PEARSON
Education

VBA mit Excel

Mit kleinen Makros zaubern
Lauffähig für die Versionen 97-2007

BERND HELD

Markt+Technik

→leicht →klar →sofort

Kapitel 3

Der Zeilen-/Spalten-Workshop

Nachdem Sie bereits gelernt haben, wie Sie mit Zellen in der Programmierung umgehen, lernen Sie in diesem Kapitel, was man alles mit Zeilen und Spalten anstellen kann. Dazu gehört es, Zeilen und Spalten einzufügen, zu löschen oder ein- und auszublenden. Des Weiteren können Sie in Spalten auch mit Filtern arbeiten, die es Ihnen erleichtern, umfangreiche Listen schnell auszuwerten.

Zeilen und Spalten markieren

Insgesamt stehen Ihnen in Excel 2007 pro Tabelle 1.048.576 Zeilen zur Verfügung. Diese können Sie markieren, indem Sie links oben auf den Schnittpunkt der Zeilen- und Spaltenbeschriftung klicken.

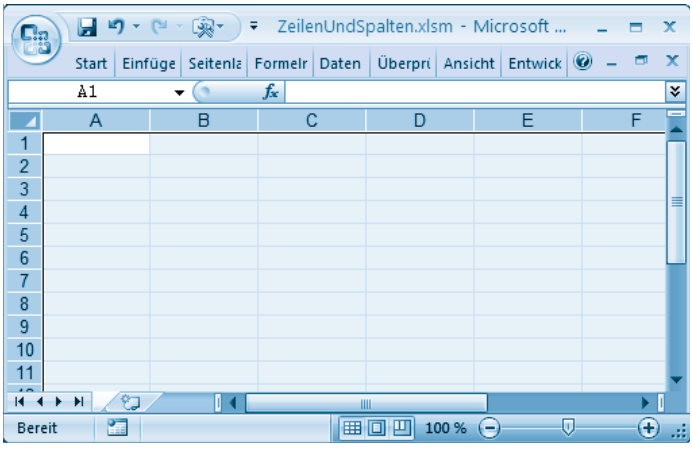
Alle Zellen markieren

Diesen manuellen Vorgang können Sie im Übrigen auch mit dem Makro aus Listing 3.1 durchführen.

```
Sub AlleZellenMarkieren()
Sheets("Tabelle1").Activate
Cells.Select
End Sub
```

Listing 3.1: Alle Zellen der Tabelle markieren

Das Objekt `Cells` enthält alle Zellen einer Tabelle. Indem Sie die Methode `Select` auf dieses Objekt anwenden, markieren Sie alle Zellen der Tabelle.



Alle Zellen sind markiert

Verwendete Zellen markieren

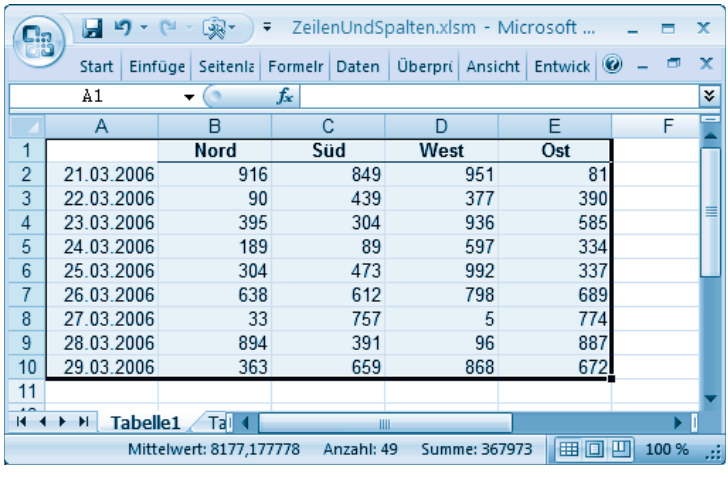
Für eine effektive Bearbeitung sollten Sie aber darauf achten, dass Sie nicht gerade alle Zellen einer Tabelle markieren, sondern besser nur die, die Sie wirklich anschließend bearbeiten möchten.

Um diese Aufgabenstellung zu lösen, setzen Sie das Makro aus Listing 3.2 ein.

```
Sub AlleBenötigtenZellenMarkieren()
Sheets("Tabelle1").Activate
ActiveSheet.UsedRange.Select
End Sub
```

Listing 3.2: Den verwendeten Bereich markieren

Über die Eigenschaft `UsedRange` erhalten Sie Auskunft darüber, welcher Bereich einer Tabelle mit Daten gefüllt ist. Mit der Methode `Select` markieren Sie diesen Bereich direkt im Anschluss.



	A	B	C	D	E	F
1		Nord	Süd	West	Ost	
2	21.03.2006	916	849	951	81	
3	22.03.2006	90	439	377	390	
4	23.03.2006	395	304	936	585	
5	24.03.2006	189	89	597	334	
6	25.03.2006	304	473	992	337	
7	26.03.2006	638	612	798	689	
8	27.03.2006	33	757	5	774	
9	28.03.2006	894	391	96	887	
10	29.03.2006	363	659	868	672	
11						

Nur die benutzten Zellen werden markiert

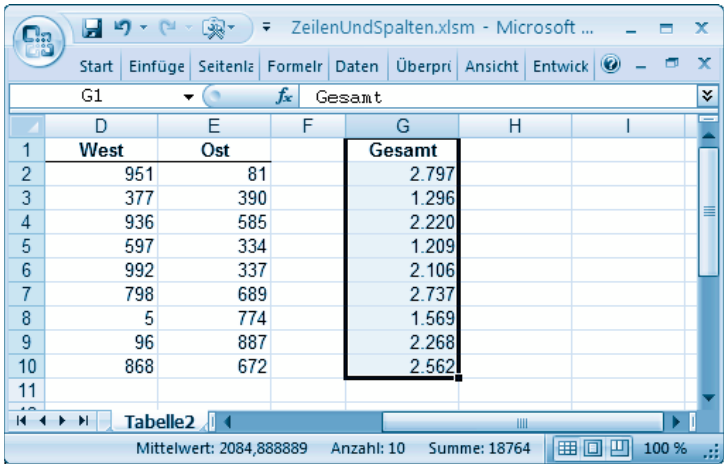
Angrenzende Zellen markieren

Im folgenden Beispiel werden, ausgehend von einer aktiven Zelle, alle umliegenden Zellen markiert. Dazu setzen Sie wie in der folgenden Abbildung den Mauszeiger auf die Zelle G4 und starten das Makro aus Listing 3.3.

```
Sub UmliegendeZellenMarkieren()
Sheets("Tabelle2").Activate
Range("G4").Select
ActiveCell.CurrentRegion.Select
End Sub
```

Listing 3.3: Die umliegenden Zellen ermitteln und markieren

Die Eigenschaft `CurrentRegion` gibt ein `Range`-Objekt zurück, das den aktuellen Bereich darstellt. Der aktuelle Bereich wird von einer beliebigen Kombination leerer Zeilen und Spalten umschlossen. Über die Methode `Select` markieren Sie diesen Bereich.



	D	E	F	G	H	I
1	West	Ost		Gesamt		
2	951	81		2.797		
3	377	390		1.296		
4	936	585		2.220		
5	597	334		1.209		
6	992	337		2.106		
7	798	689		2.737		
8	5	774		1.569		
9	96	887		2.268		
10	868	672		2.562		
11						

Umliegende Zellen markieren

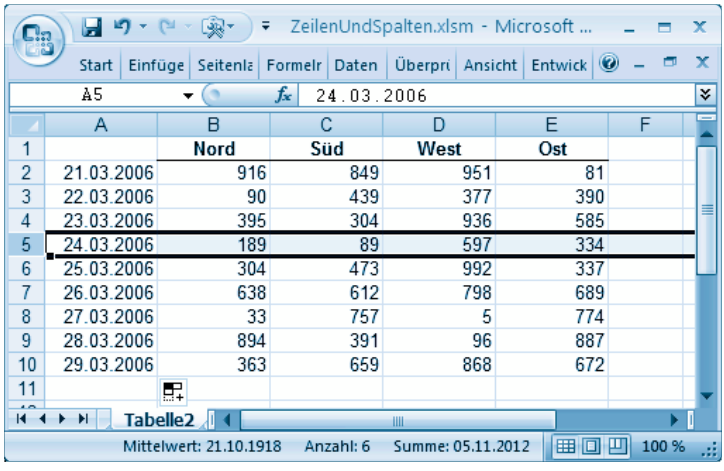
Einzelne Zeile markieren

Die Markierung einer einzelnen Zeile ist kein Problem. Eine einzelne Zeile markieren Sie, indem Sie die Eigenschaft `Rows` auf das Objekt `Range` anwenden. So markieren Sie mit dem Makro aus Listing 3.4 die fünfte Zeile in der *Tabelle2*.

```
Sub BestimmteZeileMarkieren()
 Sheets("Tabelle2").Activate
 Rows("5:5").Select
End Sub
```

Listing 3.4: Eine bestimmte Zeile markieren

Geben Sie der Eigenschaft `Rows` bekannt, welche Zeile Sie meinen, und markieren Sie diese über die Methode `Select`.



	A	B	C	D	E	F
1		Nord	Süd	West	Ost	
2	21.03.2006	916	849	951	81	
3	22.03.2006	90	439	377	390	
4	23.03.2006	395	304	936	585	
5	24.03.2006	189	89	597	334	
6	25.03.2006	304	473	992	337	
7	26.03.2006	638	612	798	689	
8	27.03.2006	33	757	5	774	
9	28.03.2006	894	391	96	887	
10	29.03.2006	363	659	868	672	
11						

Die Zeile 5 wird komplett markiert

Mehrere Zeilen auf dem Tabellenblatt markieren

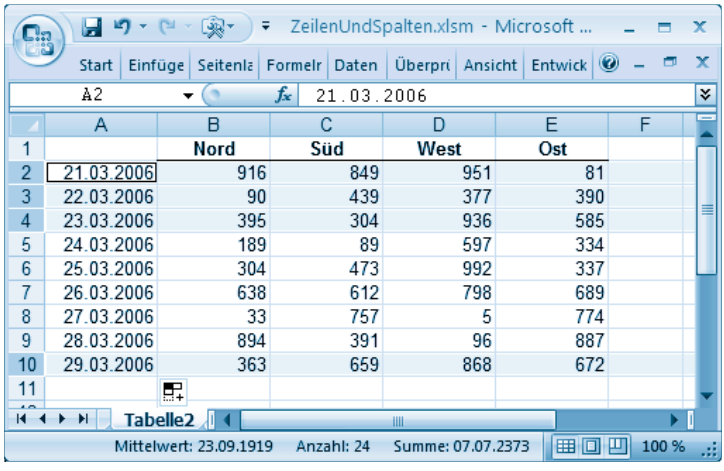
Sicher wissen Sie, dass Sie manuell mehrere Zeilen auf dem Tabellenblatt markieren können. Mit dem Einsatz der Strg-Taste im Zusammenspiel mit der linken Maustaste, mit welcher Sie auf die Zeilenköpfe klicken, markieren Sie mehrere, auch nicht zusammenliegende Zeilen. Wie funktioniert dies aber mit VBA?

Im nächsten Beispiel sollen die Zeilen 2, 3, 4 und 10 auf der *Tabelle2* markiert werden.

```
Sub MehrfachauswahlZeilen()
 Sheets("Tabelle2").Activate
 Range("2:2,3:3,4:4,10:10").Select
End Sub
```

Listing 3.5: Mehrere Zeilen markieren

Wenden Sie in diesem Fall die Eigenschaft `Range` an und übergeben Sie ihr die gewünschten Zeilen, die markiert werden sollen. Die Markierung selbst wird durch die Methode `Select` durchgeführt.



	A	B	C	D	E	F
1		Nord	Süd	West	Ost	
2	21.03.2006	916	849	951	81	
3	22.03.2006	90	439	377	390	
4	23.03.2006	395	304	936	585	
5	24.03.2006	189	89	597	334	
6	25.03.2006	304	473	992	337	
7	26.03.2006	638	612	798	689	
8	27.03.2006	33	757	5	774	
9	28.03.2006	894	391	96	887	
10	29.03.2006	363	659	868	672	
11						

Mehrere Zeilen markieren

Einzelne Spalte markieren

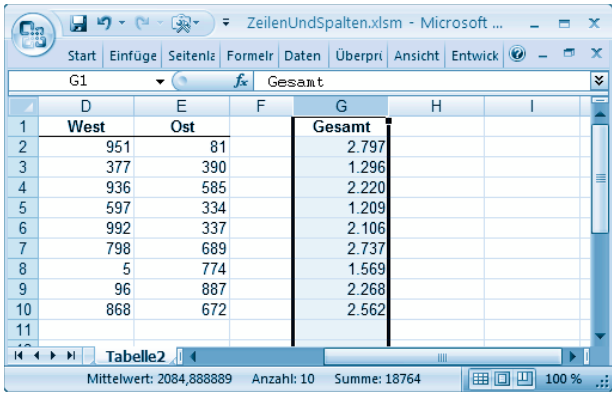
Ähnlich wie schon bei der Markierung von Zeilen funktioniert es auch bei den Spalten. Eine Spalte markieren Sie normalerweise, indem Sie mit der linken Maustaste auf die Spaltenüberschrift klicken oder die Tastenkombination Strg + Leer drücken.

Im nächsten Beispiel soll die komplette Spalte G der *Tabelle2* markiert werden. Setzen Sie zu diesem Zweck das Makro aus Listing 3.6 ein.

```
Sub EinzelneSpalteMarkieren()
 Sheets("Tabelle2").Activate
 Columns("G:G").Select
End Sub
```

Listing 3.6: Einzelne Spalte markieren

Eine einzelne Spalte markieren Sie, indem Sie die Eigenschaft `Columns` auf das `Range`-Objekt anwenden. Geben Sie bekannt, welche Spalte Sie genau bearbeiten möchten, und markieren diese über die Methode `Select`.



	D	E	F	G	H	I
1	West	Ost		Gesamt		
2	951	81		2.797		
3	377	390		1.296		
4	936	585		2.220		
5	597	334		1.209		
6	992	337		2.106		
7	798	689		2.737		
8	5	774		1.569		
9	96	887		2.268		
10	868	672		2.562		
11						

Einzelne Spalte markieren

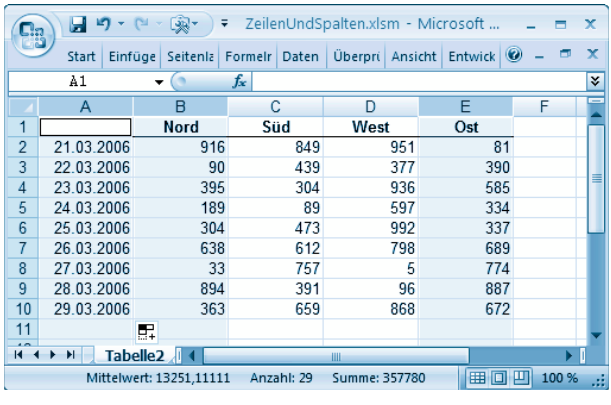
Mehrere Spalten markieren

Wenn Sie mehrere, auch nicht zusammenhängende Spalten markieren möchten, sehen Sie sich das nächste Makro in Listing 3.7 an.

```
Sub MehrereSpaltenMarkieren()
 Sheets("Tabelle2").Activate
 Range("A:A,B:B,E:E").Select
End Sub
```

Listing 3.7: Mehrere Spalten markieren

Wenden Sie in diesem Fall die Eigenschaft `Range` an und übergeben Sie ihr die gewünschten Spalten, die markiert werden sollen. Die Markierung selbst wird durch die Methode `Select` durchgeführt.



	A	B	C	D	E	F
1		Nord	Süd	West	Ost	
2	21.03.2006	916	849	951	81	
3	22.03.2006	90	439	377	390	
4	23.03.2006	395	304	936	585	
5	24.03.2006	189	89	597	334	
6	25.03.2006	304	473	992	337	
7	26.03.2006	638	612	798	689	
8	27.03.2006	33	757	5	774	
9	28.03.2006	894	391	96	887	
10	29.03.2006	363	659	868	672	
11						

Die Spalten A, B und E wurden markiert

Hinweis

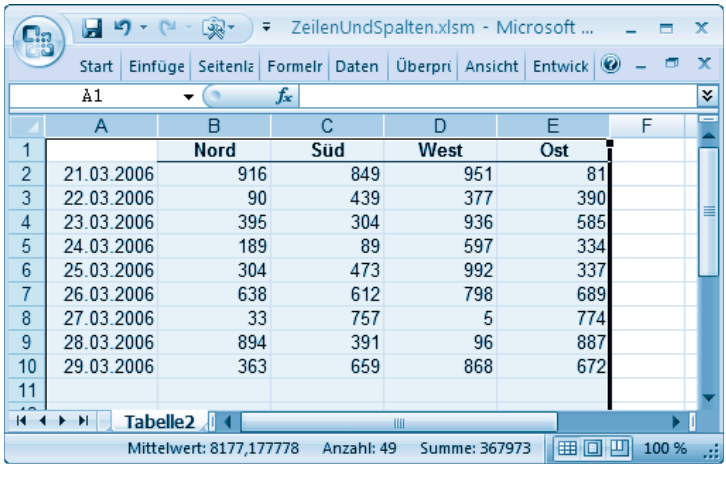
Übrigens können Sie die Spalten auch über einen Index ansteuern. Wissen müssen Sie dazu, dass die Spalte A den Index 1 und die letzte Spalte einer Tabelle den Index 16.384 hat.

Im nächsten Beispiel sollen die Spalten A bis E in der *Tabelle2* markiert werden. Das Makro für diese Aufgabe sehen Sie in Listing 3.8.

```
Sub MehrereSpaltenMarkiernIndex()
Sheets("Tabelle2").Activate
Range(Columns(1), Columns(5)).Select
End Sub
```

Listing 3.8: Mehrere Spalten über Index markieren

Wenden Sie in diesem Fall die Eigenschaft `Range` an und übergeben Sie ihr die gewünschten Spaltenindexe, die markiert werden sollen. Die Markierung selbst wird durch die Methode `Select` durchgeführt.



Mehrere Spalten über Index markieren

Zeilen und Spalten markieren

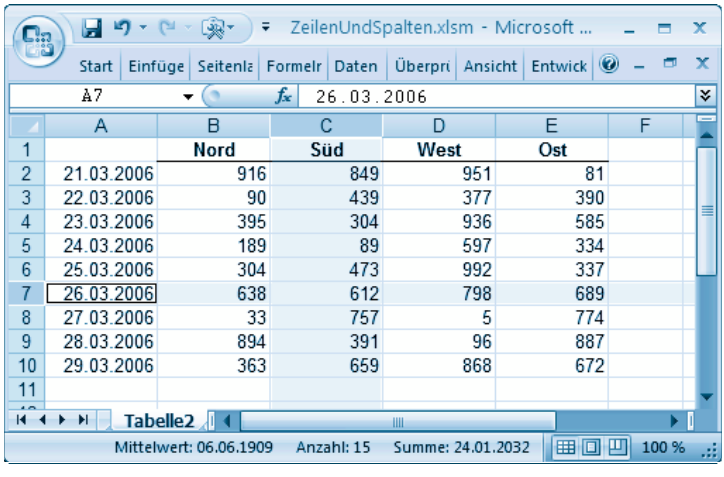
Um jeweils eine Spalte sowie eine Zeile in einem Arbeitsgang zu markieren, setzen Sie das Makro aus Listing 3.9 ein.

```
Sub BereicheMarkieren()
Dim Bereich1 As Range
Dim Bereich2 As Range
Dim Gesamtb As Range

Sheets("Tabelle2").Activate
Set Bereich1 = Rows(7)
Set Bereich2 = Columns("C:C")
Set Gesamtb = Union(Bereich1, Bereich2)
Gesamtb.Select
End Sub
```

Listing 3.9: Zeile und Spalte markieren

Legen Sie mit Hilfe der Anweisung `Set` erst einmal die Bereiche fest, die Sie gemeinsam ansprechen möchten. Danach fassen Sie die einzelnen Bereiche über die Methode `Union` zusammen. Dann markieren Sie den Gesamtbereich durch die Methode `Select`.



	A	B	C	D	E	F
1		Nord	Süd	West	Ost	
2	21.03.2006	916	849	951	81	
3	22.03.2006	90	439	377	390	
4	23.03.2006	395	304	936	585	
5	24.03.2006	189	89	597	334	
6	25.03.2006	304	473	992	337	
7	26.03.2006	638	612	798	689	
8	27.03.2006	33	757	5	774	
9	28.03.2006	894	391	96	887	
10	29.03.2006	363	659	868	672	
11						

Spalte und Zeile markieren

Anzahl der verwendeten Zeilen ermitteln

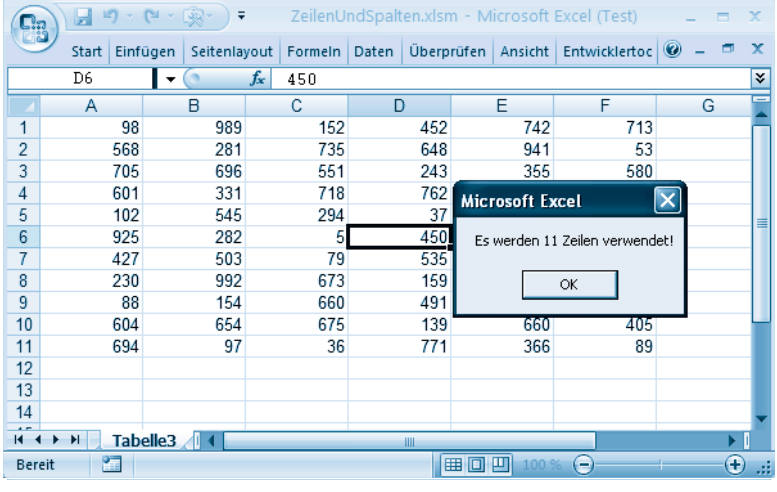
Die Anzahl der verwendeten Zeilen auf einem Tabellenblatt können Sie ermitteln, indem Sie die Eigenschaft `UsedRange` einsetzen, um den verwendeten Bereich auf dem Tabellenblatt herauszubekommen, sowie die Eigenschaften `Rows` und `Count`, um die Zeilen im verwendeten Bereich zu zählen.

```
Sub AnzahlVerwendeteZeilen()
Dim lngZeilen As Long

 Sheets("Tabelle3").Activate
 lngZeilen = ActiveSheet.UsedRange.Rows.Count
 MsgBox "Es werden " & lngZeilen & " Zeilen verwendet!"
End Sub
```

Listing 3.10: Anzahl der verwendeten Zeilen ermitteln

> **Achtung**
>
> Achten Sie darauf, dass Sie bei der Programmierung in großen Tabellen den Datentyp **Long** als »Zeilenvariable« einsetzen. Der Datentyp **Integer** streckt seine Flügel bei 32.767 Zeilen.



Die verwendeten Zeilen in der Tabelle zählen

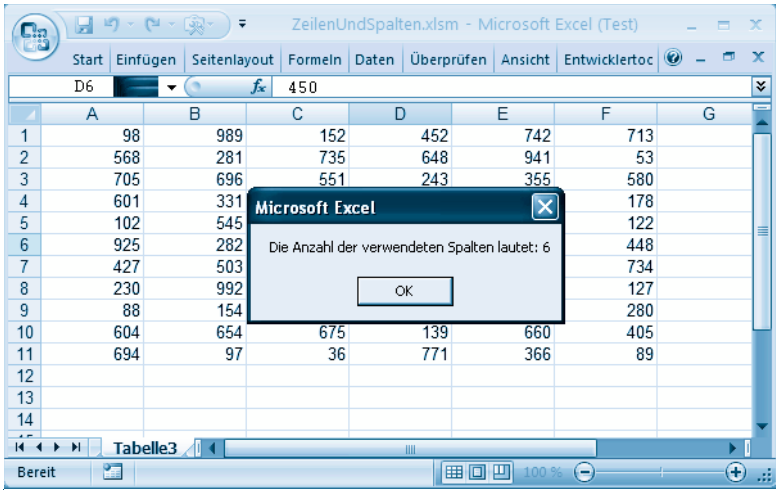
Anzahl der verwendeten Spalten ermitteln

Bei der Ermittlung der Anzahl der belegten Spalten brauchen Sie sich um die Variablendefinition keine Sorgen zu machen. Da es in einem Excel-Tabellenblatt 16.384 Spalten gibt, reicht eine Variable vom Typ `Integer` noch aus.

```
Sub AnzahlVerwendeteSpalten()
Dim intSpalten As Integer

 Sheets("Tabelle3").Activate
 intSpalten = ActiveSheet.UsedRange.Columns.Count
 MsgBox  _
"Die Anzahl der verwendeten Spalten lautet: " & intSpalten
End Sub
```

Listing 3.11: Verwendete Spalten zählen



Die Anzahl der verwendeten Spalten ermitteln

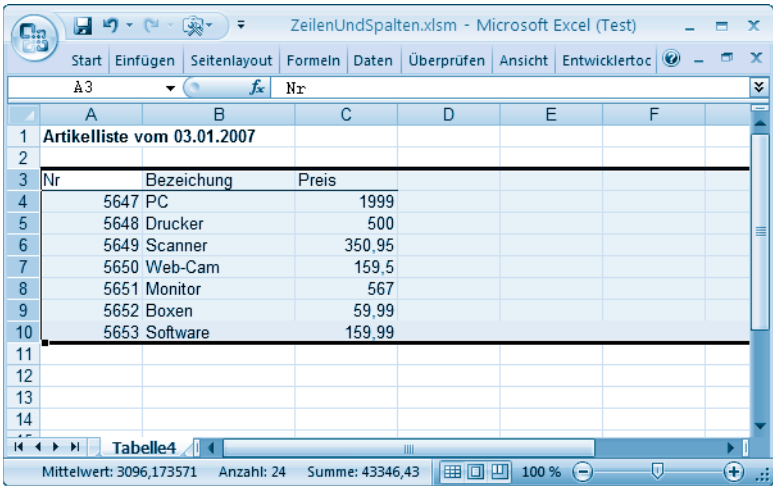
Markieren von Zeilen ab bestimmter Position

Stellen Sie sich vor, Sie haben eine Tabelle, bei der die relevanten Daten erst ab Zeile 3 beginnen. Ihre Aufgabe besteht nun darin, alle Daten ab der Zeile 3 abwärts zu markieren. Das Makro für diese Aufgabe lautet:

```
Sub MarkierenAbZeile3()
 Sheets("Tabelle4").Activate
 Range(Range("A3"), _
 Range("A3").End(xlDown)).EntireRow.Select
End Sub
```

Listing 3.12: Zeilen markieren ab Zeile 3

Markieren Sie den datenrelevanten Bereich mit Hilfe der Methode `Select`, indem Sie im ersten Argument die Startadresse der Zelle angeben und für das zweite Argument die Eigenschaft `End` mit der Richtungskonstanten `xlDown` einsetzen. Über die Eigenschaft `EntireRow` markieren Sie diese Zeilen komplett.



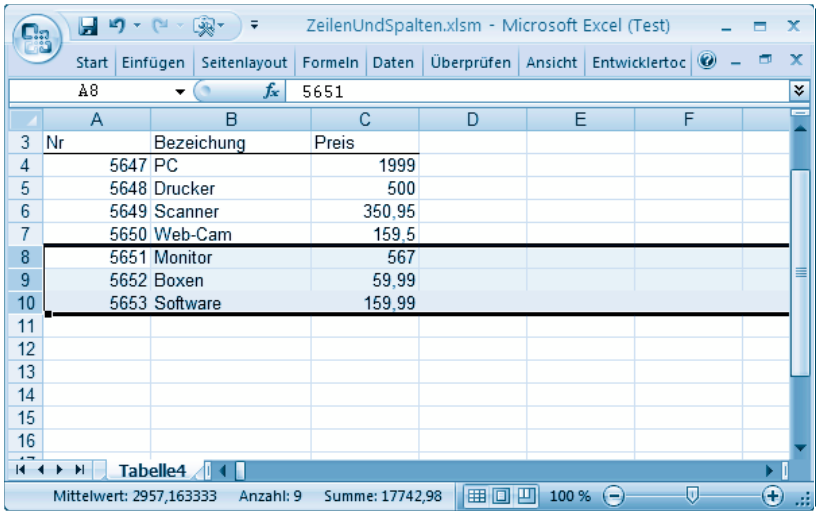
Zeilen markieren ab bestimmter Zeile

Ähnlich gelagert ist auch das folgende Beispiel. In einer Tabelle sollen lediglich die letzten drei Zeilen eines Datenbereichs markiert werden. Das Makro für diese Aufgabe können Sie Listing 3.13 entnehmen.

```
Sub Letzte3ZeilenInSpalteMarkieren()
 Sheets("Tabelle4").Activate
 Range("A1048576").End(xlUp).Select
 Range(Selection, _
 ActiveCell.Offset(-2, 0)).EntireRow.Select
End Sub
```

Listing 3.13: Die letzten Zeilen einer Liste markieren

Über die Anweisung `Range("A1048576").End(xlUp).Select` ermitteln Sie die letzte gefüllte Zelle in Spalte A. Bilden Sie nun eine Markierung, die ausgehend von dieser Zelle weitere zwei Zeilen oberhalb der Zelle markiert.



Die letzten drei Artikelzugänge in der Liste markieren

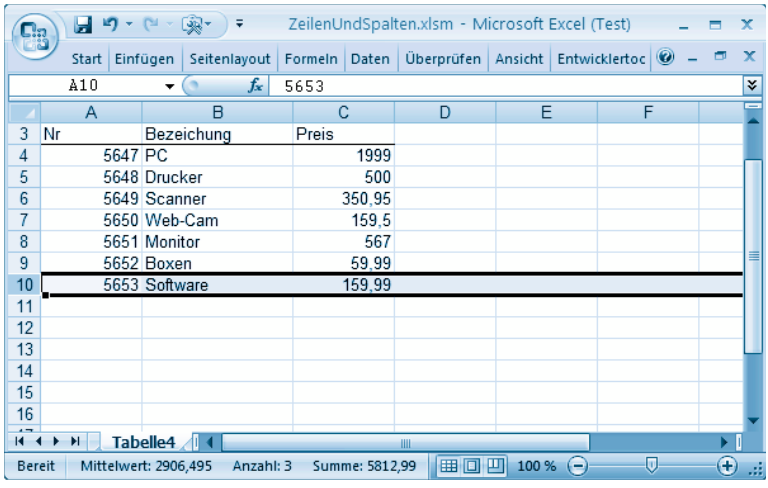
Letzte verwendete Zeile einfärben

Im nächsten Beispiel färben Sie die letzte verwendete Zeile mit der Hintergrundfarbe *Hellgelb*. Das Makro für diese Aufgabe entnehmen Sie Listing 3.14.

```
Sub LetzteZeileEinfärben()
Sheets("Tabelle4").Activate
Range("A1048576").End(xlUp).Select
 ActiveCell.EntireRow.Select
    With Selection.Interior
        .ColorIndex = 36
        .Pattern = xlSolid
    End With
End Sub
```

Listing 3.14: Die letzte Zeile in einer Tabelle ermitteln und einfärben

Ermitteln Sie über die Eigenschaft `End` sowie den Richtungsoperator `xlUp` die zuletzt verwendete Zeile. Danach markieren Sie die gesamte Zeile mit Hilfe der Eigenschaft `EntireRow`. Führen Sie daraufhin die Formatierung der Zeile durch, indem Sie den `ColorIndex` auf den Wert `36` setzen. Über die Eigenschaft `Pattern` legen Sie das Hintergrundmuster des Bereichs fest.



Die letzte Zeile in der Liste einfärben

Zeilenhöhe und Spaltenbreite einstellen

Bei der Einstellung für die richtige Spaltenbreite bzw. die korrekte Zeilenhöhe können Sie bestimmen, ob die Einstellung für Ihr ganzes Tabellenblatt oder nur für bestimmte Spalten und Zeilen gelten soll.

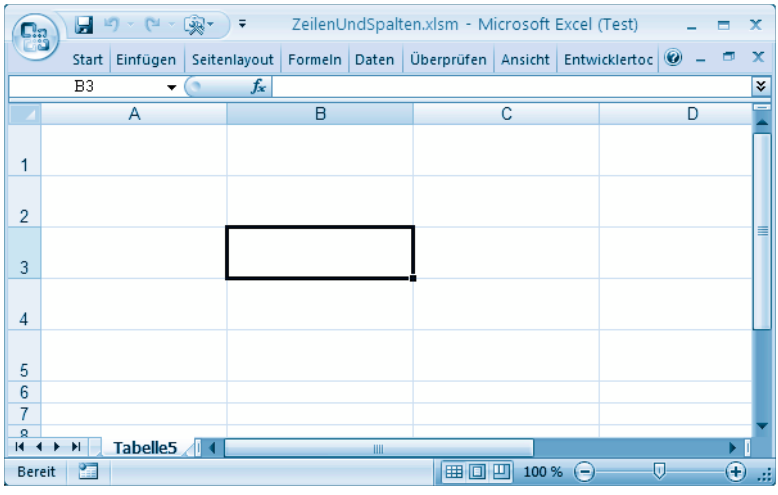
Im ersten Beispiel werden auf dem Tabellenblatt *Tabelle5* die Spalten A bis C sowie die Zeilen 1 bis 5 angepasst. Dazu setzen Sie im ersten Fall die Eigenschaften `EntireColumn` und `ColumnWidth` ein, um die Spaltenbreite festzulegen. Im zweiten Fall wenden Sie die Eigenschaften `EntireRow` und `RowHeight` auf das Objekt `Range` an, um die Zeilenhöhe festzulegen.

```
Sub SpaltenZeilenEinstellen()
 Sheets("Tabelle5").Activate
 Range("A:C").EntireColumn.ColumnWidth = 20
 Range("1:5").EntireRow.RowHeight = 30
End Sub
```

Listing 3.15: Spaltenbreite und Zeilenhöhen anpassen

Hinweis

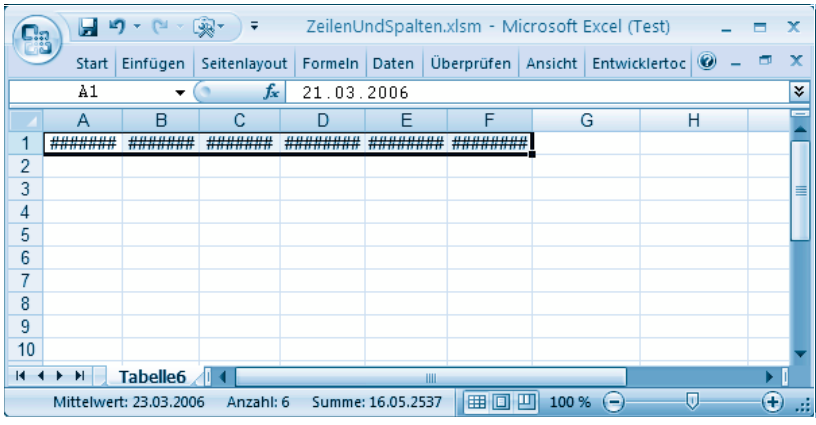
Die Einheit für die Spaltenbreite bzw. die Zeilenhöhe entspricht der Breite eines Zeichens in der Formatvorlage STANDARD. Bei Proportionalschriftarten wird die Breite des Zeichens o (Null) verwendet.



Zeilenhöhe und Spaltenbreite einstellen

Tipp

Gerade haben Sie erfahren, wie Sie Zeilen und Spalten eine exakte Höhe und Breite geben können. Ebenso haben Sie die Möglichkeit, beispielsweise den Spalten genau die Breite zu geben, die benötigt wird, um eine Eingabe in kompletter Länge anzuzeigen. Dazu verwenden Sie die Methode `AutoFit`., welche die angegebenen Spalten automatisch in der richtigen Breite einstellt. Dabei orientiert sich diese Methode am Platzbedarf der längsten Eingabe einer Zelle, die sie finden kann. Auf Basis dieser Länge werden dann auch alle anderen Zellen der Spalte angepasst. Sicher kennen Sie die Reaktion von Excel, wenn eine Spalte zu klein ist, um alle Daten anzeigen zu können. Excel quittiert dies, indem es die Zelle mit der Zeichenfolge #### auffüllt. Erst wenn Sie die Spalte mit einem Doppelklick auf die Begrenzung des jeweiligen Spaltenkopfes anpassen, wird die ganze Spalte so vergrößert, dass der komplette Inhalt angezeigt werden kann.

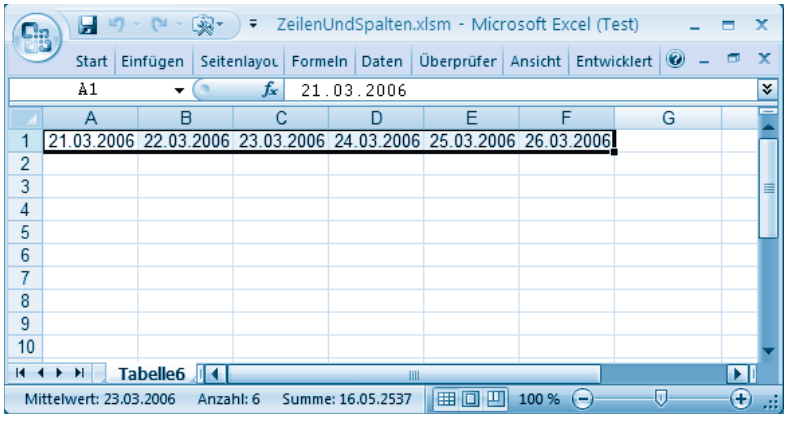


Die einzelnen Zellen sind zu klein

Um diese zu kleinen Spalten automatisch dem tatsächlichen Platzbedarf anzupassen, starten Sie das Makro aus Listing 3.16.

```
Sub SpaltenbreitenNachBedarfEinstellen()
Worksheets("Tabelle6").Columns("A:F").AutoFit
End Sub
```

Listing 3.16: Die Spalten A bis G werden automatisch richtig eingestellt



Alle Spalten sind automatisch angepasst worden

Zeilen einfügen und löschen

Wenn Sie in Ihrem Tabellenblatt Zeilen hinzufügen oder auch löschen möchten, setzen Sie die Methode `Insert` bzw. die Methode `Delete` ein. Vorher muss die gewünschte Zeile komplett markiert werden. Dies erreichen Sie mit Hilfe der Anweisung `Selection.EntireRow`.

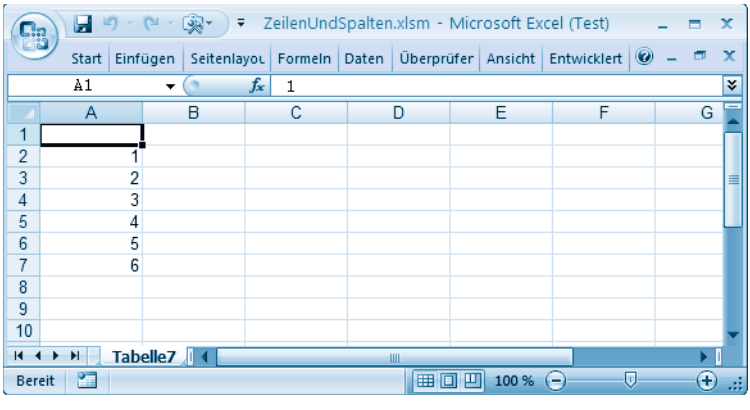
Zeilen einfügen

Im nächsten Beispiel erstellen Sie eine neue Tabelle und geben in Spalte A einige Werte untereinander an. Ihre Aufgabe besteht nun darin, eine neue, leere Zeile vor der ersten gefüllten Zeile einzufügen. Diese Aufgabenstellung lösen Sie mit dem Makro aus Listing 3.17.

```
Sub ZeileEinfügen()
 Sheets("Tabelle7").Activate
 Range("A1").EntireRow.Insert
End Sub
```

Listing 3.17: Eine einzelne Zeile einfügen

Wenden Sie die Methode `Insert` an, um eine neue Zeile einzufügen. Damit wirklich eine ganze Zeile und nicht nur eine Zelle eingefügt wird, hängen Sie vor die Methode `Insert` die Eigenschaft `EntireRow`.



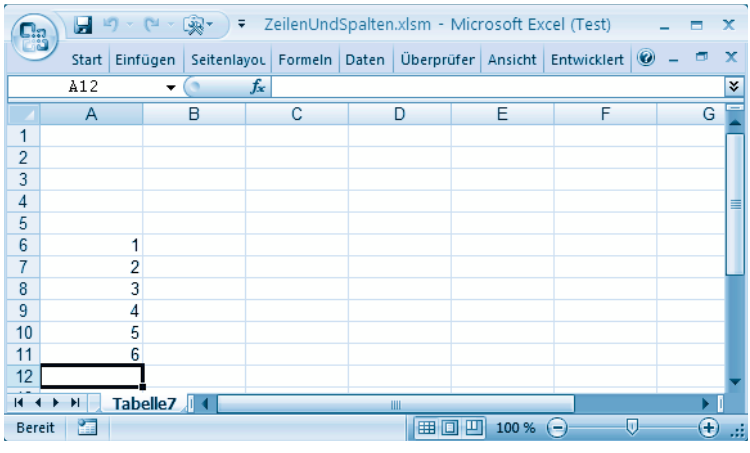
Eine leere Zeile wurde zu Beginn der Tabelle eingefügt

Sollen es gleich mehrere Zeilen sein, die eingefügt werden sollen, dann starten Sie das Makro aus Listing 3.18.

```
Sub ZeilenEinfügen()
  Sheets("Tabelle7").Activate
  Rows("1:5").Insert Shift:=xlDown
End Sub
```

Listing 3.18: Mehrere Zeilen einfügen

Hier werden genau fünf Zeilen oberhalb der ersten Zeile eingefügt, d.h. die übrigen Zeilen werden um fünf Zeilen nach unten geschoben. Die Eigenschaft `Rows` markiert die Zeilen 1 bis 5. Danach werden fünf neue Zeilen über die Methode `Insert` eingefügt.



Fünf Zeilen zu Beginn der Tabelle einfügen

Zeilen löschen

Um eine Zeile in Ihrer *Tabelle7* wieder zu löschen, starten Sie das Makro aus Listing 3.19.

```
Sub ZeileLöschen()
 Sheets("Tabelle7").Activate
 Range("A1").EntireRow.Delete
End Sub
```

Listing 3.19: Die erste Zeile der Tabelle löschen

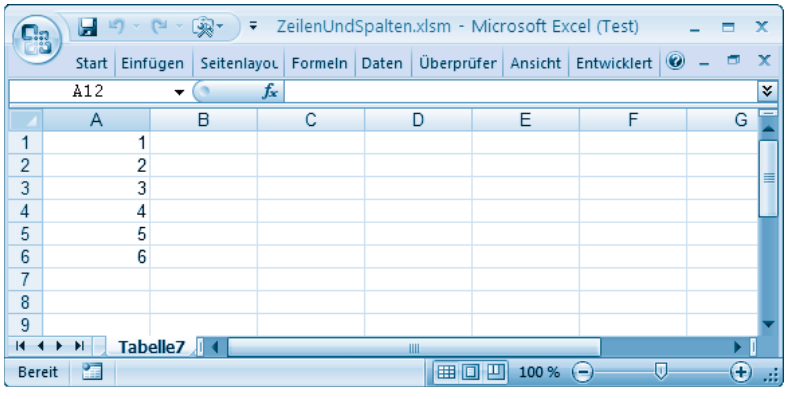
Aktivieren Sie im ersten Schritt die Tabelle, auf der Sie eine Zeile löschen möchten. Wenden Sie danach die Methode `Delete` an, um die Zeile zu entfernen. Damit auch wirklich die gesamte Zeile und nicht nur die aktive Zelle gelöscht wird, setzen Sie vor die Methode `Delete` noch die Eigenschaft `EntireRow`.

Um gleich mehrere Zeilen zu löschen, können Sie das Makro aus Listing 3.20 einsetzen.

```
Sub ZeilenLöschen()
 Sheets("Tabelle7").Activate
 Rows("1:5").Delete Shift:=xlUp
End Sub
```

Listing 3.20: Die ersten fünf Zeilen werden gelöscht

Markieren Sie zunächst per VBA die Zeilen, die Sie löschen möchten, und wenden danach die Methode `Delete` an.



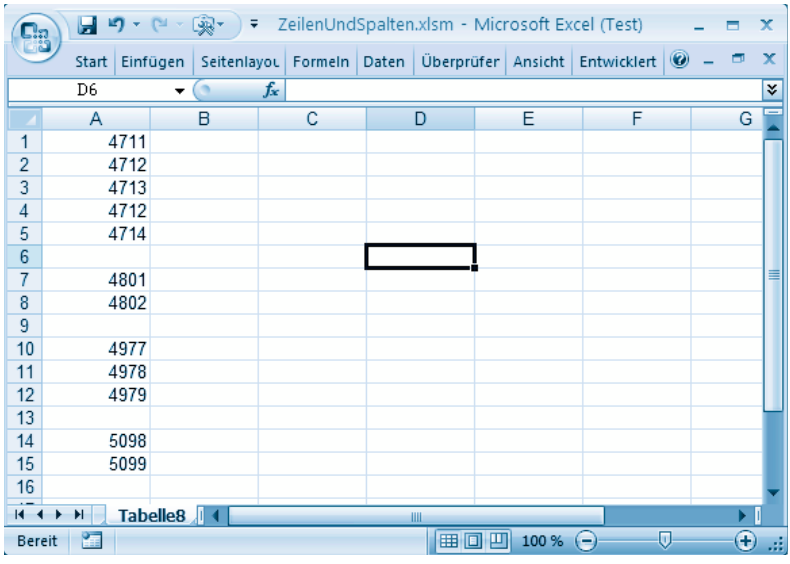
Die ersten fünf Zeilen wurden gelöscht

Bedingtes Löschen

1 Gehen Sie jetzt einen Schritt weiter und löschen Sie bestimmte Zeilen, die einer bestimmten Bedingung entsprechen.

2 Geben Sie jetzt in eine leere Tabelle in Spalte A einige Werte untereinander ein.

3 Zwischen den einzelnen Zeilen lassen Sie ruhig ein paar Zellen frei. Die Tabelle könnte dann etwa wie folgt aussehen:



Die Ausgangssituation – eine Tabelle mit Leerzeilen

4 Wie Sie sehen, sind in der Abbildung oben die Zeilen 6, 9 und 13 leer.

5 Erfassen Sie jetzt folgendes Makro aus Listing 3.21.

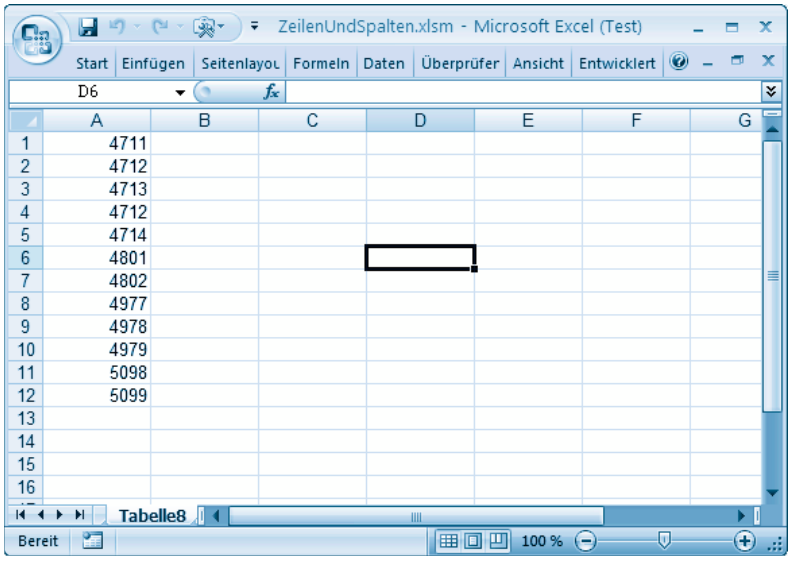
```
Sub LeerzeilenLöschen()
Dim intz As Integer
Dim intGesamt As Integer

 With Sheets("Tabelle8")
 intGesamt = .UsedRange.Rows.Count

 For intz = intGesamt To 1 Step -1
  If .Rows(intz).Text = "" Then .Rows(intz).Delete
Next intz
 End With
End Sub
```

Listing 3.21: Leerzeilen löschen

Im ersten Schritt ermitteln Sie die Anzahl der belegten Zeilen in Ihrer Tabelle. Die ermittelte Anzahl speichern Sie in der Variablen `intGesamt` vom Typ `Integer`. Danach setzen Sie eine `For Next`-Schleife auf, die so lange ausgeführt wird, bis die letzte Zeile im verwendeten Bereich erreicht wird. Innerhalb der Schleife ermitteln Sie mit Hilfe der Eigenschaft `Text`, ob überhaupt Zeichen in der Zelle enthalten sind. Wenn nicht, dann wird die komplette Zeile gelöscht.



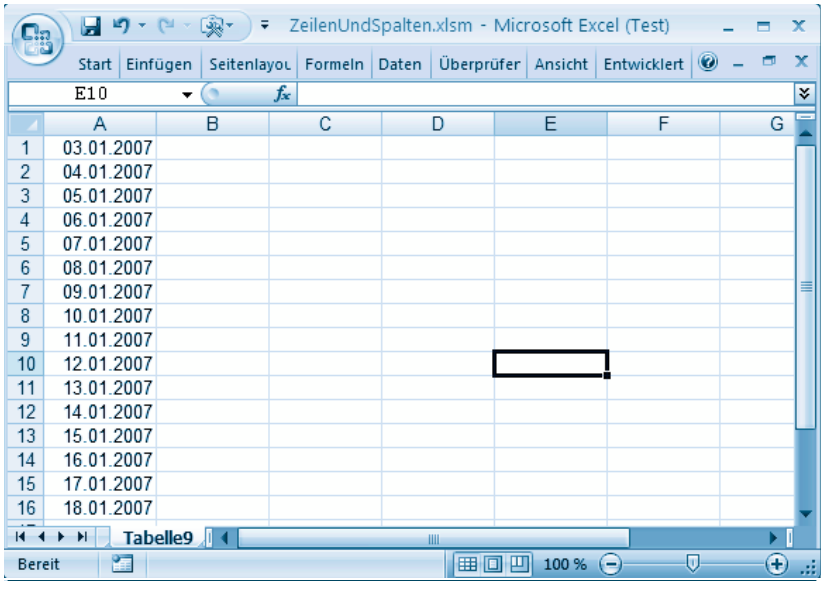
Alle Leerzeilen wurden gelöscht

Im folgenden Beispiel werden aus einer Tabelle mit Datumsangaben alle Wochenendzeilen herausgelöscht. Führen Sie dazu folgende vorbereitende Schritte durch:

1 Setzen Sie den Mauszeiger auf Zelle A1.

2 Drücken Sie die Tastenkombination Strg + .

3 Ziehen Sie das Ausfüllkästchen der Zelle A1 nach unten bis in Zelle A19. Sie haben jetzt eine Datumsleiste in der Spalte A.



Aus der Datumsleiste sollen die Wochenenden gelöscht werden

Da Sie auf den ersten Blick nicht erkennen können, ob es sich bei der jeweiligen Zelle um einen Wochentag oder um einen Tag am Wochenende handelt, können Sie die Löschaktion durch ein Makro ausführen lassen.

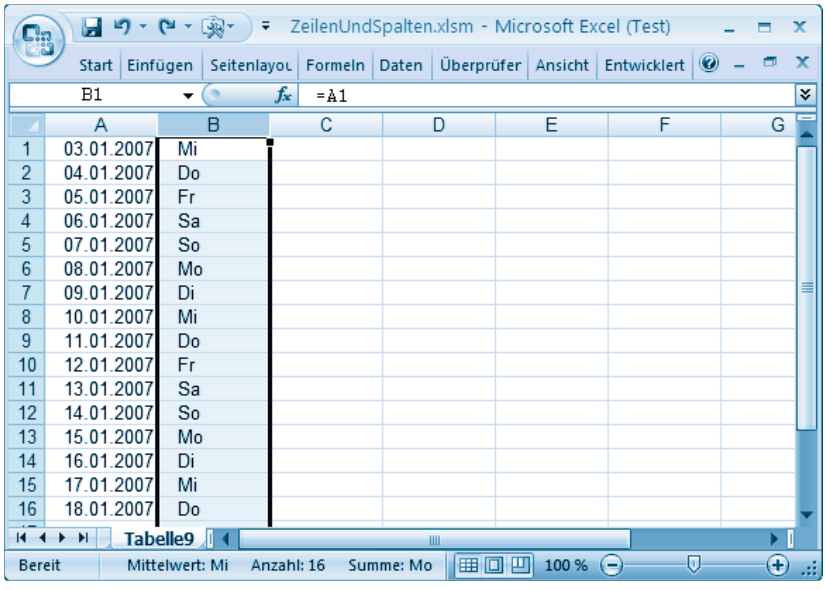
Erledigen Sie jedoch vorher folgenden Schritt, um zu testen, ob das nachfolgende Makro aus Listing 3.22 auch richtig funktioniert.

1 Kopieren Sie den Datenbereich A1:A19.

2 Fügen Sie diesen Datenbereich ab Zelle B1 ein.

3 Markieren Sie den Bereich B1:B19.

4 Weisen Sie diesem Bereich das benutzerdefinierte Format TTT zu.



Die Datumsleiste mit Tagesanzeige

Diese Darstellung ist schon besser und für Sie ist es auch eine Kontrolle, ob das nachfolgende Makro aus Listing 3.22 korrekt arbeitet. Erfassen Sie nun dieses Makro:

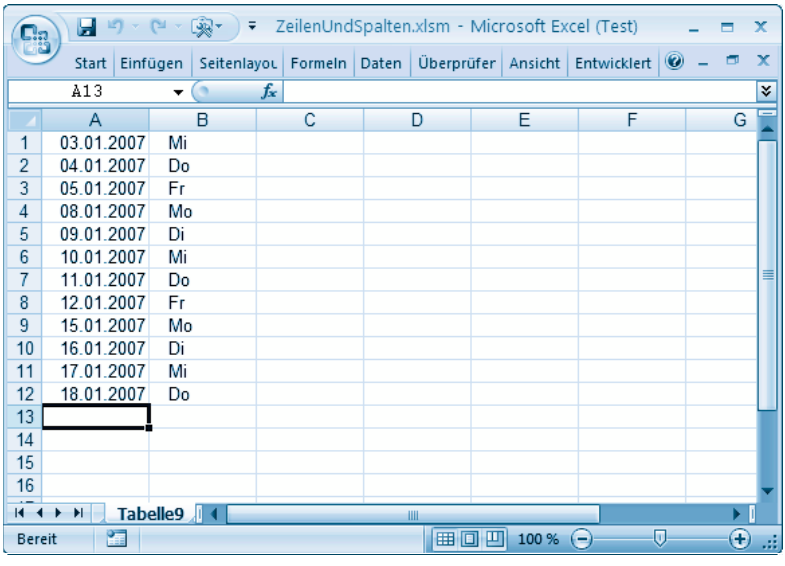
```
Sub WochenendeLöschen()
Dim intz As Integer
Dim intGesamt As Integer

With Sheets("Tabelle9")
 intGesamt = .UsedRange.Rows.Count

 For intz = intGesamt To 1 Step -1
   If Application.WorksheetFunction.Weekday _
   (.Cells(intz, 1).Value) = 1 _
    Or Application.WorksheetFunction.Weekday _
   (.Cells(intz, 1).Value) = 7 Then
   .Rows(intz).Delete
   End If
 Next intz
End With
End Sub
```

Listing 3.22: Alle Wochenendzeilen entfernen

Setzen Sie eine Schleife auf, die alle verwendeten Zeilen einer Tabelle durchläuft. Über die Funktion `Weekday` können Sie prüfen, ob eine Datumszelle ein Wochenende enthält. Da Sie einer Datumsangabe schließlich nicht ansehen können, ob dieses Datum auf ein Wochenende fällt, meldet Ihnen die Funktion `Weekday` den Wert `7` für einen Samstag und den Wert `1` für einen Sonntag zurück. In beiden Fällen löschen Sie die komplette Zeile durch die Anweisung `.Rows(intz).Delete`.



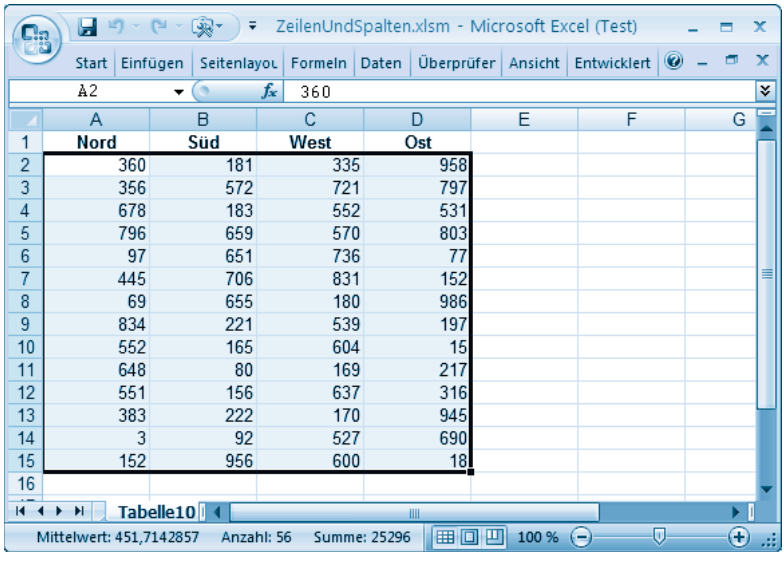
Alle Wochenendzeilen wurden gelöscht

Spalten einfügen und löschen

Das Einfügen von Spalten funktioniert in Excel analog zum Einfügen von Zeilen. Geben Sie bekannt, wo Sie eine Spalte einfügen möchten, nutzen Sie dann die Eigenschaft `EntireColumn` und fügen Sie mit der Methode `Insert` eine Spalte ein.

Spalten einfügen

Im nächsten Beispiel fügen Sie eine neue Spalte direkt vor der ersten Spalte A ein. Sehen Sie sich dazu folgende Ausgangssituation an:



	A	B	C	D
1	Nord	Süd	West	Ost
2	360	181	335	958
3	356	572	721	797
4	678	183	552	531
5	796	659	570	803
6	97	651	736	77
7	445	706	831	152
8	69	655	180	986
9	834	221	539	197
10	552	165	604	15
11	648	80	169	217
12	551	156	637	316
13	383	222	170	945
14	3	92	527	690
15	152	956	600	18
16				

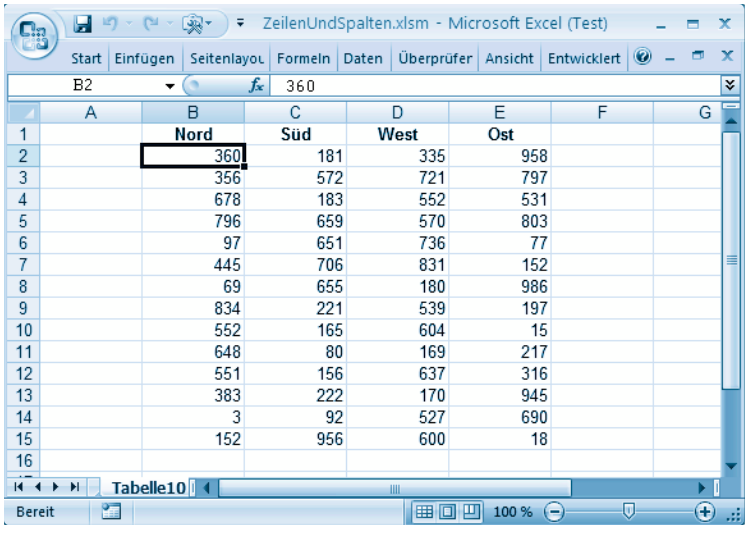
Die Ausgangssituation

Fügen Sie jetzt eine neue, leere Spalte vor Spalte A ein und verwenden Sie für diese Aufgabe das Makro aus Listing 3.23.

```
Sub SpalteEinfügen()
 Sheets("Tabelle10").Activate
 Range("A1").EntireColumn.Insert
End Sub
```

Listing 3.23: Spalte einfügen

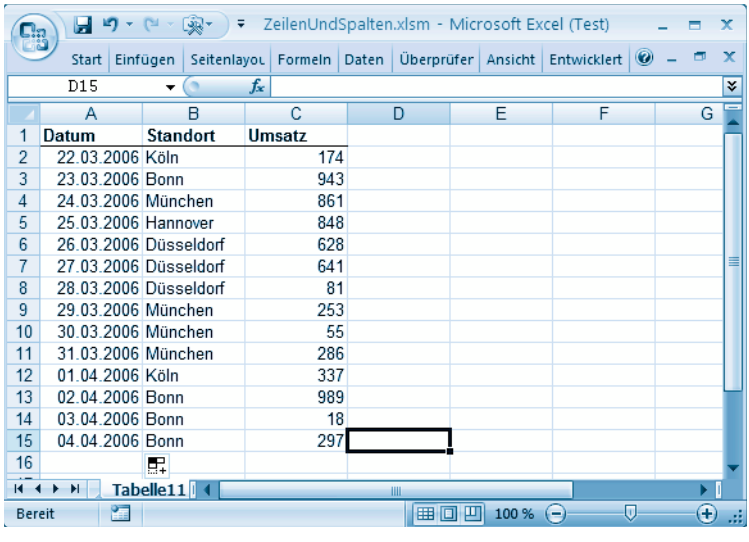
Aktivieren Sie zuerst die Tabelle, auf der Sie eine neue Spalte einfügen möchten. Fügen Sie eine neue Spalte ein, indem Sie die Methode `Insert` einsetzen, die Sie auf die komplette Spalte (`EntireColumn`) anwenden.



	A	B	C	D	E	F	G
1		Nord	Süd	West	Ost		
2		360	181	335	958		
3		356	572	721	797		
4		678	183	552	531		
5		796	659	570	803		
6		97	651	736	77		
7		445	706	831	152		
8		69	655	180	986		
9		834	221	539	197		
10		552	165	604	15		
11		648	80	169	217		
12		551	156	637	316		
13		383	222	170	945		
14		3	92	527	690		
15		152	956	600	18		
16							

Eine leere Spalte wurde eingefügt

Im nächsten Beispiel möchten Sie auf der *Tabelle11* zwei Spalten zwischen den Spalten B und C einfügen. Sehen Sie sich vorher einmal die Ausgangssituation an.



	A	B	C	D	E	F	G
1	Datum	Standort	Umsatz				
2	22.03.2006	Köln	174				
3	23.03.2006	Bonn	943				
4	24.03.2006	München	861				
5	25.03.2006	Hannover	848				
6	26.03.2006	Düsseldorf	628				
7	27.03.2006	Düsseldorf	641				
8	28.03.2006	Düsseldorf	81				
9	29.03.2006	München	253				
10	30.03.2006	München	55				
11	31.03.2006	München	286				
12	01.04.2006	Köln	337				
13	02.04.2006	Bonn	989				
14	03.04.2006	Bonn	18				
15	04.04.2006	Bonn	297				
16							

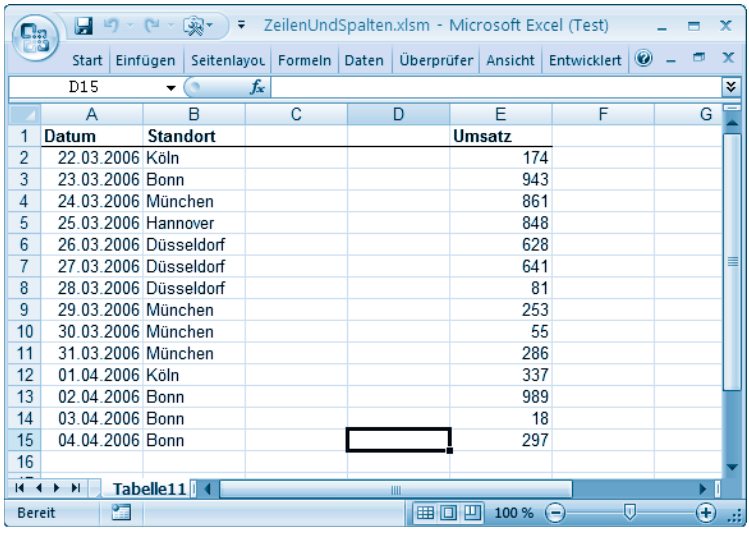
Zwischen Standort und Umsatz sollen zwei Spalten eingefügt werden

Erfassen Sie jetzt das Makro aus Listing 3.24.

```
Sub SpaltenEinfügen()
 Sheets("Tabelle11").Select
 Columns("C:D").Insert Shift:=xlToRight
End Sub
```

Listing 3.24: Mehrere Spalten einfügen

Das Einfügen der neuen Spalten wird durch die Methode `Insert` realisiert, der Sie als Konstante die Verschieberichtung `xlToRight` (nach rechts) bekannt geben.



	A	B	C	D	E	F	G
1	Datum	Standort			Umsatz		
2	22.03.2006	Köln			174		
3	23.03.2006	Bonn			943		
4	24.03.2006	München			861		
5	25.03.2006	Hannover			848		
6	26.03.2006	Düsseldorf			628		
7	27.03.2006	Düsseldorf			641		
8	28.03.2006	Düsseldorf			81		
9	29.03.2006	München			253		
10	30.03.2006	München			55		
11	31.03.2006	München			286		
12	01.04.2006	Köln			337		
13	02.04.2006	Bonn			989		
14	03.04.2006	Bonn			18		
15	04.04.2006	Bonn			297		
16							

Es wurden zwei leere Spalten eingefügt

Spalten löschen

Zum Löschen einer Spalte verwenden Sie die Methode `Delete`. Um dies an einem Beispiel zu demonstrieren, greifen Sie auf den Fall zurück, bei dem Sie vorher in *Tabelle10* eine Spalte eingefügt haben. Entfernen Sie diese jetzt wieder und setzen Sie dafür das Makro aus Listing 3.25 ein.

```
Sub SpalteLöschen()
 Sheets("Tabelle10").Activate
 Range("A1").EntireColumn.Delete
End Sub
```

Listing 3.25: Spalte löschen

Sollen gleich mehrere Spalten gelöscht werden, dann starten Sie das Makro aus Listing 3.26. Greifen Sie dazu auf das Beispiel zurück, bei dem Sie vorher in *Tabelle11* zwei zusätzliche Spalten eingefügt haben. Entfernen Sie diese jetzt wieder.

```
Sub SpaltenLöschen()
 Sheets("Tabelle11").Activate
 Columns("C:D").Delete Shift:=xlToLeft
End Sub
```

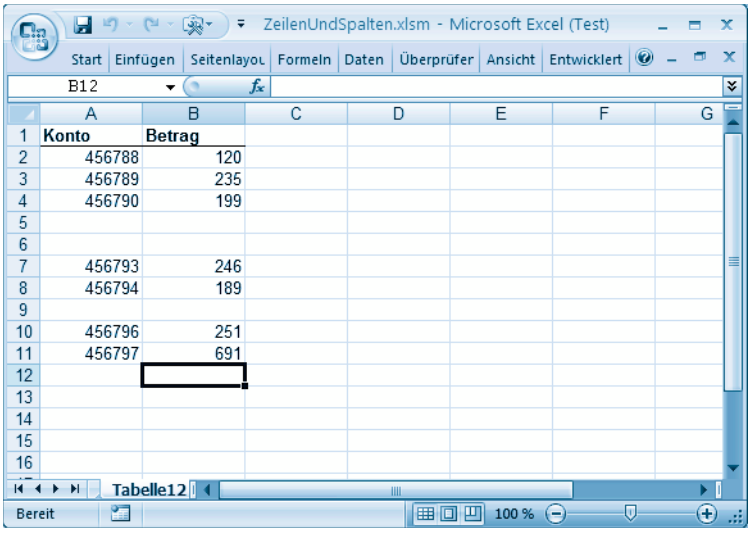
Listing 3.26: Mehrere Spalten löschen

Zeilen ein- und ausblenden

Wenn Sie Informationen in Ihrer Tabelle nicht anzeigen möchten, können Sie diese zeitweise ausblenden und bei Bedarf wieder einblenden.

Leere Zeilen ausblenden

Im folgenden Beispiel werden alle Leerzeilen ausgeblendet. Eine Zeile gilt genau dann als leer, wenn in der betreffenden Zeile in Spalte A kein Eintrag vorhanden ist. Sehen Sie sich dabei folgende Ausgangssituation an:



	A	B	C	D	E	F	G
1	Konto	Betrag					
2	456788	120					
3	456789	235					
4	456790	199					
5							
6							
7	456793	246					
8	456794	189					
9							
10	456796	251					
11	456797	691					
12							
13							
14							
15							
16							

Die Leerzeilen sollen ausgeblendet werden

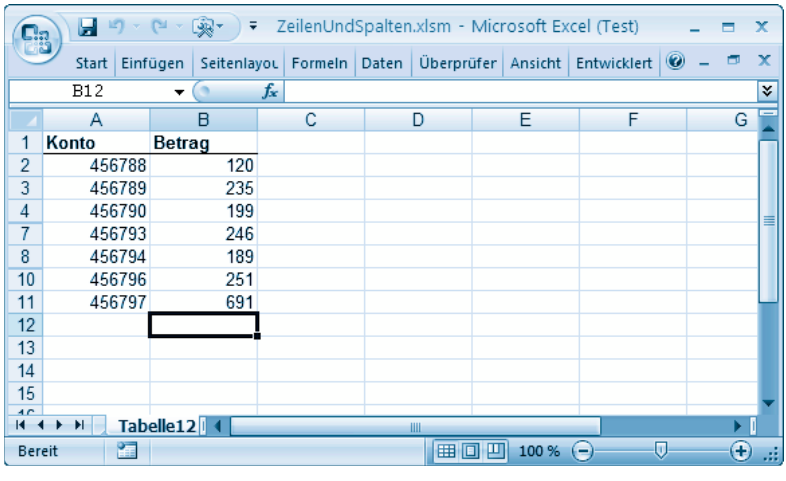
Das Makro für diese Aufgabe lautet:

```
Sub AusblendenLeereZeilen()Dim intz As Integer


With Sheets("Tabelle12")
For intz = 1 To .UsedRange.Rows.Count
 If IsEmpty(.Cells(intz, 1).Value) Then
   .Cells(intz, 1).EntireRow.Hidden = True
 End If
Next intz
End With
End Sub
```

Listing 3.27: Jede leere Zelle wird ausgeblendet

Über die Anweisung `.UsedRange.Rows.Count` ermitteln Sie die Anzahl der gefüllten Zeilen der Tabelle. Diese ermittelte Anzahl bildet die Endbedingung für die folgende Schleife. Durchlaufen Sie jetzt alle verwendeten Zellen der Spalte A und prüfen Sie, ob diese leer sind. Wenn ja, dann blenden Sie diese Zeilen über die Anweisung `.Cells(intz, 1).EntireRow.Hidden = True` aus.



Die leeren Zeilen wurden ausgeblendet

Hinweis

Wenn Sie auf die Zeilenköpfe links schauen, dann sehen Sie, dass die Nummerierung durchbrochen ist. Nach Zeilennummer 4 kommt gleich die Zeilennummer 7, was bedeutet, dass die Zeilen 5 und 6 ausgeblendet sind.

Zeilen einblenden

Um alle versteckten Zeilen einer Tabelle einzublenden, starten Sie das Makro aus Listing 3.28.

```
Sub AlleAusgeblendetenZeilenEinblenden()
Dim rngZeile As Range

With Sheets("Tabelle12")
 For Each rngZeile In .UsedRange.Rows
   rngZeile.Hidden = False
 Next rngZeile
End With
End Sub

'oder ganz kurz

Sub AllesSichtbar()
 Sheets("Tabelle12").Rows.Hidden = False
End Sub
```

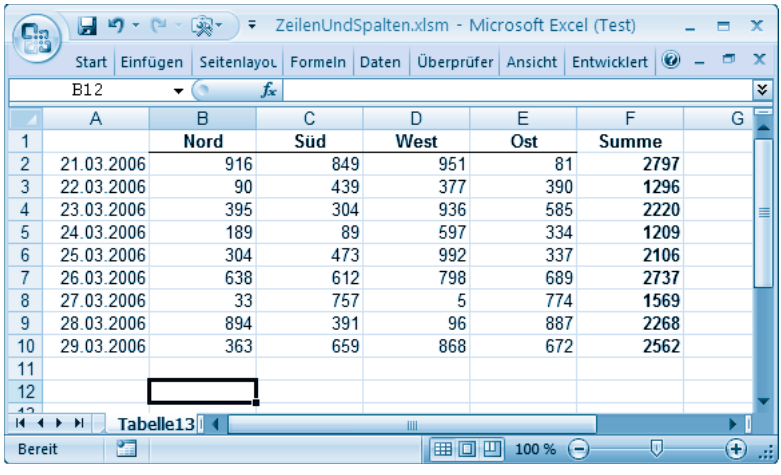
Listing 3.28: Alle ausgeblendeten Zeilen wieder einblenden

Definieren Sie zuerst eine Objektvariable, die die Zeile darstellen soll. Danach durchlaufen Sie den benutzten Bereich in einer `For Each`-Schleife und setzen die Eigenschaft `Hidden` jeder Zeile auf den Wert `True`. Damit blenden Sie alle versteckten Zeilen automatisch wieder ein. Alternativ können Sie auch dem Auflistungsobjekt `Rows`, in dem standardmäßig alle Zeilen einer Tabelle verzeichnet sind, den Wert `False` bei der Eigenschaft `Hidden` zuweisen. Damit werden alle Zeilen der Tabelle automatisch wieder sichtbar gemacht.

Spalten ein- und ausblenden

Wenn Sie sehr umfangreiche Tabellen besitzen, in denen Sie zwar alle Spalten benötigen, aber je nach Aufgabe jeweils nur ein Teil davon sichtbar zu sein braucht, dann können Sie die momentan nicht benötigten Informationen ausblenden.

Im nächsten Beispiel werden in der *Tabelle13* die Spalten B bis E nicht benötigt. Sehen Sie sich dazu einmal folgende Ausgangssituation an:



	A	B	C	D	E	F	G
1		Nord	Süd	West	Ost	Summe	
2	21.03.2006	916	849	951	81	2797	
3	22.03.2006	90	439	377	390	1296	
4	23.03.2006	395	304	936	585	2220	
5	24.03.2006	189	89	597	334	1209	
6	25.03.2006	304	473	992	337	2106	
7	26.03.2006	638	612	798	689	2737	
8	27.03.2006	33	757	5	774	1569	
9	28.03.2006	894	391	96	887	2268	
10	29.03.2006	363	659	868	672	2562	
11							
12							

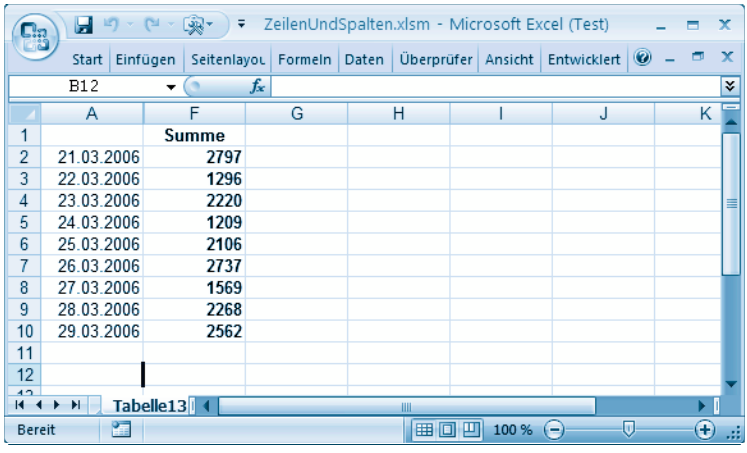
Die Spalten B bis E sollen ausgeblendet werden

Blenden Sie jetzt diese Spalten aus, indem Sie das Makro aus Listing 3.29 starten.

```
Sub SpaltenAusblenden()
 Sheets("Tabelle13").Activate
Columns("B:E").EntireColumn.Hidden = True
End Sub
```

Listing 3.29: Nicht benötigte Spalten ausblenden

Mit Hilfe der Eigenschaft `EntireColumn` sprechen Sie die Spalten B:E an. Auf diese Spalten wenden Sie dann die Eigenschaft `Hidden` an, indem Sie diese auf den Wert `True` setzen. Damit werden diese Spalten ausgeblendet.



Mehrere Spalten wurden ausgeblendet

> **Hinweis**
>
> Wenn Sie die Spaltenköpfe oben betrachten, dann werden Sie feststellen, dass zwischen den Buchstaben A und F eine Lücke klafft. Die Spalten B, C, D und E sind ausgeblendet.

Spalten einblenden

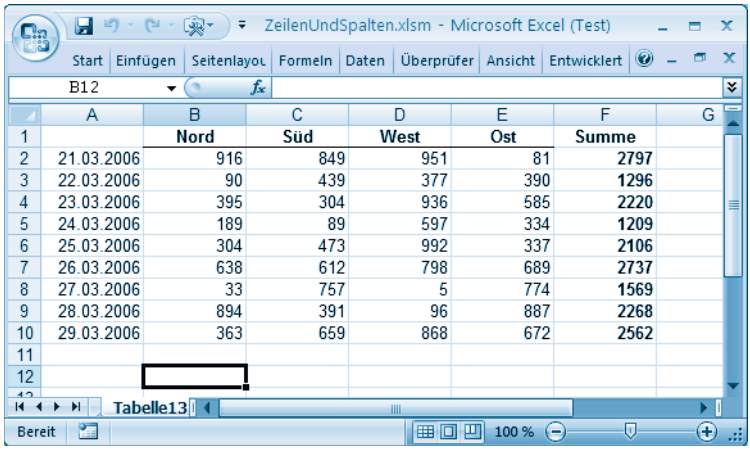
Um alle versteckten Zeilen einer Tabelle einzublenden, starten Sie das Makro aus Listing 3.30.

```
Sub AlleAusgeblendetenSpaltenEinblenden2()
Dim rngSpalte As Range

With Sheets("Tabelle13")
 For Each rngSpalte In .UsedRange.Columns
   rngSpalte.Hidden = False
 Next rngSpalte
End With
End Sub
'oder ganz kurz
Sub AlleSpaltenSichtbar()
 Sheets("Tabelle13").Columns.Hidden = false
End Sub
```

Listing 3.30: Alle versteckten Spalten einer Tabelle einblenden

Definieren Sie zuerst eine Objektvariable, die die Spalte darstellen soll. Danach durchlaufen Sie den benutzten Bereich in einer `For Each`-Schleife und setzen die Eigenschaft `Hidden` jeder Spalte auf den Wert `True`. Damit blenden Sie alle versteckten Spalten der Tabelle automatisch wieder ein. Alternativ dazu können Sie auch dem Auflistungsobjekt `Columns`, in dem standardmäßig alle Spalten einer Tabelle verzeichnet sind, den Wert `False` bei der Eigenschaft `Hidden` zuweisen. Damit werden alle Spalten der Tabelle automatisch wieder sichtbar gemacht.



	A	B	C	D	E	F	G
1		Nord	Süd	West	Ost	Summe	
2	21.03.2006	916	849	951	81	2797	
3	22.03.2006	90	439	377	390	1296	
4	23.03.2006	395	304	936	585	2220	
5	24.03.2006	189	89	597	334	1209	
6	25.03.2006	304	473	992	337	2106	
7	26.03.2006	638	612	798	689	2737	
8	27.03.2006	33	757	5	774	1569	
9	28.03.2006	894	391	96	887	2268	
10	29.03.2006	363	659	868	672	2562	
11							
12							

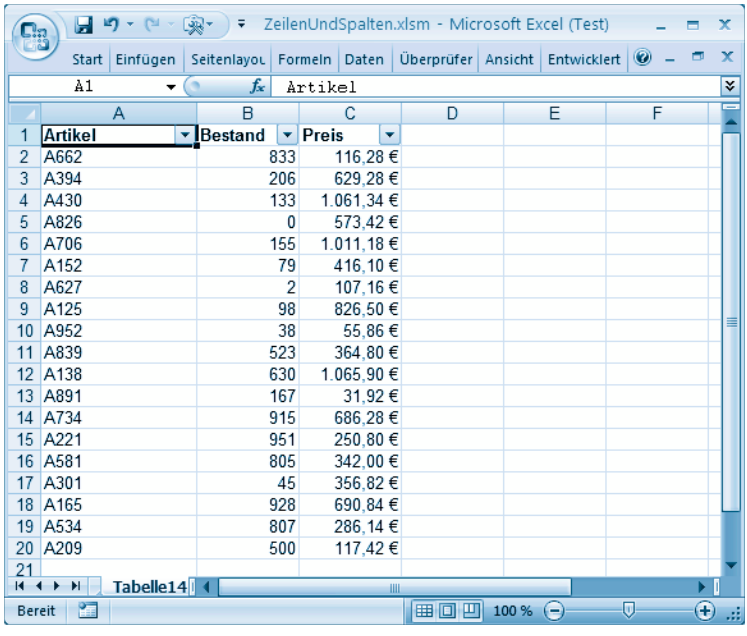
Zeilen filtern

In Excel können Sie Daten filtern. In einer gefilterten Liste werden nur die Zeilen angezeigt, die den Kriterien entsprechen, die Sie für eine Spalte angeben. Andersherum gesehen werden beim Filtern alle Zeilen ausgeblendet, die nicht den eingestellten Kriterien entsprechen. Excel stellt zwei Befehle zum Filtern von Listen zur Verfügung: die AutoFilter mit der Möglichkeit, nach der ausgewählten Zelle zu filtern, für einfache Kriterien sowie die Spezialfilter für komplexere Aufgaben. Beim Filtern von Zeilen wird die Liste nicht neu angeordnet, es werden nur vorübergehend Zeilen ausgeblendet, die nicht angezeigt werden sollen.

AutoFilter aktivieren und deaktivieren

Den AutoFilter können Sie auf Ihrer Tabelle aktivieren, indem Sie aus dem Menü *Daten* den Befehl *Filter/AutoFilter* wählen. Danach blendet Excel für jede Spalte im benutzten Bereich einen AutoFilter ein. Zunächst befinden sie sich im Ruhezustand, das heißt, es wurde noch keine Filterung vorgenommen, was

Sie auch am schwarzen Pfeil der AutoFilter erkennen können. AutoFilter, bei denen schon Filterkriterien eingestellt sind, haben einen blauen Pfeil. Beim folgenden Makro aus Listing 3.31 wird der AutoFilter, sofern er noch nicht verfügbar ist, eingeblendet.



	A	B	C
1	Artikel	Bestand	Preis
2	A662	833	116,28 €
3	A394	206	629,28 €
4	A430	133	1.061,34 €
5	A826	0	573,42 €
6	A706	155	1.011,18 €
7	A152	79	416,10 €
8	A627	2	107,16 €
9	A125	98	826,50 €
10	A952	38	55,86 €
11	A839	523	364,80 €
12	A138	630	1.065,90 €
13	A891	167	31,92 €
14	A734	915	686,28 €
15	A221	951	250,80 €
16	A581	805	342,00 €
17	A301	45	356,82 €
18	A165	928	690,84 €
19	A534	807	286,14 €
20	A209	500	117,42 €
21			

Der AutoFilter wurde aktiviert

```
Sub AutoFilterEinschalten()
 Sheets("Tabelle14").Activate
 If Not ActiveSheet.AutoFilterMode = True Then
  Range("A1").AutoFilter
 End If
End Sub
```

Listing 3.31: Den AutoFilter einschalten

Die Eigenschaft `AutoFilterMode` liefert den Wert `True`, wenn die Dropdown-Pfeile für AutoFilter momentan auf dem Tabellenblatt sichtbar sind. Liefert diese Eigenschaft den Wert `False` zurück, dann blenden Sie die AutoFilter mit Hilfe der Methode `AutoFilter` ein.

Hinweis

Wenn Sie den AutoFilter aus Ihrer Tabelle verbannen möchten, dann setzen Sie die Eigenschaft `AutoFilterMode` auf den Wert `False`.

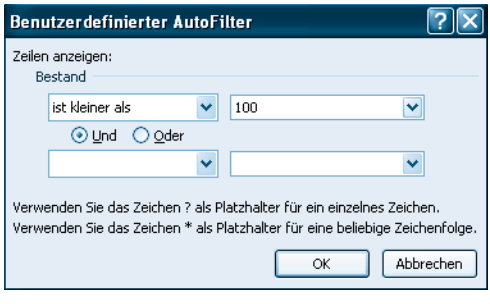
Daten filtern

Ist der AutoFilter in der Tabelle verfügbar, können einzelne Kriterien für die Spalten festgelegt werden, nach denen dann bestimmte Zeilen gefiltert werden können. Setzen Sie jetzt auf dem Beispiel aus der vorherigen Abbildung auf.

Im nächsten Beispiel sollen alle Zeilen angezeigt werden, die einen Bestand < 100 haben und daher bald nachbestellt werden müssen. Das Makro für diese Bestandsprüfung sehen Sie in Listing 3.32.

```
Sub BestandKleiner100()
 Sheets("Tabelle14").Activate
 Selection.AutoFilter _
  Field:=2, Criteria1:="<100", Operator:=xlAnd
End Sub
```

Listing 3.32: Einzelne Zeilen herausfiltern



Der benutzerdefinierte AutoFilter

Die komplette Syntax der Methode `AutoFilter` lautet:

```
Ausdruck.AutoFilter(Field, Criteria1, Operator, Criteria2,
VisibleDropDown)
```

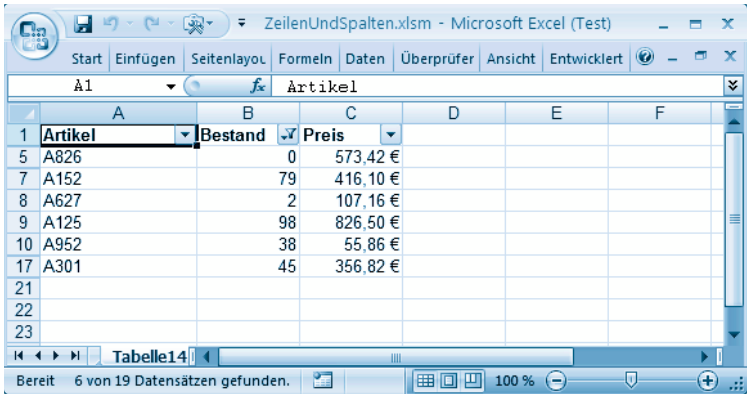
Mit der AutoFilter-Methode haben Sie der Spalte B (Field:=2) das Anzeigekriterium für Zeilen bekannt gegeben (Criteria1:="<100"). Bei dem Argument Operator haben Sie die Auswahl aus mehreren Konstanten, die Sie nachfolgender Tabelle entnehmen können.

Konstante	Erklärung
XlAnd	Es müssen zwei Kriterien erfüllt sein (Criteria1 und Criteria2).
XlOr	Es muss entweder das eine oder das andere Kriterium erfüllt sein (Criteria1 oder Criteria2).
xlTop10Items	Die n-höchsten Einträge aus der Liste werden angezeigt (Absolutbetrachtung).
xlTop10Percent	Die n-höchsten Einträge aus der Liste werden angezeigt (prozentuale Betrachtung).
XlBottom10Items	Die n-niedrigsten Einträge aus der Liste werden angezeigt (Absolutbetrachtung).
XlBottom10Percent	Die n-niedrigsten Einträge aus der Liste werden angezeigt (prozentuale Betrachtung).
Neu in Excel 2007	
xlFilterCellColor	Farbe der Zelle
xlFilterDynamic	Dynamischer Filter
xlFilterFontColor	Farbe der Schriftart
xlFilterIcon	Filtersymbol
xlFilterValues	Filterwerte

Tabelle 3.1: Die Operatoren der Methode AutoFilter

Das Argument Criteria2 stellt ein mögliches zweites Kriterium dar. Es wird zusammen mit Criteria1 und dem Argument Operator zum Erstellen von zusammengesetzten Kriterien verwendet.

Das letzte Argument VisibleDropDown ist standardmäßig mit dem Wert True voreingestellt, was bedeutet, dass die Dropdown-Pfeile für das gefilterte Feld angezeigt werden. Wenn das Argument auf den Wert False gesetzt wird, werden die Dropdown-Pfeile des AutoFilters für das gefilterte Feld ausgeblendet.



Alle Artikel mit Bestand < 100 werden angezeigt

Die kleine Erfolgskontrolle

Beantworten Sie am Ende dieses Kapitels bitte folgende Verständnisfragen:

1. Wie können Sie alle Zellen einer Tabelle markieren?
2. Welche Funktion hat die Anweisung ActiveSheet.Usedrange. Select?
3. Über welche Anweisung können Sie die letzte verwendete Zelle in Spalte B ermitteln?
4. Wie können Sie die Funktion AutoFilter für Ihre Tabelle per VBA einschalten?