

## Kapitel IV

# Allerlei in Kopf- und Fußzeilen

### In diesem Kapitel:

Seitennummer eintragen	3
Firmenlogo einfügen	5
Wasserzeichen einfügen	8
Falz- und Lochmarke setzen	12
Absenderadresse eintragen	15
Dateiname und Dokumentdatum setzen	17
Zusammenfassung	21

Dieses Kapitel enthält Lösungsvorschläge zu regelmäßig auftauchenden Fragen im Zusammenhang mit Kopf- und Fußzeilen. Die Lösungen bauen auf den nachstehenden Objekten, Eigenschaften und Methoden auf.

---

**Im Blickpunkt:****Word-Objekte**

Application.CentimetersToPoints, Application.PointsToCentimeters  
Documents.Add  
Fields.Add  
HeaderFooter  
Shape.Delete, Shape.LockAnchor, Shape.RelativeHorizontalPosition, Shape.Rotation,  
Shape.ZOrder  
Shapes.AddLine, Shapes.AddPicture, Shapes.AddTextBox, Shapes.AddTextEffect  
Table.Add, Table.Borders, Table.Shading, Table.Style, Table.Condition  
Range.Collapse, Range.InsertAfter, Range.InsertBefore, Range.NextStoryRange,  
Range.ParagraphFormat, Range.Text

---

Bei den Kopf- und Fußzeilen handelt es sich um Bereiche eines Dokuments, deren Informationen auf jeder Seite des Dokuments ausgegeben werden.

Es werden Lösungsbeispiele zum Einfügen von Seitennummern, Firmenlogos, Wasserzeichen sowie Falz- und Lochmarken in der Kopfzeile aufgeführt. Weitere Beispiele behandeln das Eintragen von Absenderadresse, Dateiname und ähnlichen Informationen in die Fußzeile.

**HINWEIS**

Die Beispieldateien für dieses Kapitel sind gleich aufgebaut. Jede Datei enthält mindestens zwei VBA-Module. Davon trägt eines jeweils die Bezeichnung `vbmDemo` und beinhaltet, wie in Listing IV.1 dargestellt, die Programmsequenz, um ein neues Dokument zu erzeugen und dieses – soweit nötig – vorzubereiten.

Ein zweites VBA-Modul beinhaltet jeweils die Programmzeilen, die zum eigentlichen Beispiel gehören. Nur diese Prozedur wird im Beispiel erläutert.

Eventuelle weitere VBA-Module beinhalten in erster Linie Hilfsprozeduren, die für das Makro benötigt, aber nicht speziell erläutert werden.

---

**Listing IV.1**

Einstiegsprozedur für das Programmbeispiel in der Datei *Bsp/IV\_01.dot*

```
Sub Demo_Bsp23_01()  
    Dim docDemo As Word.Document  
  
    'Neues Dokument erstellen  
    Set docDemo = Documents.Add(Template:=ThisDocument.FullName)  
  
    'Kopfzeile mit Seitennummer einfügen
```

Listing IV.1 Einstiegsprozedur für das Programmbeispiel in der Datei *BsplIV\_01.dot* (Fortsetzung)

```

procKopfzeileSeitennummerEinfügen _
  doc:=docDemo, _
  intKopfzeile:=wdHeaderFooterPrimary, _
  intVariante:=eSeitenNrKeineXvonY

  Set docDemo = Nothing
End Sub

```

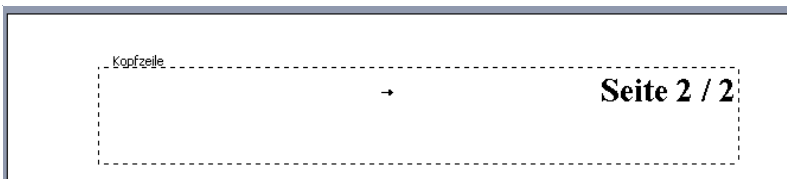
**HINWEIS** Für die nachstehenden Beispiele wurden die zugehörigen Formatvorlagen der Kopf- und Fußzeile auf die gewünschte Darstellung hin formatiert. Aus diesem Grunde konnte auf Formatierungsanweisungen innerhalb des Codes verzichtet werden.

## Seitennummer eintragen

In den meisten Dokumenten werden auch die Seitenzahlen ausgegeben, und zwar sinnvollerweise in der Kopf- oder Fußzeile. Im folgenden Beispiel wird die Seitennummer oben rechts im Dokument eingetragen.

Damit die Prozedur möglichst viele Anforderungen abdeckt, soll das Format der Seitennummer frei definiert werden können. Dies gilt für das Präfix und das Trennzeichen sowie die eingefügten Seitenzahlen. Zudem sollen sie mit (vgl. Abbildung IV.1) oder ohne Angabe der Gesamtseitenzahl dargestellt werden können.

Abbildg. IV.1 Eingefügte Seitenzahl in der Kopfzeile des Dokuments



In Listing IV.2 sind die Anforderungen an eine allgemein gültige Prozedur umgesetzt. Diese verfügt über fünf Argumente, die kurz vorgestellt werden.

Tabelle IV.1 Eingabeargumente der Prozedur *procKopfzeileSeitennummerEinfügen*

Argument	Bedeutung
doc	Objektvariable auf das zu bearbeitende Dokument.
intKopfzeile	Einen Wert aus der <b>WdHeaderFooterIndex</b> -Enumeration. Legt fest, welche Kopfzeile bearbeitet werden soll.
intVariante	Ein Wert aus der <b>EVarianteSeitennummer</b> -Enumeration. Legt das Darstellungsformat der Seitennummer fest. <b>eSeitenNrKeine</b> – keine Seitennummer wird eingefügt <b>eSeitenNrX</b> – nur die Seitennummer wird eingefügt <b>eSeitenNrXvonY</b> – die Seitennummer und die Gesamtanzahlseiten wird eingefügt

**Tabelle IV.1** Eingabeargumente der Prozedur *procKopfzeileSeitennummerEinfügen* (Fortsetzung)

Argument	Bedeutung
strPräfix	Optional. Präfix vor der eigentlichen Seitennummer.
strTrennzeichen	Optional. Trennzeichen zwischen den beiden Feldern.

**Listing IV.2** Prozedur zum Einfügen der Seitennummer mit frei definierbarem Format

```

Public Enum EVarianteSeitennummer
    eSeitenNrKeine = 0
    eSeitenNrX = 1
    eSeitenNrXvonY = 2
End Enum

Public Sub procKopfzeileSeitennummerEinfügen( _
    ByVal doc As Word.Document, _
    ByVal intKopfzeile As WdHeaderFooterIndex, _
    ByVal intVariante As EVarianteSeitennummer, _
    Optional strPräfix As String = "", _
    Optional strTrennzeichen As String = " / ")

    Const FORMAT As String = "\# ""#,##0"" \* Arabic"

    Dim rng As Word.Range

    'Kopfzeilen-Bereich als Range festlegen.
    Set rng = doc.Sections(1).Headers(intKopfzeile).Range

    'Inhalt der Kopfzeile sicherheitshalber löschen
    'und Formatvorlage »Kopfzeile« zuweisen
    With rng
        .Delete
        .Style = wdStyleHeader
    End With

    'Seitennummer in der Kopfzeile setzen.
    'Feld »AnzahlSeiten«
    If intVariante = eSeitenNrXvonY Then
        .Fields.Add Range:=rng, Type:=wdFieldNumPages, _
            Text:=FORMAT, PreserveFormatting:=True
        .Collapse (wdCollapseStart)
        .InsertBefore (strTrennzeichen)
    End If
    'Feld »Seiten«
    If Not intVariante = eSeitenNrKeine Then
        .Collapse (wdCollapseStart)
        .Fields.Add Range:=rng, Type:=wdFieldPage, _
            Text:=FORMAT, PreserveFormatting:=True
        .Collapse (wdCollapseStart)
        .InsertBefore (strPräfix)
    End If
    .Collapse (wdCollapseStart)
    .InsertBefore (vbTab)
End With
End Sub

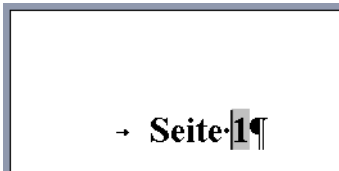
```

Die Programmzeilen innerhalb der Prozedur müssen nicht einzeln erläutert werden. Speziell hingegen ist der eigentliche Aufbau der Kopfzeile, die bewusst von rechts nach links aufgebaut wird.

Der Grund dazu ist das Range-Objekt, das in diesem Fall ganz anders verwaltet werden kann, als wenn der Aufbau von links nach rechts erfolgen würde.

Würde der Aufbau, wie es eigentlich logisch wäre, von links gestartet, so müsste nach dem Einfügen einer Feldfunktion das Range-Objekt neu definiert werden, da das Feld außerhalb des Bereichs hinzugefügt und das Kollabieren zum Endpunkt dieses Feld noch nicht beinhalten würde (vgl. Abbildung IV.2).

Abbildg. IV.2 Die Einfügemarke steht vor der Feldfunktion.



Listing IV.3 Probleme entstehen, wenn das *Range*-Objekt von links her aufgebaut wird.

```
Sub Test()
    Dim doc As Word.Document
    Dim rng As Word.Range
    Set doc = Documents.Add
    Set rng = doc.Paragraphs(1).Range
    rng.InsertAfter vbTab & "Seite "
    rng.Collapse (wdCollapseEnd)
    rng.Fields.Add rng, wdFieldNumPages
    rng.Collapse (wdCollapseEnd)
    rng.Select
End Sub
```

Wird die Kopfzeile jedoch von rechts nach links aufgebaut, so kann dieses Problem elegant umgangen werden. Das Range-Objekt muss nie neu definiert werden, und die gewünschte Darstellung der Seitennummerierung kann mit wenigen Programmzeilen umgesetzt werden.



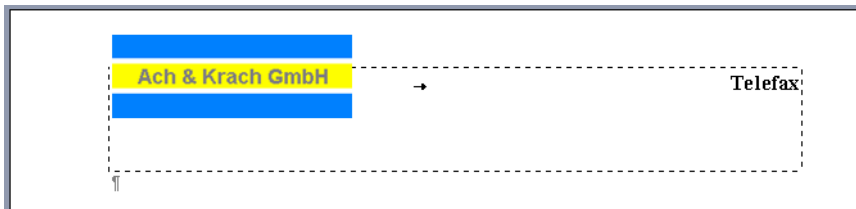
Die Prozeduren in obigem Abschnitt finden Sie in der Beispieldatei *Bsp/IV\_01.dot*. Diese befindet sich auf der CD-ROM zum Buch im Ordner *\Beispiele\KapIV*.

## Firmenlogo einfügen

Wurde früher das offizielle Firmenpapier mit dem Logo und der Geschäftsadresse von der Druckerei hergestellt, so kommt heute oft nur noch weißes Papier zum Einsatz. Das Logo und die Adresse werden direkt während des Ausdrucks erzeugt. Ein weiterer Grund für diese Vorgehensweise kann das Erstellen von *.pdf*-Dateien für die Veröffentlichung im Internet oder für den Versand via E-Mail sein. In beiden Fällen möchte der Ersteller, dass die Vorgaben des Corporate Design auch beim Empfänger eingehalten werden, und dazu gehört auch das Logo.

In diesem Beispiel wird das Logo bewusst nur auf der ersten Seite des Dokuments eingetragen. Damit die Prozedur flexibel eingesetzt werden kann, sollen die Grafikdatei und deren Position frei definiert werden können. Zusätzlich kann ein optionaler Wert als Argument übergeben werden, welcher als Dokumenttyp in der rechten oberen Ecke des Dokuments ausgegeben wird.

**Abbildg. IV.3** Eingefügtes Firmenlogo in der Kopfzeile des Dokuments



In Listing IV.4 sind die entsprechenden Anforderungen an eine solche allgemeingültige Programmsequenz umgesetzt. Die Prozedur verfügt über fünf Argumente, die folgende Bedeutung haben.

**Tabelle IV.2** Eingabeargumente der Prozedur *procKopfzeileLogoEinfügen*

Argument	Bedeutung
doc	Objektvariable auf das zu bearbeitende Dokument
strLogoDatei	Dateiname der Logodatei. Dieser sollte jeweils eindeutig, am besten mit einer absoluten Pfadangabe, übergeben werden.
sngLogoPositionOben	Abstand zwischen dem oberen Rand des Logos und der Seite (Papierrand). Die Angabe erfolgt in Zentimeter.
sngLogoPositionLinks	Abstand zwischen dem linken Rand des Logos und der Seite (Papierrand). Die Angabe erfolgt in Zentimeter.
strDokumentTyp	Optional. Bezeichnung des Dokumenttyps, die in der Kopfzeile zusätzlich aufgeführt wird.

**Listing IV.4** Prozedur zum Einfügen des Logos an einer definierbaren Stelle auf der ersten Seite des Dokuments

```
Public Sub procKopfzeileLogoEinfügen( _
    ByVal doc As Word.Document, _
    ByVal strLogoDatei As String, _
    ByVal sngLogoPositionOben As Single, _
    ByVal sngLogoPositionLinks As Single, _
    Optional strDokumentTyp As String)

    Dim hdr As Word.HeaderFooter
    Dim shp As Word.Shape

    'Kopfzeilen-Bereich als Range festlegen.
    Set hdr = doc.Sections(1).Headers(wdHeaderFooterFirstPage)

    'Inhalt der Kopfzeile sicherheitshalber löschen
    'und Formatvorlage "Kopfzeile" zuweisen
    With hdr
```

**Listing IV.4** Prozedur zum Einfügen des Logos an einer definierbaren Stelle auf der ersten Seite des Dokuments (*Fortsetzung*)

```

.Range.Delete
.Range.Style = wdStyleHeader

'Logo einfügen
If fktExistiertDatei(strLogoDatei) Then
    Set shp = .Shapes.AddPicture( _
        FileName:=strLogoDatei, Anchor:=hdr.Range, _
        LinkToFile:=False, SaveWithDocument:=True)

    With shp
        .Name = "Logo_" & Format$(Now, "yymmddhhnnss")
        .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
        .RelativeVerticalPosition = wdRelativeVerticalPositionPage
        .LockAnchor = True
        .Left = CentimetersToPoints(sngLogoPositionLinks)
        .Top = CentimetersToPoints(sngLogoPositionOben)
    End With
End If

'Dokumenttyp einfügen
If Not Len(Trim$(strDokumentTyp)) = 0 Then
    .Range.InsertBefore (strDokumentTyp)
    .Range.InsertBefore (vbTab)
End If
End With
End Sub

```

Die eigentliche Funktionsweise der Prozedur muss nicht explizit erläutert werden. Einzelne Programmzeilen wollen wir aber dennoch etwas genauer betrachten.

Das Firmenlogo wird als Shape-Objekt in das Dokument eingefügt und darin gespeichert. So steht das Logo auch dann zur Verfügung, wenn das Dokument auf einem anderen System bearbeitet wird.

Das Logo wird innerhalb der Kopfzeile hinzugefügt (`hdr.Shapes.AddPicture`) und zusätzlich mit derselben verbunden (`Anchor:=hdr.Range`).

```

Set shp = hdr.Shapes.AddPicture( _
    FileName:=strLogoDatei, Anchor:=hdr.Range, _
    LinkToFile:=False, SaveWithDocument:=True)

```

Damit das Logo zu einem späteren Zeitpunkt gezielt bearbeitet werden kann, wird es mit einem Namen versehen. Der eigentliche Namen wird aus einem Präfix (Logo\_) und einer laufenden Nummer zusammengesetzt, welche vom aktuellen Datum und der momentanen Uhrzeit abgeleitet wird. So ist sichergestellt, dass das Shape-Objekt innerhalb der Shapes-Auflistung gefunden wird und der Name garantiert eindeutig ist:

```

shp.Name = "Logo_" & Format$(Now, "yymmddhhnnss")

```

Bevor ein Shape-Objekt im Dokument positioniert werden kann, muss unbedingt festgelegt werden, von welcher Position her gemessen wird. In diesem Fall wird von der Seite, also von der Kante des Papiers, gemessen:

```
shp.RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
shp.RelativeVerticalPosition = wdRelativeVerticalPositionPage
```

Im Weiteren sollte jedes Shape-Objekt, nachdem es mit einem bestimmten Absatz verbunden wurde, zusätzlich verankert werden, damit die definierte Verbindung nicht unabsichtlich verändert wird:

```
shp.LockAnchor = True
```



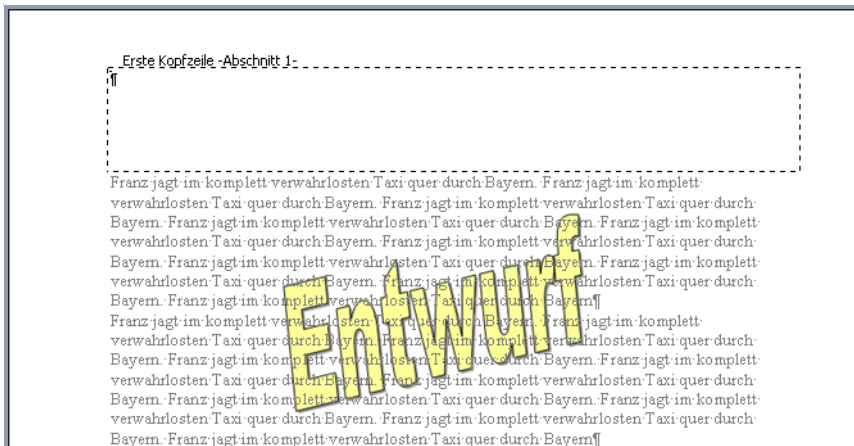
Die Prozeduren in obigem Abschnitt finden Sie in der Beispieldatei *BspIV\_02.dot*. Diese befindet sich auf der CD-ROM zum Buch im Ordner *\Beispiele\KapIV*.

## Wasserzeichen einfügen

Soll ein Dokument klassifiziert werden, eignen sich für diesen Fall so genannte Wasserzeichen ganz besonders. Das Wasserzeichen wird normalerweise in der Kopfzeile eingefügt und in der Zeichnungsebene hinter den Text gelegt. Das Einfügen innerhalb der Kopfzeilen stellt in erster Linie sicher, dass das Wasserzeichen auf allen Seiten des Dokuments sichtbar ist. Die eigentliche Position der Grafik kann auch außerhalb des Kopfzeilenbereichs liegen – beispielsweise auf der Seite zentriert, wie in Abbildung IV.4 dargestellt.

Damit die Prozedur für unterschiedliche Wasserzeichen genutzt werden kann, soll der Dateiname der Grafik als Argument übergeben werden können. Das aktuelle Beispiel zentriert die Grafik auf dem Dokument. Es wäre jedoch ein Leichtes, die Prozedur so zu erweitern, dass die effektive Position als zusätzliches Argument übergeben werden könnte.

Abbildg. IV.4 Das eingefügte Wasserzeichen ist in der Kopfzeile hinterlegt





In Listing IV.5 sind die entsprechenden Anforderungen an eine allgemein gültige Prozedur umgesetzt. Diese verfügt über zwei Argumente, die anschließend kurz vorgestellt werden.

Das eigentliche Einfügen der Grafik im Dokument erfolgt in einer Hilfsprozedur mit der Bezeichnung *procKopfzeileWasserzeichenEinfügen\_2*. Die zugehörigen Programmzeilen sind in Listing IV.6 dargestellt. Die Prozedur ist als *Private* deklariert; so ist sichergestellt, dass kein direkter Aufruf derselben erfolgen kann.

**Tabelle IV.3** Eingabeargumente der Prozedur *procKopfzeileWasserzeichenEinfügen*

Argument	Bedeutung
Doc	Objektvariable auf das zu bearbeitende Dokument
strWasserzeichenDatei	Dateiname der Grafik, die als Wasserzeichen eingefügt wird. Dieser sollte jeweils eindeutig, am besten mit einer absoluten Pfadangabe, übergeben werden.

**HINWEIS** Damit die Funktionsweise der Prozedur in Listing IV.5 besser verständlich ist, wurde die Beispieldatei *BspIV\_03.dot* bereits entsprechend formatiert.

In der Dokumentvorlage wurde ein Abschnittswechsel eingefügt. In allen Abschnitten wurden die beiden Layouteigenschaften für die Kopf- und Fußzeilen (*Gerade/ungerade anders* und *Erste Seite anders*) aktiviert. Bei der Kopfzeile der ersten Seite des zweiten Abschnitts wurde die Verknüpfung zur vorherigen gelöst. Dies bedeutet, dass das Dokument über vier unterschiedliche Kopfzeilen verfügt. Die Hilfsprozedur wird dementsprechend viermal abgearbeitet.

**Listing IV.5** Prozedur zum Einfügen eines Wasserzeichens mit frei wählbarer Grafikdatei

```
Public Sub procKopfzeileWasserzeichenEinfügen( _
    ByVal doc As Word.Document,
    ByVal strWasserzeichenDatei As String)

    Dim rng As Word.Range
    Dim intHdr As WdHeaderFooterIndex

    If fktExistiertDatei(strWasserzeichenDatei) Then
        'Alle Kopfzeilen bearbeiten
        For intHdr = wdHeaderFooterPrimary To wdHeaderFooterEvenPages
            Set rng = doc.Sections(1).Headers(intHdr).Range
            procKopfzeileWasserzeichenEinfügen_2 _
                rng:=rng, _
                strGrafik:=strWasserzeichenDatei

            'Nächste Kopfzeile vom gleichen Typ
            While Not (rng.NextStoryRange Is Nothing)
                Set rng = rng.NextStoryRange
                procKopfzeileWasserzeichenEinfügen_2 _
                    rng:=rng, _
                    strGrafik:=strWasserzeichenDatei
            Wend
        Next intHdr
    End If
End Sub
```

Die Prozedur stellt sicher, dass alle drei möglichen Kopfzeilen bearbeitet werden. Dies erfolgt unabhängig davon, wie die Layouteigenschaften des Dokuments gesetzt wurden:

```
For intHdr = wdHeaderFooterPrimary To wdHeaderFooterEvenPages
```

Anhand der NextStoryRange-Eigenschaft kann festgestellt werden, ob für einen einzelnen Kopfzeilenbereich weitere Bereiche in einem weiteren Abschnitt vorhanden sind. Alle diese zusätzlichen Bereiche werden innerhalb einer Schleife bearbeitet. Das StoryRange-Objekt sowie die NextStoryRange-Eigenschaft wurden bereits in Kapitel 6 vorgestellt.

```
While Not (rng.NextStoryRange Is Nothing)
Set rng = rng.NextStoryRange
```

Die Hilfsprozedur aus Listing IV.6 fügt das eigentliche Wasserzeichen in der entsprechenden Kopfzeile ein und setzt die Grafik an die gewünschte Position.

**Listing IV.6** Die Hilfsprozedur *procKopfzeileWasserzeichenEinfügen\_2* fügt die Grafik ein und formatiert sie.

```
Private Sub procKopfzeileWasserzeichenEinfügen_2( _
    ByVal rng As Word.Range,
    ByVal strGrafik As String)

    Dim shp As Word.Shape

    'Wasserzeichen einfügen
    Set shp = ActiveDocument.Shapes.AddPicture( _
        FileName:=strGrafik, Anchor:=rng, _
        LinkToFile:=False, SaveWithDocument:=True)

    With shp
        .Name = "Wasserzeichen_" & CStr(Rnd())
        .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
        .RelativeVerticalPosition = wdRelativeVerticalPositionPage
        .LockAnchor = True
        .Left = (ActiveDocument.Sections(1).PageSetup.PageWidth - .Width) / 2
        .Top = (ActiveDocument.Sections(1).PageSetup.PageHeight - .Height) / 2
        .ZOrder msoSendToBack
    End With
End Sub
```

Die eigentliche Funktionsweise der Hilfsprozedur muss nicht explizit erläutert werden. Dennoch wollen wir einzelne Programmzeilen etwas genauer betrachten.

Das Wasserzeichen wird ebenso behandelt wie das Firmenlogo aus dem Abschnitt »Firmenlogo einfügen« in diesem Kapitel. Die Datei wird in das Dokument eingefügt und im Kopfzeilenbereich verbunden.

Alle Grafiken werden mit einem Namen versehen. Dieser besteht aus einem statischen Präfix und einem dynamischen Suffix. Das Suffix wird automatisch anhand der Rnd-Funktion erzeugt:

```
shp.Name = "Wasserzeichen_" & CStr(Rnd())
```

Die eigentliche Position der Grafik wird dynamisch berechnet. So ist sichergestellt, dass die Grafik unabhängig von ihrer Größe stets zentriert in das Dokument eingefügt wird:

```
shp.Left = (ActiveDocument.Sections(1).PageSetup.PageWidth - .Width) / 2
shp.Top = (ActiveDocument.Sections(1).PageSetup.PageHeight - .Height) / 2
```

### Wasserzeichen als WordArt-Objekt

Im Beispiel aus Listing IV.5 wurde eine externe Grafikdatei als Wasserzeichen in das Dokument eingefügt. Neben einer solchen Grafik könnte auch ein Wordart-Objekt als so genanntes Wasserzeichen eingesetzt werden. Ein vereinfachtes Beispiel ist in Listing IV.7 dargestellt.

Für den geneigten Leser sollte es ein Leichtes sein, die beiden Prozeduren so anzupassen, dass anstelle einer externen Grafikdatei ein Wordart-Objekt als Wasserzeichen eingetragen wird.

**Listing IV.7** Ein Wasserzeichen ohne externe Grafikdatei mittels eines WordArt-Objekts erzeugen

```
Sub Demo_WordartEinfügen()
    Dim doc As Word.Document
    Dim hdr As Word.HeaderFooter
    Dim shp As Word.Shape

    Set doc = Documents.Add
    Set hdr = doc.Sections(1).Headers(wdHeaderFooterPrimary)
    Set shp = hdr.Shapes.AddTextEffect( _
        PresetTextEffect:=msoTextEffect1, _
        Text:="Entwurf", _
        FontName:="Arial", FontSize:=40, FontBold:=True, FontItalic:=True, _
        Left:=100, Top:=100, _
        Anchor:=hdr.Range)

    With shp
        .Name = "Wasserzeichen"
        .LockAnchor = True
        .Rotation = -30
        .ZOrder msoSendToBack
    End With
End Sub
```

### Wasserzeichen entfernen

Wird die Klassifikation des Dokuments geändert, muss das Wasserzeichen entsprechend angepasst werden. Dies bedeutet, dass zuerst das alte Wasserzeichen entfernt und anschließend bei Bedarf ein neues eingefügt wird.

In Listing IV.8 ist eine Prozedur aufgeführt, die alle Shape-Objekte, deren Namen mit dem Präfix *Wasserzeichen\_* beginnen, entfernt.

**Listing IV.8** Prozedur zum Entfernen der eingefügten Wasserzeichen

```
Public Sub vbmKopfzeileWasserzeichenEntfernen()
    Dim hdr As Word.HeaderFooter
    Dim shp As Word.Shape

    Set hdr = ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary)
```

**Listing IV.8** Prozedur zum Entfernen der eingefügten Wasserzeichen (Fortsetzung)

```

For Each shp In hdr.Shapes
    If Not InStr(1, shp.Name, "Wasserzeichen_", vbTextCompare) = 0 Then
        shp.Delete
    End If
Next shp
End Sub

```

Die einzelnen Shape-Objekte, die sich innerhalb der Kopfzeile befinden, werden mit einer Schleife bearbeitet. Enthält der Name des Objekts die Zeichenkette *Wasserzeichen\_*, wird dieses entfernt.

**HINWEIS**

Word verwaltet die Shape-Objekte, die sich innerhalb der Kopfzeilen befinden, auf eine ganz spezielle Art. So reicht eine einfache Schleife innerhalb irgendeines Kopfzeilenbereiches aus, um alle Shape-Objekte in *allen* unterschiedlichen Kopfzeilenbereichen unabhängig vom entsprechenden Abschnitt zu bearbeiten.

Aus diesem Grunde können die Wasserzeichen, wie in Listing IV.8 dargestellt, bedeutend einfacher entfernt werden, als diese vorher eingefügt wurden.



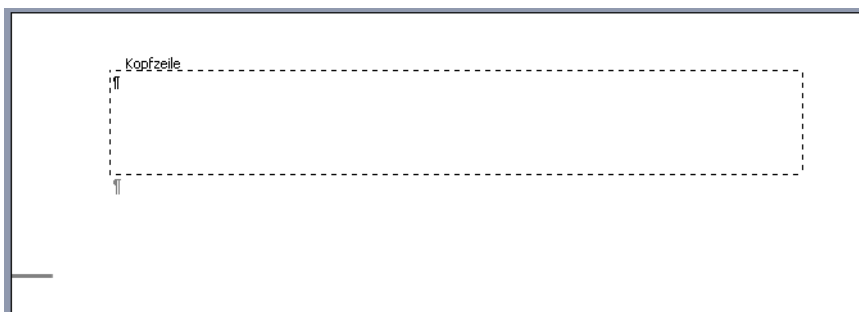
Die Prozeduren in obigem Abschnitt finden Sie in der Beispieldatei *BspIV\_03.dot*. Diese befindet sich auf der CD-ROM zum Buch im Ordner *\Beispiele\KapIV*.

## Falz- und Lochmarke setzen

Spezielle Hilfslinien im Dokument können den Anwender bei seiner täglichen Arbeit unterstützen. Zu solchen Hilfslinien können die Loch- bzw. Falzmarken am linken Rand des Dokuments gezählt werden.

Die Prozedur soll für beide Arten von Hilfslinien eingesetzt werden können. Die eigentliche Position der Lochmarke wird berechnet. Diese Linie sollte sich immer in der Mitte des Blattes befinden, unabhängig vom Format und der Ausrichtung.

Die Position der beiden Falzmarken wird fix auf ein Maß von 10,5 cm bzw. 21,0 cm festgelegt. Dies entspricht den Maßen aus der DIN 676. So passt das Dokument in einen Briefumschlag vom Format C5/6.

**Abbildg. IV.5** Eingefügte Falzmarke in der Kopfzeile des Dokuments


In Listing IV.9 wurde eine entsprechende Prozedur erstellt, die den gestellten Anforderungen gerecht wird. Sie verfügt über zwei Eingabeargumente, die kurz zusammengefasst werden.

**Tabelle IV.4** Eingabeargumente der Prozedur *procKopfzeileFalzLochmarkeEinfügen*

Argument	Bedeutung
doc	Objektvariable auf das zu bearbeitende Dokument
inhHilfslinie	Ein Wert aus der <b>EVarianteHilfslinien</b> -Enumeration. Legt die gewünschte Hilfslinie, die eingefügt wird, fest. <b>eHilfslinieLochmarke</b> – eine Lochmarke wird eingefügt <b>eHilfslinieFalzmarke</b> – die beiden Falzmarken werden eingefügt <b>eHilfslinieFalzUndLochmarke</b> – Loch- und Falzmarken werden eingefügt

Das eigentliche Einfügen der Hilfslinie wurde in eine separate Prozedur ausgelagert, diese ist in Listing IV.10 aufgeführt.

**Listing IV.9** Prozedur zum Einfügen der Loch- bzw. Falzmarke in das entsprechende Dokument

```

Const sngFALZMARKE_OBEN As Single = 10.5
Const sngFALZMARKE_UNTEN As Single = 21#

Public Enum EVarianteHilfslinien
    eHilfslinieLochmarke = 1
    eHilfslinieFalzmarke = 2
    eHilfslinieFalzUndLochmarke = 3
End Enum

Public Sub procKopfzeileFalzLochmarkeEinfügen( _
    ByVal doc As Word.Document, _
    ByVal inhHilfslinie As EVarianteHilfslinien)

    'Position der Marke bestimmen
    Select Case inhHilfslinie
        Case eHilfslinieFalzmarke
            subHilfslinieEinfügen docA:=doc, sngPositionTop:=sngFALZMARKE_OBEN
            subHilfslinieEinfügen docA:=doc, sngPositionTop:=sngFALZMARKE_UNTEN
        Case eHilfslinieLochmarke
            subHilfslinieEinfügen docA:=doc, sngPositionTop:= _
                PointsToCentimeters(doc.Sections(1).PageSetup.PageHeight / 2)
        Case eHilfslinieFalzUndLochmarke
            subHilfslinieEinfügen docA:=doc, sngPositionTop:=sngFALZMARKE_OBEN
            subHilfslinieEinfügen docA:=doc, sngPositionTop:=sngFALZMARKE_UNTEN
            subHilfslinieEinfügen docA:=doc, sngPositionTop:= _
                PointsToCentimeters(doc.Sections(1).PageSetup.PageHeight / 2)
    End Select
End Sub

```

Die Funktionsweise der Prozedur bedarf keiner Erklärung. In einem ersten Schritt werden die übertragenden Argumente ausgewertet und es erfolgen die Aufrufe der Hilfsprozedur, welche die eigentliche Marke in die Kopfzeile des Dokuments einfügt.

**Listing IV.10** Hilfsprozedur zum Zeichnen der Hilfslinie im Dokument

```

Private Sub subHilfslinieEinfügen( _
    ByVal docA As Word.Document, _
    ByVal sngPositionTop As Single)

    Dim hdr As Word.HeaderFooter
    Dim shp As Word.Shape
    Dim intHdr As WdHeaderFooterIndex

    'Kopfzeile bestimmen
    If docA.Sections(1).PageSetup.DifferentFirstPageHeaderFooter Then
        intHdr = wdHeaderFooterFirstPage
    Else
        intHdr = wdHeaderFooterPrimary
    End If
    Set hdr = docA.Sections(1).Headers(intHdr)

    'Loch- bzw. Falzmarke erstellen und formatieren
    Set shp = hdr.Shapes.AddLine(1, 1, 1, 1, hdr.Range)
    With shp
        With .Line
            .Weight = 0.25
            .Style = msoLineSingle
            .DashStyle = msoLineSolid
            .ForeColor.RGB = RGB(128, 128, 128)
            .BeginArrowheadStyle = msoArrowheadNone
            .EndArrowheadStyle = msoArrowheadNone
        End With
        .Height = CentimetersToPoints(0)
        .Width = CentimetersToPoints(0.8)
        .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
        .RelativeVerticalPosition = wdRelativeVerticalPositionPage
        .Left = CentimetersToPoints(0)
        .Top = CentimetersToPoints(sngPositionTop)
        .LockAnchor = True
    End With
End Sub

```

Die Funktionsweise der Hilfsprozedur bedarf ebenfalls keiner detaillierten Erklärung. Nachdem in einem ersten Schritt die entsprechende Kopfzeile festgelegt wurde, wird ein Zeichenobjekt (Linie) in das Dokument eingefügt.

Als Startposition werden irgendwelche Werte eingegeben, die zu einem späteren Zeitpunkt korrigiert werden, sobald die Eigenschaften `RelativeHorizontalPosition` und `RelativeVerticalPosition` gesetzt wurden:

```
Set shp = hdr.Shapes.AddLine(1, 1, 1, 1, hdr.Range)
```

Damit die Hilfslinie im Dokument diskret dargestellt wird, wurde sie statt schwarz in einem Grauton formatiert:

```
shp.ForeColor.RGB = RGB(128, 128, 128)
```

**HINWEIS**

Werden Zeichenelemente mittels einer VBA-Prozedur auf ein Dokument übertragen, sollten unbedingt *alle* Eigenschaften des betreffenden Objekts bearbeitet werden.

Der Grund dazu ist nahe liegend. Für alle Eigenschaften, die nicht bearbeitet werden, wird der so genannte Standardwert verwendet. Welchen Wert eine einzelne Eigenschaft zum Zeitpunkt der Bearbeitung aufweist, kann auf jeder Arbeitsstation unterschiedlich sein.

Der Anwender hat sogar die Möglichkeit, ein Objekt nach seinem Gutdünken zu formatieren und diese Formate als Standardeigenschaften einzutragen. Dazu muss im Kontextmenü (rechte Maustaste betätigen) des Zeichenobjekts der Befehl *Als Standard für AutoForm festlegen* gewählt werden.



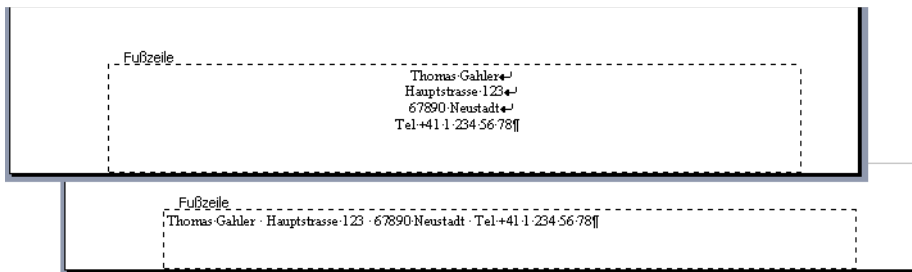
Die Prozedur in obigem Abschnitt finden Sie in der Beispieldatei *BspIV\_04.dot*. Diese befindet sich auf der CD-ROM zum Buch im Ordner *\Beispiele\KapIV*.

## Absenderadresse eintragen

Die Fußzeile ist ein geeigneter Bereich, um allgemeine Absenderangaben im Dokument zu platzieren. In diesem Beispiel kann eine Adresse, bestehend aus einer unbegrenzten Anzahl von Einzelelementen, eingefügt werden.

Damit die Prozedur möglichst viele Anforderungen abdeckt, soll das Trennzeichen zwischen den Elementen und der Adresse sowie die Ausrichtung anhand von Argumenten definiert werden können (vgl. Abbildung IV.6).

Abbildg. IV.6 Eingefügte Absenderadresse in der Fußzeile des Dokuments



In Listing IV.11 sind die Anforderungen an eine allgemeingültige Prozedur umgesetzt. Diese verfügt über vier Argumente, die kurz vorgestellt werden.

Tabelle IV.5 Eingabeargumente der Prozedur *procFusszeileAdresseEinfügen*

Argument	Bedeutung
doc	Objektvariable auf das zu bearbeitende Dokument
strAdresse()	Ein Datenfeld, das die einzelnen Datenwerte für die Absenderadresse enthält

**Tabelle IV.5** Eingabeargumente der Prozedur *procFusszeileAdresseEinfügen* (Fortsetzung)

Argument	Bedeutung
intTrennzeichen	Ein Wert aus der <b>EVarianteTrennzeichen</b> -Enumeration. Legt das verwendete Trennzeichen zwischen den einzelnen Datenelementen der Absenderadresse fest. <b>eTrennzeichenLF</b> – fügt eine Zeilenschaltung ein <b>eTrennzeichenCR</b> – für eine Absatzmarke ein <b>eTrennzeichenPunkt</b> – fügt einen Punkt ein
intAusrichtung	Ein Wert aus der <b>wdParagraphAlignment</b> -Enumeration. Legt die Ausrichtung der Adresse in der Fußzeile fest. <b>wdAlignParagraphLeft</b> – linksbündig <b>wdAlignParagraphCenter</b> – zentriert <b>wdAlignParagraphRight</b> – rechtsbündig

**Listing IV.11** Prozedur zum Einfügen der Absenderadresse mit unterschiedlichen Darstellungsvarianten

```

Public Enum EVarianteTrennzeichen
    eTrennzeichenLF = 11
    eTrennzeichenCR = 13
    eTrennzeichenPunkt = 183
End Enum

Public Sub procFusszeileAdresseEinfügen ( _
    ByVal doc As Word.Document, _
    ByRef strAdresse() As String, _
    ByVal intTrennzeichen As EVarianteTrennzeichen, _
    ByVal intAusrichtung As WdParagraphAlignment)

    Dim rng As Word.Range
    Dim intFtr As WdHeaderFooterIndex
    Dim strText As String
    Dim strTrenner As String

    'Tatsächliches Trennzeichen bestimmen
    Select Case intTrennzeichen
        Case eTrennzeichenPunkt
            strTrenner = " " & Chr$(intTrennzeichen) & " "
        Case Else
            strTrenner = Chr$(intTrennzeichen)
    End Select

    'Absenderadresse aufbereiten
    For intFtr = LBound(strAdresse) To UBound(strAdresse)
        strText = strText & strTrenner & strAdresse(intFtr)
    Next intFtr
    strText = Mid$(strText, Len(strTrenner) + 1)

    'Alle Fußzeilen bearbeiten
    For intFtr = wdHeaderFooterPrimary To wdHeaderFooterEvenPages
        Set rng = doc.Sections(1).Footers(intFtr).Range
        rng.ParagraphFormat.Alignment = intAusrichtung
        rng.Text = strText
    Next intFtr
End Sub

```



Die Funktionsweise der Prozedur ist schnell erklärt. In einem ersten Schritt wird das Trennzeichen zwischen den Datenelementen der Absenderadresse aufbereitet. Ein zweiter Schritt ist für die Aufbereitung der eigentlichen Absenderadresse zuständig. Abschließend wird die Ausrichtung der Fußzeile festgelegt und dieser der gesamte Text zugewiesen. Dies geschieht unabhängig von den eingestellten Layouteigenschaften für alle drei Fußzeilenarten.

Der Punkt als Trennzeichen muss speziell aufbereitet werden, da diesem Zeichen ein zusätzliches Leerzeichen vor- und nachgestellt wird. Alle anderen Trennzeichen können direkt umgesetzt werden, weil innerhalb der `EVarianteTrennzeichen`-Enumeration die einzelnen Werte dem Zeichen-Code des einzusetzenden Zeichens entsprechen:

```
Case eTrennzeichenPunkt
    strTrenner = " " & Chr$(intTrennzeichen) & " "
Case Else
    strTrenner = Chr$(intTrennzeichen)
```

Die Absenderadresse wird innerhalb einer Schleife aufgebaut. Alle Elemente des Datenfelds werden in eine Zeichenkette übernommen. Zu jedem Eintrag wird das definierte Trennzeichen hinzugefügt. Da die Trennzeichen nur zwischen den einzelnen Elementen benötigt werden, wird jenes am Anfang der Zeichenkette anschließend wieder entfernt:

```
For intFtr = LBound(strAdresse) To UBound(strAdresse)
    strText = strText & strTrenner & strAdresse(intFtr)
Next intFtr
strText = Mid$(strText, Len(strTrenner) + 1)
```



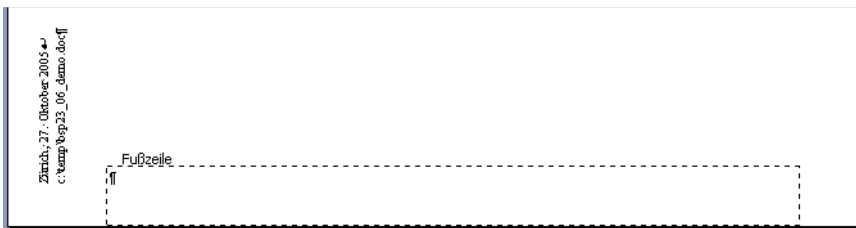
Die Prozedur in obigem Abschnitt finden Sie in der Beispieldatei *BspIV\_05.dot*. Diese befindet sich auf der CD-ROM zum Buch im Ordner *\Beispiele\KapIV*.

## Dateiname und Dokumentdatum setzen

Neben den eigentlichen Informationen, die ein Dokument enthält, werden oftmals noch zusätzliche Statusinformationen ausgegeben. Dazu gehören unter anderem der Dateiname oder ein Datum. In diesem Beispiel soll beides zusammen in der linken unteren Ecke des Dokuments eingetragen werden. Der Eintrag soll senkrecht am Papierrand erscheinen.

Damit die Prozedur für unterschiedliche Zwecke verwendet werden kann, sollen die beiden Informationen frei definiert und bei Bedarf sogar weggelassen werden können (vgl. Abbildung IV.7).

Abbildg. IV.7 Datum und Dateiname, eingefügt in einem unsichtbaren Textfeld in der Fußzeile des Dokuments



In Listing IV.12 sind die festgelegten Anforderungen an eine allgemeingültige Prozedur umgesetzt. Diese verfügt über vier Argumente.

**Tabelle IV.6** Eingabeargumente der Prozedur *procFusszeileDateinameDatumEinfügen*

Argument	Bedeutung
doc	Objektvariable auf das zu bearbeitende Dokument
intDateiname	Ein Wert aus der <b>EVarianteDateiname</b> -Enumeration. Legt fest, wie der Dateiname in dem Dokument ausgegeben wird. <b>eKeinDateiname</b> – kein Dateiname wird eingefügt <b>eNurDateiname</b> – das Feld <i>FileName</i> wird eingefügt, der Dateiname wird in dem Dokument ausgegeben. <b>eDateinameInklPfad</b> – das Feld <i>FileName</i> wird eingefügt, der Dateiname wird zusammen mit dem Speicherpfad ausgegeben.
intDatum	Ein Wert aus der <b>EVarianteDatum</b> -Enumeration. Legt fest, wie das Datum in dem Dokument ausgegeben wird. <b>eKeinDatum</b> – kein Datum wird eingefügt <b>eErstellDatum</b> – das Erstelldatum des Dokuments wird eingefügt <b>eSpeicherDatum</b> – das letzte Speicherdatum des Dokuments wird eingefügt <b>eDruckDatum</b> – das aktuelle Datum beim Ausdruck des Dokuments wird eingefügt <b>eAktuellesDatum</b> – das aktuelle Systemdatum wird als dynamisches Feld eingefügt <b>eHeuteAlsStatischesDatum</b> – das aktuelle Datum wird als statischer Text in das Dokument eingefügt
strOrtZumDatum	Optional. Zum Datum kann eine zusätzliche Bezeichnung des Ortes eingefügt werden.

**Listing IV.12** Prozedur zum Einfügen des Dateinamens und des Dokumentdatums mit definierbarem Format

```
Public Enum EVarianteDatum
    eErstellDatum = wdFieldCreateDate
    eSpeicherDatum = wdFieldSaveDate
    eDruckDatum = wdFieldPrintDate
    eAktuellesDatum = wdFieldDate
    eHeuteAlsStatischesDatum = wdFieldQuote
    eKeinDatum = 0
End Enum

Public Enum EVarianteDateiname
    eKeinDateiname = 0
    eNurDateiname = 1
    eDateinameInklPfad = 2
End Enum

Public Sub procFusszeileDateinameDatumEinfügen( _
    ByVal doc As Word.Document, _
    ByVal intDateiname As EVarianteDateiname, _
    ByVal intDatum As EVarianteDatum, _
    Optional strOrtZumDatum As String)

    Dim ftr As Word.HeaderFooter
    Dim rng As Word.Range
    Dim shp As Word.Shape
```

**Listing IV.12** Prozedur zum Einfügen des Dateinamens und des Dokumentdatums mit definierbarem Format (Fortsetzung)

```

Dim intFtr As WdHeaderFooterIndex
Dim strFeldText As String

'Alle Fußzeilen bearbeiten
For intFtr = wdHeaderFooterPrimary To wdHeaderFooterEvenPages
    Set ftr = doc.Sections(1).Footers(intFtr)

    'Textfeld für Dateiname, Datum usw. am linken Papierrand einfügen
    Set shp = ftr.Shapes.AddTextbox(Orientation:=msoTextOrientationUpward, _
        Left:=1, Top:=1, Width:=1, Height:=1, Anchor:=ftr.Range)
    With shp
        .Name = "Dateiname_" & CStr(Rnd())
        .Fill.Visible = msoFalse
        .Line.Visible = msoFalse
        With .TextFrame
            .MarginLeft = 0#
            .MarginRight = 0#
            .MarginTop = 0#
            .MarginBottom = 0#
        End With
        .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
        .RelativeVerticalPosition = wdRelativeVerticalPositionPage
        .Left = CentimetersToPoints(0.7)
        'Position in Abhängigkeit mit der Ausrichtung überarbeiten.
        If doc.Sections(1).PageSetup.Orientation = wdOrientPortrait Then
            .Top = CentimetersToPoints(11.7)
        Else
            .Top = CentimetersToPoints(3#)
        End If
        .Height = CentimetersToPoints(17#)
        .Width = CentimetersToPoints(0.8)
        .LockAnchor = True
        .WrapFormat.Type = wdWrapNone
        Set rng = .TextFrame.TextRange

    'Textfeld formatieren
    With .TextFrame.TextRange
        .ParagraphFormat.Style = wdStyleFooter
        With .Font
            .Size = 7
            .ColorIndex = wdGray25
        End With
    End With

    'Dateiname als Feld in Textfeld einfügen
    If Not intDateiname = eKeinDateiname Then
        If intDateiname = eDateinameInklPfad Then
            strFeldText = "\" & Lower & p"
        Else
            strFeldText = "\" & Lower"
        End If
        .Fields.Add Range:=rng, Type:=wdFieldFileName, _
            Text:=strFeldText, PreserveFormatting:=True
    End If

    'Zeilenschaltung einfügen

```

**Listing IV.12** Prozedur zum Einfügen des Dateinamens und des Dokumentdatums mit definierbarem Format (*Fortsetzung*)

```

        .InsertBefore (Chr$(11))
        rng.SetRange rng.Start, rng.Start

'Datum als Feld in Textfeld einfügen
    If Not intDatum = eKeinDatum Then
        If intDatum = eHeuteAlsStatischesDatum Then
            strFeldText = Chr$(34) & Format$(Now, "d. MMMM yyyy") & Chr$(34)
        Else
            strFeldText = "\@ " & Chr$(34) & "d. MMMM yyyy" & Chr$(34) & _
                " \* MERGEFORMAT"
        End If
        .Fields.Add Range:=rng, Type:=intDatum, Text:=strFeldText, _
            PreserveFormatting:=True
        'Ort zum Datum
        .InsertBefore (strOrtZumDatum)
    End If
End With
End With
Next intFtr
End Sub

```

Die Prozedur fügt als Erstes ein Textfeld in die Fußzeile des Dokuments ein. Dieses Textfeld wird formatiert und positioniert. Die Ausrichtung des Textes innerhalb des Textfeldes ist senkrecht zur Ausrichtung des Papiers. Da dem Textfeld weder eine Rahmen- noch eine Hintergrundfarbe zugewiesen wurde, bleibt es auf dem Ausdruck unsichtbar.

In einem weiteren Schritt werden die beiden Informationen, der Dateiname bzw. ein Dokumentdatum, in das Textfeld eingetragen.

Ob der Dateiname mit oder ohne Pfadangabe auf dem Dokument ausgegeben wird, wird über ein Argument des FileName-Feldes gesteuert:

```

If intDateiname = eDateinameInklPfad Then
    strFeldText = "\* Lower \p"
Else
    strFeldText = "\* Lower"
End If

```

Die Übergabe der Argumente an die Feldfunktion wird beim Hinzufügen des Field-Objektes im Text-Argument abgewickelt:

```

.Fields.Add Range:=rng, Type:=intDatum, Text:=strFeldText, PreserveFormatting:=True

```

Damit die unterschiedlichen Datumsfelder mit einigen wenigen Programmzeilen eingefügt werden konnten, wurden innerhalb der EVarianteDatum-Enumeration die einzelnen Werte mit den definierten Konstanten aus der WdFieldType-Enumeration gleichgesetzt.

Für das statische Datumsfeld wird ein Quote-Feld verwendet. Mit diesem Feld kann ein Text in ein Dokument eingefügt werden – also genau das, was mit dem statischen Datum bezweckt werden soll. Das Text-Argument dieses Feldes muss gesondert aufgebaut werden:

```

If intDatum = eHeuteAlsStatischesDatum Then
    strFeldText = Chr$(34) & Format$(Now, "d. MMMM yyyy") & Chr$(34)
Else
    strFeldText = "\" & Chr$(34) & "d. MMMM yyyy" & Chr$(34) & " \* MERGEFORMAT"
End If
.Fields.Add Range:=rng, Type:=intDatum, Text:=strFeldText, PreserveFormatting:=True

```



Die Prozedur in obigem Abschnitt finden Sie in der Beispieldatei *BspIV\_06.dot*. Diese befindet sich auf der CD-ROM zum Buch im Ordner *\Beispiele\KapIV*.

## Zusammenfassung

In diesem Kapitel wurden verschiedene Lösungsbeispiele aufgezeigt, die alle im Zusammenhang mit der Kopf- bzw. der Fußzeile stehen. Alle Beispiele können sehr schnell erweitert und an persönliche Bedürfnisse angepasst werden:

- Im Bereich der Kopfzeilen wurde gezeigt, wie man Seitenzahlen (Seite 3 ff.), Firmenlogo (Seite 5 ff.), Wasserzeichen (Seite 8 ff.) oder Falz- bzw. Lochmarken (Seite 12 ff.) einfügen kann.
- Im Bereich der Fußzeilen wurde aufgezeigt, wie man Absenderadresse (Seite 15 ff.) oder Datei-name bzw. Dokumentdatum (Seite 17 ff.) einfügen kann.

