglich beim Einfügen eines Shape-Objekts über die *AddPicture*-Methode bestimmt werden. Zwar hat das Shape-Objekt eine *Anker*-Eigenschaft, diese ist jedoch nur lesbar und gibt den ankernden Bereich (ein *Range*-Objekt) zurück.

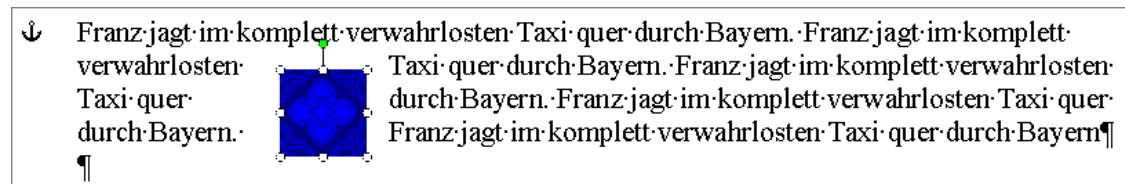
Tipp Beginn

Profitipp

Wenn Sie ein bereits im Dokument vorhandenes grafisches Objekt explizit mit einem bestimmten Absatz verankern wollen, müssen Sie das Objekt ausschneiden (*Cut*-Methode), und diesen Absatz als Zielbereich für das Einfügen (*Paste*-Methode) auswählen.

Tipp Ende

Abbildung 6.3: Der Anker eines grafischen Objekts befindet sich immer links des Absatzes, mit dem das Objekt verankert ist



Verankern = LockAnchor

Wird das grafische Objekt mit der Maus verschoben, springt der Anker meistens zum nächst liegenden Absatz. Um dieses Verhalten zu unterbinden, muss das Kontrollkästchen *Verankern* auf der Registerkarte *Bildposition* aktiviert werden. Diese Option entspricht der Eigenschaft `LockAnchor` des `Shape`- oder `Frame`-Objekts.

Viele meinen, diese Option würde das grafische Objekt sperren, so dass es nicht verschoben werden kann, oder gar fest mit einer bestimmten Seite verbunden wird. Das ist nicht der Fall; dafür gibt es in Word gar keine Möglichkeit (außer des allgemeinen Dokumentschutzes).

Hinweis Beginn

Es ist möglich, mit Code ein grafisches Objekt mit einer Seite zu assoziieren, so dass es immer wieder auf die angegebene Seite zurückgesetzt werden kann. Die Lösung finden Sie in Kapitel IV im Bonusteil auf der CD und enthält auch Beispiele für die in diesem Abschnitt vorgestellten Eigenschaften.

Hinweis Ende

Objekt mit Text verschieben

Grundsätzlich gibt es zwei Arten der Positionierung für grafische Objekte: relativ zur Seite oder relativ zum verankernden Text. In der Benutzerschnittstelle entsprechen diese der Einstellung des Kontrollkästchens *Objekt mit Text verschieben* auf der Registerkarte *Bildposition*. Ist die Option nicht aktiviert, bleibt die Grafik an der gleichen Position auf der Seite, in sich der verankernde Absatz befindet, egal wo der Absatz auf der Seite steht. Wird er zu einer anderen Seite verschoben, geht das grafische Objekt mit, und steht wieder in der gleichen Position relativ zur neuen Seite. War die Grafik beispielsweise 3 cm vom linken und 5 cm vom oberen Seitenrand auf Seite 4 zu finden, steht sie 3 cm vom linken und 5 cm vom oberen Seitenrand auch auf Seite 5, 6, oder gar 75.

Die genaue Position wird mit den weiteren Optionen in den beiden oberen Abschnitten der Registerkarte festgelegt. Die Möglichkeiten sind vielfältig und etwas verwirrend. Im Objektmodell sorgen eine Kombination von `RelativeVerticalPosition` (senkrecht) und von `RelativeHorizontalPosition` (waagrecht) in Zusammenspiel mit den `Top`- (oben) bzw. `Left`- (links) Eigenschaften für die Festlegung des Layouts. Die möglichen Werte der ersten beiden Eigenschaften sind in Tabelle 6.2 sowie in Tabelle 6.3 zusammengefasst.

Tabelle 6.2: Werte für die Eigenschaft `RelativeVerticalPosition`

Enumeration	Wert	Beschreibung
<code>wdRelativeVerticalPositionFromText</code>	3	Der Abstand wird relativ zur verankernden Zeile gemessen (mit Text verschieben)
<code>wdRelativeVerticalPositionFromPage</code>	0	Der Abstand wird vom oberen Textrand gemessen

wdRelativeVerticalPositionPage	1	Der Abstand wird vom oberen Seitenrand gemessen
wdRelativeVerticalPositionParagraph	2	Der Abstand wird relativ zum verankernden Absatz gemessen (mit Text verschieben)

Tabelle 6.3: Werte für die Eigenschaft RelativeHorizontalPosition

Enumeration	Wert	Beschreibung
wdRelativeHorizontalPositionCharacter	3	Der Abstand wird relativ zum verankernden Textzeichen gemessen (mit Text verschieben)
wdRelativeHorizontalPositionColumn	2	Der Abstand wird vom linken Spaltenrand gemessen
wdRelativeHorizontalPositionMargin	0	Der Abstand wird vom linken Textrand gemessen
wdRelativeHorizontalPositionPage	1	Der Abstand wird vom linken Seitenrand gemessen

Zusätzlich zu einem numerischen Wert des Datentyps Single akzeptieren Left und Top noch die Enumerationen in Tabelle 6.4.

Tabelle 6.4: Werte für der Eigenschaften Left und Top

Enumeration	Wert	Position, relativ zum verankernden Text
wdShapeBottom	-999997	Unten
wdShapeCenter	-999995	Zentriert
wdShapeInside	-999994	Die Grafik wird am Bundrand eines Buches ausgerichtet (<i>Gerade/ungerade anders</i> in der Menüfolge <i>Datei/Seiteneinrichten/Layout</i>) links
wdShapeLeft	-999998	links
wdShapeOutside	-999993	Die Grafik wird am Außenrand eines Buches ausgerichtet (<i>Gerade/ungerade anders</i> in der Menüfolge <i>Datei/Seiteneinrichten/Layout</i>) rechts
wdShapeRight	-999996	Rechts
wdShapeTop	-999999	Oben

Um eine Grafik 3 cm vom linken Seitenrand und bündig mit dem oberen Textrand zu positionieren, wird wie in Listing 6.7 vorgegangen.

Listing 6.7: Ein grafisches Objekt relativ zur Seite positionieren

```
Sub GrafikGenauAufDerSeitePositionieren()
    Dim shp As Word.Shape

    'Zweites InlineShape im Dokument in ein Shape konvertieren
    Set shp = ActiveDocument.InlineShapes(2).ConvertToShape
    'Relativ zur Seite positionieren
    shp.RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
    shp.Left = CentimetersToPoints(3)
    shp.RelativeVerticalPosition = wdRelativeVerticalPositionMargin
    shp.Top = wdShapeTop
End Sub
```

CD-ROM Beginn

Die Beispieldatei *Bsp06_03_Graf.doc* finden Sie auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

CD-ROM Ende

Tipp Beginn

Die Funktion *CentimetersToPoints*

In vielen Word-Dialogfeldern können Dimensionen in verschiedenen Maßen angegeben werden, wie etwa Zoll (Inches), Zentimeter und Points. Intern erwartet Word aber nur eine dieser Maßangaben – meistens Points –, was bedeutet, die anderen müssen konvertiert werden. Word stellt im Objektmodell eine Sammlung solcher Konvertierfunktionen zur Verfügung, wie `PointsToCentimeters`, `InchesToPoints`, `MillimetersToPoints`, `LinesToPoints`, `PicasToPoints` und umgekehrt. Diese Funktionen gehören ausschließlich dem Word- und keinem anderen Office-Objektmodell an.

Tipps Ende

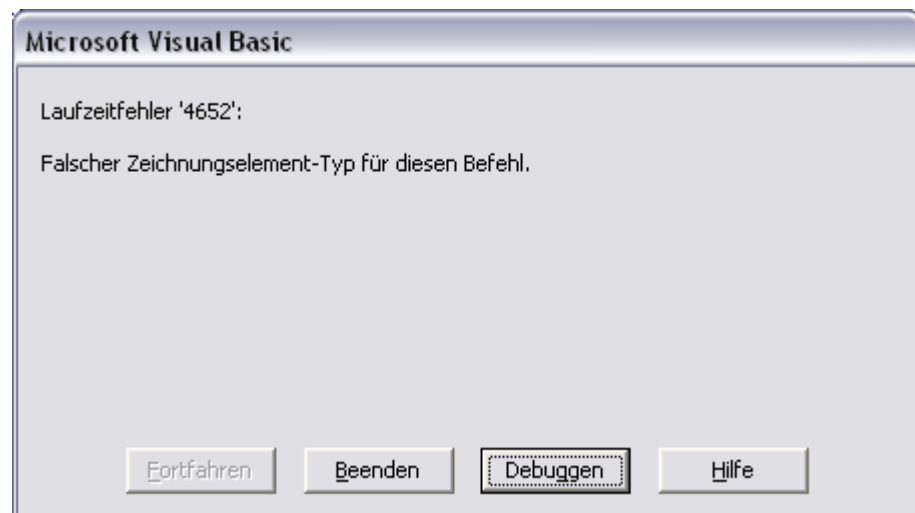


2007

Neue Positionierungs- und Vergrößerungsmöglichkeiten in Office 2007

Zusätzlich zu den oben beschriebenen Positionierungsmöglichkeiten wurden für Office 2007 Grafiken weitere eingeführt. Diese stehen in Word hauptsächlich AutoFormen zur Verfügung. Der Laufzeitfehler in **Abbildung 6.4** erscheint, wenn sie mit einem Shape-Objekt des falschen Typs verwendet werden.

Abbildung 6.4: Fehlermeldung, wenn ein Shape-Objekt mit den neuen Grafikeigenschaften nicht kompatibel ist



Die Tabelle 6.5 bis Tabelle 6.7 bieten einen Überblick der neuen Enumerationen.

Das Office-Objektmodell wurde entsprechend um die Eigenschaften `LeftRelative`, `WidthRelative`, `TopRelative` sowie `HeightRelative` erweitert, um die Position bzw. Größe des grafischen Objekts festzulegen.

Diese neue Funktionalität unterstützt die Positionierung und Größenänderung relativ zu den Seitenränder. Angaben werden in Prozent der Breite bzw. Höhe des angegebenen Seitenrands festgelegt. Ein Beispiel für deren Verwendung sehen Sie in **Listing 6.8**.

Tabelle 6.5: Zusätzliche Werte für die Eigenschaft `RelativeVerticalPosition` in Office 2007

Enumeration	Wert	Beschreibung
-------------	------	--------------

wdRelativeVerticalPositionTopMarginArea	4	Positioniert an den oberen Seitenrand
wdRelativeVerticalPositionBottomMarginArea	5	Positioniert an den unteren Seitenrand
wdRelativeVerticalPositionInnerMarginArea	6	Bei gespiegelten Seitenränder, positioniert an den inneren Setienand
wdRelativeVerticalPositionOuterMarginArea	7	Bei gespiegelten Seitenränder, positioniert an den äusseren Setienand
wdShapePositionRelativeNone	-99999	Prozentuelle Positionierungen werden ignoriert

Tabelle 6.6: Zusätzliche Werte für die Eigenschaft RelativeHorizontalPosition in Office 2007

Enumeration	Wert	Beschreibung
wdRelativeHorizontalPositionLeftMarginArea	4	Positioniert an den linken Seitenrand
wdRelativeHorizontalPositionRightMarginArea	5	Positioniert an den rechten Rand
wdRelativeHorizontalPositionInnerMarginArea	6	Bei gespiegelten Seitenränder, positioniert an den inneren Setienand
wdRelativeHorizontalPositionOuterMarginArea	7	Bei gespiegelten Seitenränder, positioniert an den äusseren Setienand

Tabelle 6.7: Werte für die Eigenschaften RelativeHorizontalSize sowie RelativeVerticalSize in Office 2007

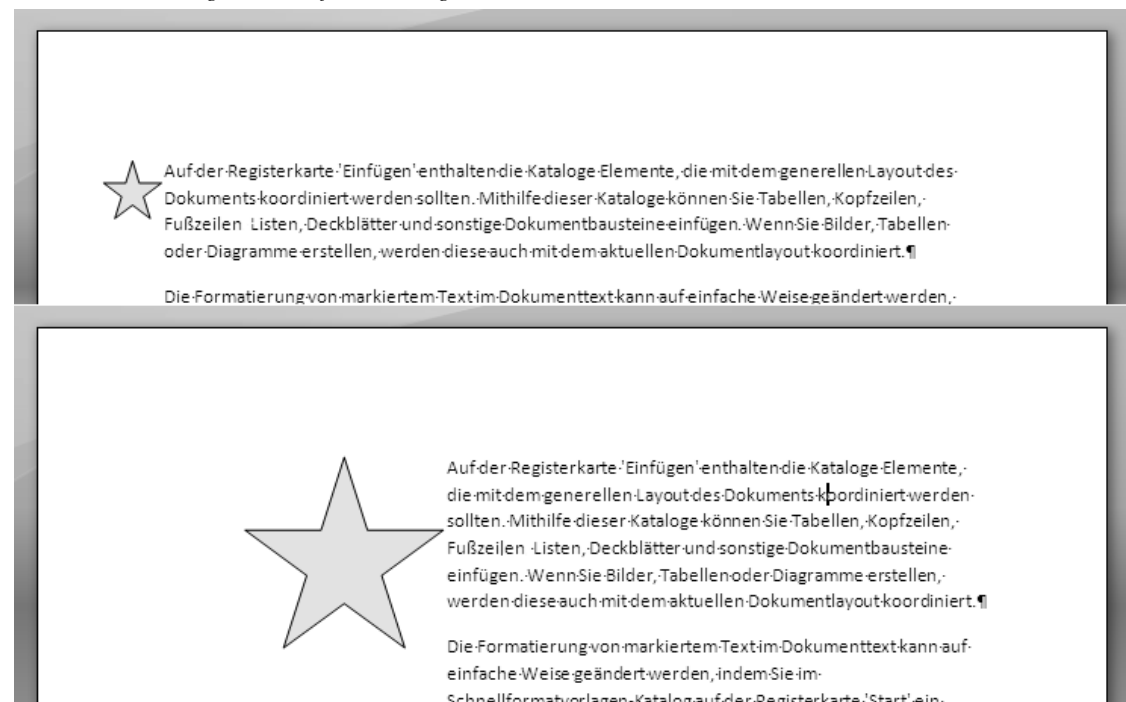
Enumeration	Wert	Beschreibung
wdRelativeHorizontalSizeInnerMarginArea	4	Die Breite ist relativ zu der Größe des inneren Rands: bei ungeraden Seiten relativ zu der Größe des linken Seitenrands, bei geraden Seiten relativ zu der Größe des rechten Seitenrands
wdRelativeHorizontalSizeLeftMarginArea	2	Die Breite ist relativ zu der Größe des linken Seitenrands
wdRelativeHorizontalSizeMargin	0	Die Breite ist relativ zu dem Abstand zwischen dem linken und dem rechten Seitenrand
wdRelativeHorizontalSizeOuterMarginArea	5	Die Breite ist relativ zu der Größe des äußeren Rands: bei ungeraden Seiten relativ zu der Größe des rechten Seitenrands, bei geraden Seiten relativ zu der Größe des linken Seitenrands.
wdRelativeHorizontalSizePage	1	Die Breite ist relativ zu der Breite der Seite.
wdRelativeVerticalSizeBottomMarginArea	3	Die Höhe ist relativ zu der Größe des unteren Seitenrands.
wdRelativeVerticalSizeInnerMarginArea	4	Die Höhe ist relativ zu der Größe des inneren Rands: bei ungeraden Seiten relativ zu der Größe des oberen Seitenrands, bei geraden Seiten relativ zu der Größe des unteren Seitenrands.
wdRelativeVerticalSizeMargin	0	Die Höhe ist relativ zu dem Abstand zwischen dem linken und dem rechten Seitenrand.
wdRelativeVerticalSizeOuterMarginArea	5	Die Höhe ist relativ zu der Größe des äußeren Rands: bei ungeraden Seiten relativ zu der Größe des unteren

		Seitenrands, bei geraden Seiten relativ zu der Größe des oberen Seitenrands.
wdRelativeVerticalSizePage	1	Die Höhe ist relativ zu der Höhe der Seite.
wdRelativeVerticalSizeTopMarginArea	2	Die Höhe ist relativ zu der Größe des oberen Seitenrands.
wdShapeSizeRelativeNone	-	Prozentuelle Größenänderungen werden ignoriert.
	9999	
	99	

Die Prozedur `FormRelativ` arbeitet mit einem Zeichnungsobjekt, das im Dokument vorgängig benannt wurde (dessen `Name`-Eigenschaft auf »Stern« festgelegt wurde). Die Größe wird auf 50% der Breite des linken Seitenrands festgelegt. In **Abbildung 6.5** ist ersichtlich, wie die Änderung der Seitenrandbreite sich auf die Breite des Sterns auswirkt. Zudem wird der linken Rand relativ zur linken Seitenrand auf eine Weite von 50% der Breite des Seitenrands festgelegt.

Die senkrechte Position ist einfach zum oberen Rand; die Höhe ist gleich die Absolute breite des Objekts festgelegt. Dies hat zur Folge, dass sie sich bei einer Änderung des linken Rands nicht anpasst. Um das Ergebnis in **Abbildung 6.5** zu erzielen wurde die Prozedur nochmals durchgeführt.

Abbildung 6.5: Die relative Positionierung sowie Größenänderung einer AutoForm in Word 2007



Listing 6.8:

```
Sub FormRelativ()
    Dim shp As Word.Shape

    'Die AutoForm wurde vorgängig explizit benannt
    Set shp = ThisDocument.Shapes("Stern")
    shp.LockAspectRatio = msoFalse
    shp.RelativeHorizontalSize = wdRelativeHorizontalSizeLeftMarginArea
```

```

shp.WidthRelative = 50
shp.Height = shp.Width
shp.RelativeHorizontalPosition = wdRelativeHorizontalPositionLeftMarginArea
shp.LeftRelative = 50
shp.RelativeVerticalPosition = wdRelativeVerticalPositionMargin
shp.Top = 0
shp.LockAspectRatio = msoTrue
End Sub

```

CDROM Beginn

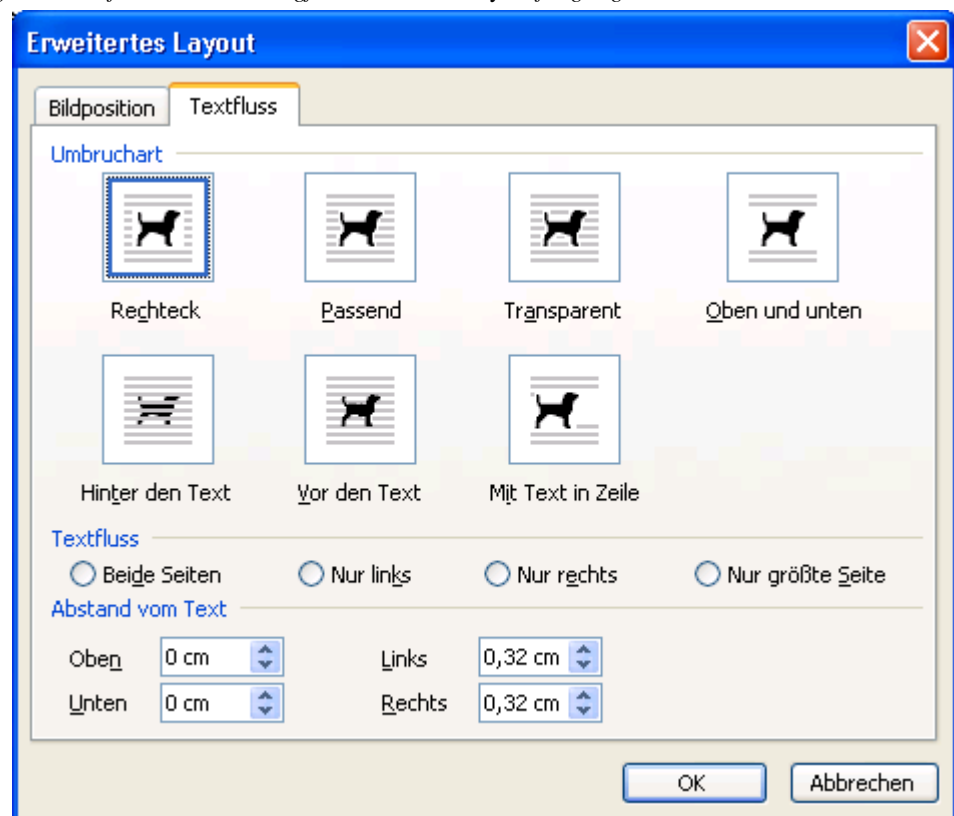
Das Word 2007 Beispieldokument *Bsp06_06_Graf.docm* finden Sie im Ordner *\Beilagen\Kap06_Grafiken\Bsp* auf der CD.

CDROM Ende

Textfluss-Formatierung

Neben der Registerkarte *Bildposition* enthält das Dialogfeld *Erweitertes Layout* die Registerkarte *Textfluss* (Abbildung 6.6), die einige zusätzliche Optionen zur Registerkarte *Layout* im Dialogfeld *Grafik formatieren* anbietet.

Abbildung 6.6: Der genaue Textfluss wird im Dialogfeld *Erweitertes Layout* festgelegt



Textfluss

Im Objektmodell umfasst die Eigenschaft `WrapFormat` die Optionen im Abschnitt *Textfluss* und *Abstand vom Text*. Auch die meisten Umbrucharten sind dieser Eigenschaft zugeteilt, außer *Hinter den Text* und *Vor den Text*, die unter die

Methode `ZOrder` fallen. Die `WrapFormat`-Eigenschaften mit ihren Werten und Wirkung sind in Tabelle 6.8 aufgelistet.

Tabelle 6.8: Die `WrapFormat`-Eigenschaften, die den Textfluss festlegen

Eigenschaft	Enumeration	Wert	Beschreibung oder Dialogfeldoption
Type	<code>wdWrapInline</code>	7	Gilt nur für Office-AutoFormen, die nicht in <code>InlineShape</code> -Objekte umgewandelt werden können. Die <code>AutoForm</code> verhält sich wie ein <code>InlineShape</code> , ist aber keines, was an den weißen Anfassern zu erkennen ist.
	<code>wdWrapNone</code>	3	Stellt das grafische Objekt vor den Text. Wird zurückgeben, wenn die Grafik vor oder hinter dem Text steht.
	<code>wdWrapSquare</code>	0	Entspricht der Option <i>Rechteck</i>
	<code>wdWrapThrough</code>	2	Entspricht der Option <i>Transparent</i> . (Die Hintergrundfarbe scheint durch die Stellen, die mit »transparente Farbe« formatiert sind. Steht nicht bei allen Grafiktypen zur Verfügung.)
	<code>wdWrapTight</code>	1	Entspricht der Option <i>Passend</i> . Der Textfluss folgt seitlich dem Umriss des Hauptteils des Bildes und ignoriert den Hintergrund.
	<code>wdWrapTopBottom</code>	4	Entspricht der Option <i>Oben und unten</i> . Allgemein ist ein <code>InlineShape</code> , allein stehend in einem Absatz, dieser Einstellung vorzuziehen.
Side	<code>wdWrapBoth</code>	0	<i>Textfluss/Beide Seiten</i>
	<code>wdWrapLargest</code>	3	<i>Textfluss/Nur größte Seite</i>
	<code>wdWrapLeft</code>	1	<i>Textfluss/Nur links</i>
	<code>wdWrapRight</code>	2	<i>Textfluss/Nur rechts</i>
DistanceBottom			<i>Abstand vom Text/Unten</i>
DistanceLeft			<i>Abstand vom Text/Links</i>
DistanceRight			<i>Abstand vom Text/Rechts</i>
DistanceTop			<i>Abstand vom Text/Oben</i>

Wenn wir das grafische Layout eines Word-Dokuments betrachten, sehen wir, dass es aus drei dimensional »Ebenen« besteht: aus der Textebene, der Ebene dahinter, und der Ebene davor. Diese sind in der Abbildung 6.7 klar ersichtlich. Von links nach rechts: hinter dem Text, die Textebene, und vor dem Text.

Abbildung 6.7: Die drei grafische Ebenen eines Word-Dokuments

wdTightAll	1	Text umfließt den Inhalt von Textfeldern auf allen Zeilen.
wdTightFirstAndLastLines	2	Text umfließt nur die erste und letzte Zeile.
wdTightFirstLineOnly	3	Text umfließt nur die erste Zeile.
wdTightLastLineOnly	4	Text umfließt nur die letzte Zeile.
wdTightNone	0	Text umfließt den Inhalt eines Textfelds nicht.

Um dies zu realisieren, muss das Textfeld die folgenden Bedingungen erfüllen:

- Es darf weder einen Rahmen noch eine Schattierung haben.
- Der Textumbruch für das Textfeld muss auf *passend* festgelegt werden.

Listing 6.9 zeigt, wie das Textfeld in **Abbildung 6.8** erstellt wurde.

Listing 6.9: Ein Textfeld mit Textumbruch um den Text im Textfeld formatieren

```
Sub TextFeldMitUmbruch()
    Dim shp As Word.Shape
    Dim rng As Word.Range

    Set rng = ActiveDocument.Paragraphs(2).Range
    Set shp = ActiveDocument.Shapes.AddTextbox(Orientation:=msoTextOrientationHorizontal, _
        Left:=0, Top:=0, Width:=120, Height:=140, Anchor:=rng)

    shp.RelativeHorizontalPosition = wdRelativeHorizontalPositionColumn
    shp.Left = wdShapeCenter
    shp.RelativeVerticalPosition = wdRelativeVerticalPositionParagraph
    shp.Top = wdShapeCenter
    shp.Fill.Visible = msoFalse
    shp.Line.Visible = msoFalse
    shp.WrapFormat.Type = wdWrapTight

    With shp.TextFrame
        .TextRange = str1 & vbCrLf & str2
        .TextRange.Bold = True
        .TextRange.Font.Name = "Times New Roman"
        .TextRange.ParagraphFormat.TextboxTightWrap = wdTightAll
        .TextRange.ParagraphFormat.Alignment = wdAlignParagraphCenter
    End With
End Sub
```

Zeichnen/Reihenfolge

Zudem können in allen drei Ebenen Grafiken übereinander liegen. Diese Reihenfolge wird in der Benutzerschnittstelle über den Menübefehl *Reihenfolge* der Schaltfläche *Zeichnen* in der gleichnamigen Symbolleiste definiert. (In Word 2007 stehen die Befehle in der Gruppe *Anordnen* der Registerkarte *Zeichentools/Format*.) Im Objektmodell entspricht diesem Befehl die Methode `ZOrder`, deren Werte in Tabelle 6.10 aufgelistet sind.

Tabelle 6.10: Die Werte der Methode `ZOrder`

Enumeration	Menübefehl
<code>msoBringForward</code>	<i>Eine Ebene nach vorne</i>
<code>msoBringToFront</code>	<i>Vor den Text bringen</i>
<code>msoSendBackward</code>	<i>In den Vordergrund</i>
<code>msoSendBehindText</code>	<i>Eine Ebene nach hinten</i>
	<i>Hinter den Text bringen</i>

msoSendToBack

In den Hintergrund

Das Zusammenspiel der `WrapFormat`-Eigenschaften und der `ZOrder`-Methode kann recht spannend sein. Leider gibt es im Word-Objektmodell keine Methode, um festzustellen, in welcher dreidimensionalen Reihenfolge Grafiken in einer Ebene zueinander stehen. Es bietet zwar die Eigenschaft `ZOrderPosition`, aber diese verrät uns nur, in welcher Reihenfolge die Objekte ins Dokument eingefügt wurden (die zuletzt eingefügte steht an oberster Stelle), nicht welche Grafik vor oder hinter einer anderen steht.

Zusammenfassend:

- Sie können jederzeit herausfinden, mit welchem Textfluss eine Grafik formatiert ist, was meist verrät, ob sie sich in der Textebene befindet (nur `wdWrapNone` ist nicht in der Textebene).
- Es ist immer möglich, ein grafisches Objekt in eine beliebige Ebene, mit einem beliebigen Textfluss, zu positionieren.
- Auch die Festlegung einer dreidimensionalen Reihenfolge stellt kein Problem dar.
- Nicht herausfinden können Sie allerdings mit VBA, in welcher Reihenfolge grafische Objekte in der gleichen Ebene übereinander liegen.

Das folgende Beispiel soll diese Prinzipien grafisch veranschaulichen.

In Abbildung 6.9 sehen Sie drei AutoFormen (alle in der gleichen Ebene vor dem Text stehend), die in der Reihenfolge nummeriert sind, wie sie in das Dokument eingefügt wurden. Die Prozedur *GrafikenAusrichtenUndEinordnen* in Listing 6.10 richtet sie zuerst waagrecht sowie senkrecht zentriert auf der Seite (also übereinander) aus. Wie in Abbildung 6.10 ersichtlich, steht die zuletzt eingefügte Grafik (Nummer 3) vorn. Danach durchläuft das Makro nochmals alle Grafiken und schickt zuerst die Grafik mit `ZOrderPosition` 1 nach hinten, dann die Nummer 2 und zuletzt die Nummer 3, so dass am Schluss die Nummer 1 wie in Abbildung 6.11 vorn erscheint.

Wenn anschließend das orangefarbene Dreieck hinter den Text gestellt wird, behält es seine `ZOrderPosition`, erscheint aber hinter den anderen Grafiken, da es hinter dem Text steht. Dieser Zustand ist jedoch, was der `ZOrderPosition` angeht, rein optisch.

Hinweis Beginn

Bitte bemerken Sie, wenn Sie mit einer `For Each...Next`-Schleife alle `Shape` Objekte in einem Dokument oder Bereich durchlaufen, dass diese in der Reihenfolge ihrer Verankerung geschieht. Mit dieser Reihenfolge hat die `ZOrderPosition` nichts zu tun.

Hinweis Ende

Abbildung 6.9: Die Grafiken sind in der Reihenfolge nummeriert, in welcher sie eingefügt wurden

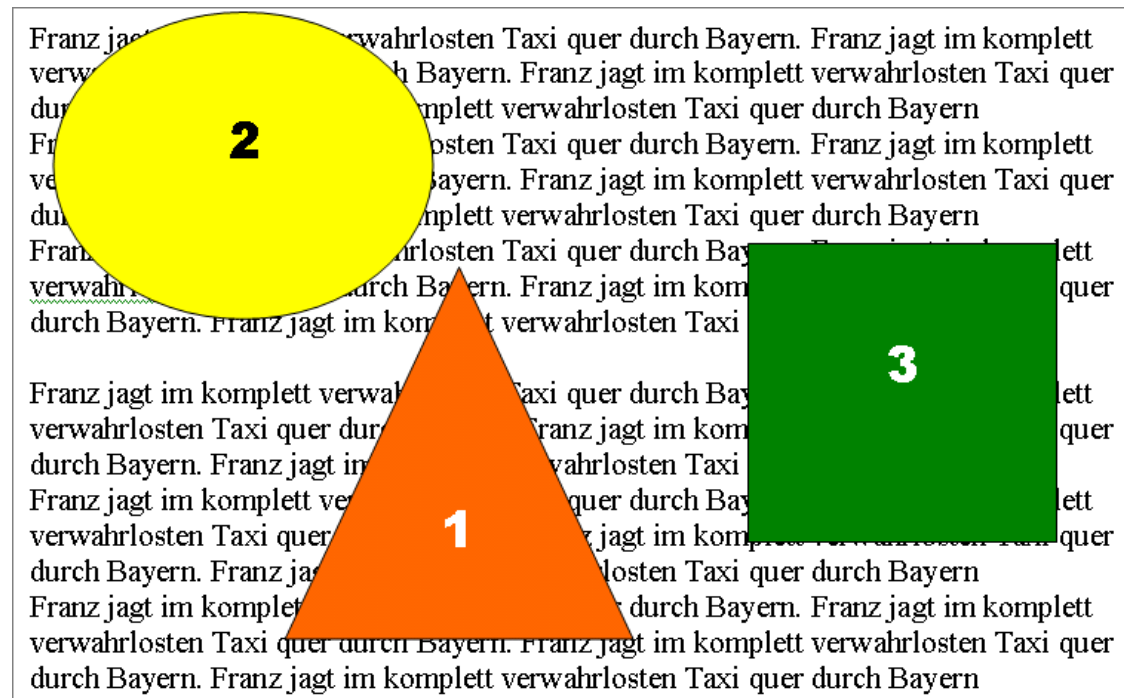


Abbildung 6.10: Zieht man sie über einander, liegt die zuletzt eingefügte zuoberst

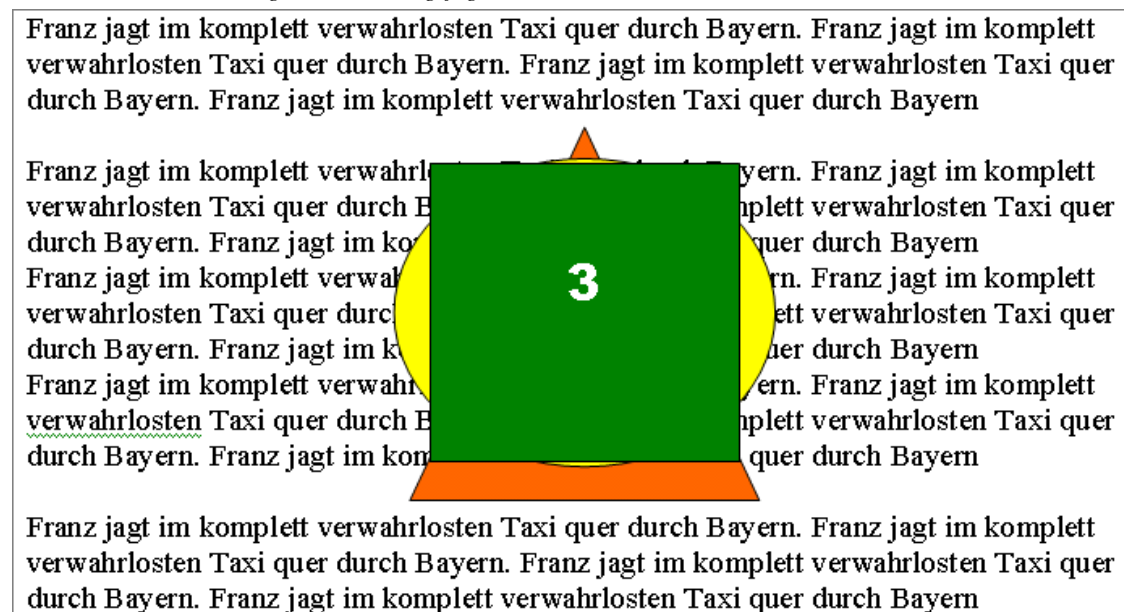
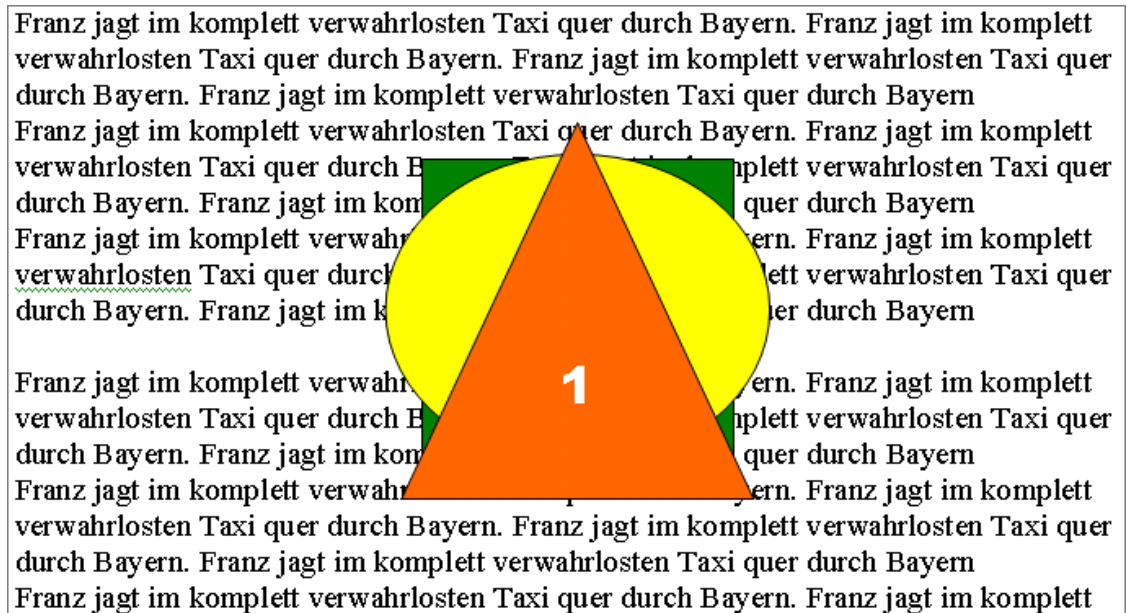


Abbildung 6.11: Die Prozedur in Listing 6.10 hat die ZOrder umgekehrt



Listing 6.10: AutoFormen zentriert ausrichten und hintereinander anordnen

```

Sub GrafikenAusrichtenUndEinordnen()
    Dim rng As Word.Range
    Dim shpRng As Word.ShapeRange
    Dim shp As Word.Shape
    Dim pgs As Word.PageSetup

    On Error GoTo FehlerBehandlung
    ' Markierung in den Text verschieben.
    Selection.GoTo What:=wdGoToPage, _
        Count:=Selection.Information(wdActiveEndPageNumber)
    ' Nur die Grafiken dieser Seite bearbeiten.
    Set rng = Selection.Bookmarks("\Page").Range
    Set shpRng = rng.ShapeRange
    ' Objektvariable, um Seitenrandinformationen zu ermitteln.
    Set pgs = rng.Sections(1).PageSetup

    'Die Grafiken werden in der Reihenfolge ihrer Verankerungen bearbeitet!
    For Each shp In shpRng
        'Grafik zentriert relativ zu den Texträndern positionieren.
        shp.RelativeHorizontalPosition = wdRelativeHorizontalPositionMargin
        shp.Left = wdShapeCenter
        shp.RelativeVerticalPosition = wdRelativeVerticalPositionMargin
        shp.Top = wdShapeCenter
    Next shp

    'Die Reihenfolge der Grafiken umkehren.
    For Each shp In shpRng
        Debug.Print shp.Name, shp.ZOrderPosition
        If InStr(shp.TextFrame.TextRange.Text, "1") <> 0 Then shp.ZOrder msoSendToBack
    Next shp

```

```

For Each shp In shpRng
    Debug.Print shp.Name, shp.ZOrderPosition
    If InStr(shp.TextFrame.TextRange.Text, "2") <> 0 Then shp.ZOrder msoSendToBack
Next shp
For Each shp In shpRng
    Debug.Print shp.Name, shp.ZOrderPosition
    If InStr(shp.TextFrame.TextRange.Text, "3") <> 0 Then shp.ZOrder msoSendToBack
    shp.Select
Next shp
' Die oberste Grafik hinter den Text schicken; sie erscheint jetzt
' hinter allen anderen. Aber ihre ZOrderPosition bleibt die gleiche.
' Um diese Wirkung zu sehen, die folgenden Zeilen auskommentieren.
' For Each shp In shpRng
'     If shp.ZOrderPosition = (3) Then shp.ZOrder msoSendBehindText
'     Debug.Print shp.Name, shp.ZOrderPosition
' Next shp
Exit Sub

FehlerBehandlung:
Select Case Err.Number
Case 5852
    MsgBox "Fehler: " & Err.Number & ". (Objekt nicht vorhanden)" & vbCrLf & _
        "Das Makro findet keine grafischen Objekte auf dieser Seite, " & _
        "die über dem Text liegen.", vbCritical + vbOKOnly
Case Else
    MsgBox "Fehler: " & Err.Number & vbCrLf & _
        "Beschreibung: " & Err.Description, vbCritical + vbOKOnly
End Select
End Sub

```

CD-ROM Beginn

Die Beispieldatei *Bsp06_04_Graf.doc* finden Sie auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

CD-ROM Ende

Zeichnungsobjekte (AutoFormen)

Wie schon in diesem Abschnitt erwähnt, ist die Zeichnungsfunktionalität in Word ein gemeinsames Office-Anwendungspaket, das von mehreren Anwendungen (wie PowerPoint und Excel) benutzt wird. Wir werden daher die in VBA zur Verfügung gestellte Zeichnungsfunktionalität nicht eingehend diskutieren, da das Thema selbst ein ganzes Buch füllen würde und in anderen Büchern behandelt wird. Wir greifen lediglich einige immer wiederkehrende Fragen auf, die eine allgemeine Idee geben, wie mit VBA-AutoFormen in einem Word-Dokument erstellt werden.

Hinweis Beginn

Angaben zu WordArt, das ein Objektmodell besitzt, finden Sie in Kapitel 12.

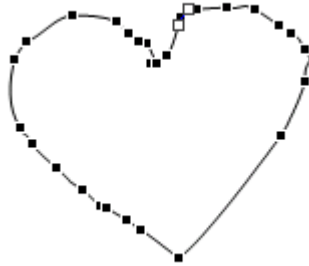
Hinweis Ende

Freihandformen bearbeiten

Etwas, das immer wieder für Unklarheit sorgt, ist die `Points`-Eigenschaft der `ShapeNodes`-Auflistung. Ein `ShapeNode` ist entweder ein Eckpunkt oder bei

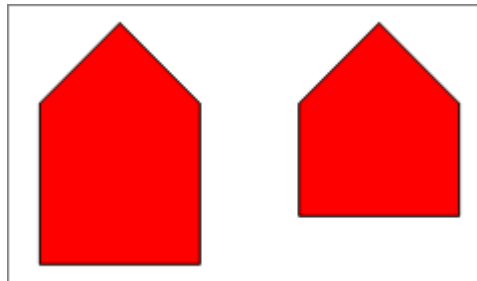
Kurven ein Punkt, der die Rundung bestimmt. Durch Änderung der Koordinaten (Points) wird das Aussehen einer *von Hand gezeichneten* AutoForm geändert. In der Benutzerschnittstelle werden sie über den Menüpunkt *Punkte bearbeiten* der *Zeichnen*-Symbolschaltfläche in der gleichnamigen Symbolleiste aktiviert und sehen wie in Abbildung 6.12 aus. (In Word 2007 befindet sich der Befehl im Menü *Form bearbeiten* der Gruppe *Formen einfügen* in der Registerkarte *Zeichentools/Format*.)

Abbildung 6.12: Durch Bearbeitung der Punkte kann die Form einer gezeichneten Grafik angepasst werden. Die Punkten können verschoben, gelöscht oder neu hinzugefügt werden.



Die Unklarheit stammt aus den Hilfe-Angaben zur Eigenschaft. Obwohl die Objekt-Hierarchie `Shape.Nodes.Item.Points` lautet, erscheint eine Fehlermeldung (»die Typen sind unverträglich«), wenn eine als `Shape` deklarierte Objekt-Variable eingesetzt wird. Um das Problem zu umgehen, muss die Objekt-Variable als allgemeiner Typ `Object` deklariert werden. Listing 6.11 veranschaulicht das Problem und die Lösung. Das Makro erstellt ein Fünfeck als Vieleck (Polygon), dann werden die zwei senkrechten Seiten verkürzt, wie in Abbildung 6.13 ersichtlich.

Abbildung 6.13: Ein Fünfeck mit VBA erstellen und bearbeiten



Zuerst wird eine Objekt-Variable – `shp` – für die AutoForm als `Shape` deklariert. In der Datenfeld-Variable `aEckPunkte` werden die Koordinaten des Fünfecks festgehalten. Je ein Koordinatenpaar braucht es für jeden Richtungswechsel, zusätzlich eines für den Anfangs- sowie den Endpunkt. Diese werden in der ersten Dimension des Datenfelds festgehalten. Die zweite Dimension des Datenfelds hält die X- und Y-Koordinaten jedes Punkts relativ zur Seite fest.

Hinweis Beginn

Haben Anfangs- und Endpunkt einer Freihandform (Polyline) wie hier die gleichen Koordinaten, ist die Form geschlossen und kann gefüllt werden.

Hinweis Ende

Nachdem die Koordinaten dem Datenfeld zugewiesen wurden, fügt die Prozedur das Fünfeck ein und setzt es der shp-Objektvariablen gleich. Damit kann die AutoForm weiter bearbeitet und formatiert werden – wie hier mit der Füllfarbe *Rot*.

Eigentlich sollten wir die gleiche Objekt-Variable einsetzen können, um die Eckpunktkoordinaten zu ändern. Nur tritt leider das beschriebene Problem auf. Es wird also die Objektvariable o_shp als Object deklariert und dem Shape gleich gesetzt.

Noch ein Datenfeld wird gebraucht, um die Koordinaten des Punktes festzuhalten, den wir verschieben möchten. Auch hier muss eine neue Objekt-Variable her, da Points nur einer Objekt-Variablen des Datentyps Variant (aber keinem Datenfeld) zugewiesen werden kann. Zwei Ungereimtheiten also, worauf zu achten sind.

aPunkte hält also die X- und Y- Koordinaten des gewählten Eckpunktes (Node) fest, die den Variablen x und y zugewiesen werden. Mit der Methode SetPosition werden diese geändert. In diesem Fall bleibt die waagrechte Einstellung gleich, die senkrechte wird um 15 Punkte (typographisches Maß) verkürzt (höher gestellt).

Listing 6.11: Ein Polygon erstellen und anschließend die Eckpunkte ändern

```
Sub EinfünfeckZeichnen()  
    Dim shp As Word.Shape  
    Dim aEckPunkte(1 To 6, 1 To 2) As Single  
    Dim vw As Word.View  
    Dim bCurVw As Boolean  
  
    Set vw = ActiveDocument.ActiveWindow.View  
    'In Word 2002 können Grafiken falsch positioniert werden,  
    'wenn die oberen und unteren Seitenränder ausgeblendet sind.  
    'Die Benutzereinstellung festhalten und am Schluss wieder herstellen  
    'WORD 2000: die beiden folgenden Zeilen entfernen!  
    bCurVw = vw.DisplayPageBoundaries  
    If bCurVw = False Then vw.DisplayPageBoundaries = True  
  
    aEckPunkte(1, 1) = 25  
    aEckPunkte(1, 2) = 25  
    aEckPunkte(2, 1) = 50  
    aEckPunkte(2, 2) = 50  
    aEckPunkte(3, 1) = 50  
    aEckPunkte(3, 2) = 100  
    aEckPunkte(4, 1) = 0  
    aEckPunkte(4, 2) = 100  
    aEckPunkte(5, 1) = 0  
    aEckPunkte(5, 2) = 50  
    aEckPunkte(6, 1) = 25  
    aEckPunkte(6, 2) = 25  
  
    Set shp = ActiveDocument.Shapes.AddPolyline(aEckPunkte)  
    shp.RelativeHorizontalPosition = wdRelativeHorizontalPositionMargin  
    shp.Left = wdShapeRight  
    shp.RelativeVerticalPosition = wdRelativeVerticalPositionParagraph  
    shp.Top = 0  
    shp.WrapFormat.Type = wdWrapSquare
```

```

shp.Fill.ForeColor = 255

Dim o_shp As Object
Dim aPunkte As Variant
Dim x As Single
Dim y As Single
Set o_shp = shp
With o_shp.Nodes
    aPunkte = .Item(3).Points
    x = aPunkte(1, 1)
    y = aPunkte(1, 2)
    .SetPosition 3, x, y - 15
    aPunkte = .Item(4).Points
    x = aPunkte(1, 1)
    y = aPunkte(1, 2)
    .SetPosition 4, x, y - 15
End With

'WORD 2000: die folgende Zeile entfernen!
vw.DisplayPageBoundaries = bCurVw
End Sub

```

Text in AutoFormen ansprechen

Den meisten AutoFormen kann Text hinzugefügt werden. Leider ist es nicht möglich, AutoFormen oder Textfeldern generell eine Formatvorlage oder andere Formatierung zu zuweisen. Neue AutoFormen werden immer mit der Formatvorlage *Standard* formatiert. Während es möglich ist, AutoFormen zu kopieren, stellen wir zunehmend Probleme in Word 2002 und 2003 fest. Word kann offensichtlich eine eingefügte AutoForm nicht zuverlässig vom kopierten Original unterscheiden, was zu mühsamen Vorgehensweisen bei der Dokumentbearbeitung führt.

Es ist also ratsam, jede AutoForm und Textfeld einzeln zu erstellen und zu formatieren. Beinhaltet das Dokument viele solche Objekte, ist dieser Vorgang kaum weniger mühsam. Eine programmtechnische Lösung drängt sich also auf. In Listing 6.12 finden Sie Beispielcode, der alle Textfelder (und ausschließlich diese) mit Textinhalt im Dokumenttext gleich formatiert.

Listing 6.12: Den Text in allen Textfeldern des Dokumentkörpers mit Arial 10, fett, formatieren

```

Sub AlleTextBereicheFormatieren()
    Dim shp As Word.Shape

    For Each shp In ActiveDocument.Shapes
        If shp.Type = msoTextBox Then
            With shp.TextFrame
                If .HasText Then
                    .TextRange.Font.Name = "Arial"
                    .TextRange.Font.Size = 10
                    .TextRange.Bold = True
                End If
            End With
        End If
    Next shp
End Sub

```

CD-ROM Beginn

Sie finden den Code in der Beispieldatei *Bsp06_05_Graf.doc* auf der CD-ROM zum Buch im Ordner *\Beilagen\Kap06_Grafiken\Bsp*.

CD-ROM Ende
