R|R|Z|N
Regionales Rechenzentrum für Niedersachsen

Leibniz Universität Hannover

Excel – Automatisierung von Arbeitsschritten
Diagramme automatisiert erstellen

Diagramme ...

- visualisieren Werte in Abhängigkeit von bestimmten Randbedingungen.
- stellen grafisch Beziehungen zwischen verschiedenen Daten dar.
- stellen Abhängigkeiten zwischen Daten in kompakter und übersichtlicher Form dar.
- veranschaulichen Zusammenhänge zwischen zwei oder mehr abhängigen Werten.

Arbeitsschritte

- Das Menüband Einfügen ist geöffnet.
- Es wird eine Zelle in einem leeren Tabellenblatt ausgewählt.
- Klick auf *Einfügen – Säule: 2D Säule; Gruppierte Säule*. Ein Balkendiagramm wird erstellt. Das Menüband Diagrammtools wird geöffnet.
- Anschließend wird die Datenquelle festgelegt (*Daten auswählen*).
- Ein Diagrammlayout wird ausgewählt (*Diagrammlayouts*).
- Ein Diagrammtitel wird eingefügt und angepasst.
- Die Datenreihe wird wie gewünscht formatiert.

Arbeitsschritte automatisieren

- Voraussetzung: Die Entwicklertools sind eingeblendet.
- Die Arbeitsschritte werden einmalig mit Hilfe eines Makros aufgezeichnet.
- Nach Beendigung der Aufzeichnung werden die Arbeitsschritte automatisiert über ein Symbol oder Tastenkürzel gestartet.

Entwicklertools einblenden

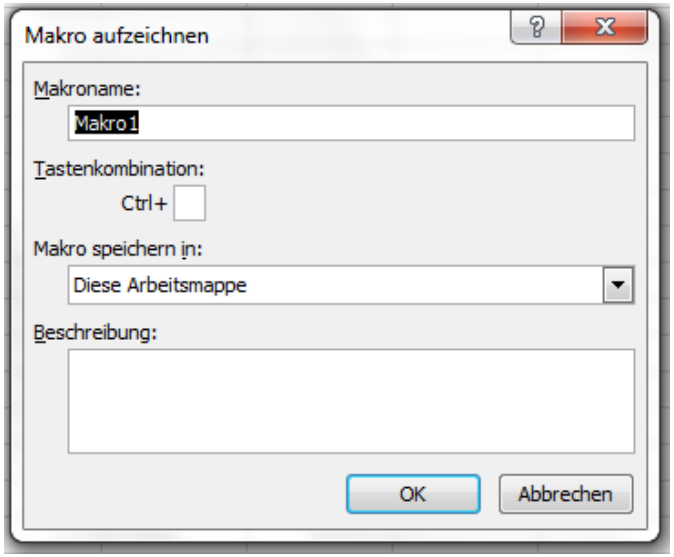
- *Datei – Optionen*.
- *Menüband anpassen*.
- In der rechten Liste Menüband anpassen werden alle Hauptregisterkarten angezeigt. Das Menüband Entwicklertools ist standardmäßig ausgeblendet (Kästchen ist leer).
- Durch einen Mausklick links von der Bezeichnung des Menübandes wird dieses aktiviert. In dem Kästchen wird ein Häkchen angezeigt.

Arbeitsschritte aufzeichnen

- Das Menüband Entwicklertools ist eingeblendet.
- Durch einen Mausklick wird die Aktion *Makro aufzchn.* in der Gruppe Code gestartet.
- Es öffnet sich der Dialog Makro aufzeichnen. In dem Dialog werden die Grundeinstellungen der Aufzeichnung festgelegt. *OK* schließt den Dialog.
- Hinweis: Die Aufzeichnung endet nicht automatisch.

Dialog „Makro aufzeichnen“

- In dem obersten Textfeld wird ein eindeutiger Name für die Aufzeichnung eingegeben.
- Zum Starten des Makros kann ein Buchstabe eingegeben werden. Hinweis: Einige Kombinationen wie <CTRL>+<C> sind belegt.
- Mit Hilfe des Kombinationsfeldes wird der Speicherort des Makros festgelegt. Standardmäßig wird ein Makro in der aktuellen Arbeitsmappe gespeichert.
- In dem unteren Textfeld kann eine Beschreibung für das Makro eingegeben werden.



Aufzeichnung beenden

- Das Menüband Entwicklertools ist eingeblendet.
- Mit einem Mausklick auf die Aktion *Aufzeichnung beenden* wird die momentan laufende Aufzeichnung beendet.
- Hinweis: Alle Arbeitsschritte zwischen dem Beginn und dem Ende der Aufzeichnung sind in der Programmiersprache VBA gespeichert.

Visual Basic for Application (VBA) ...

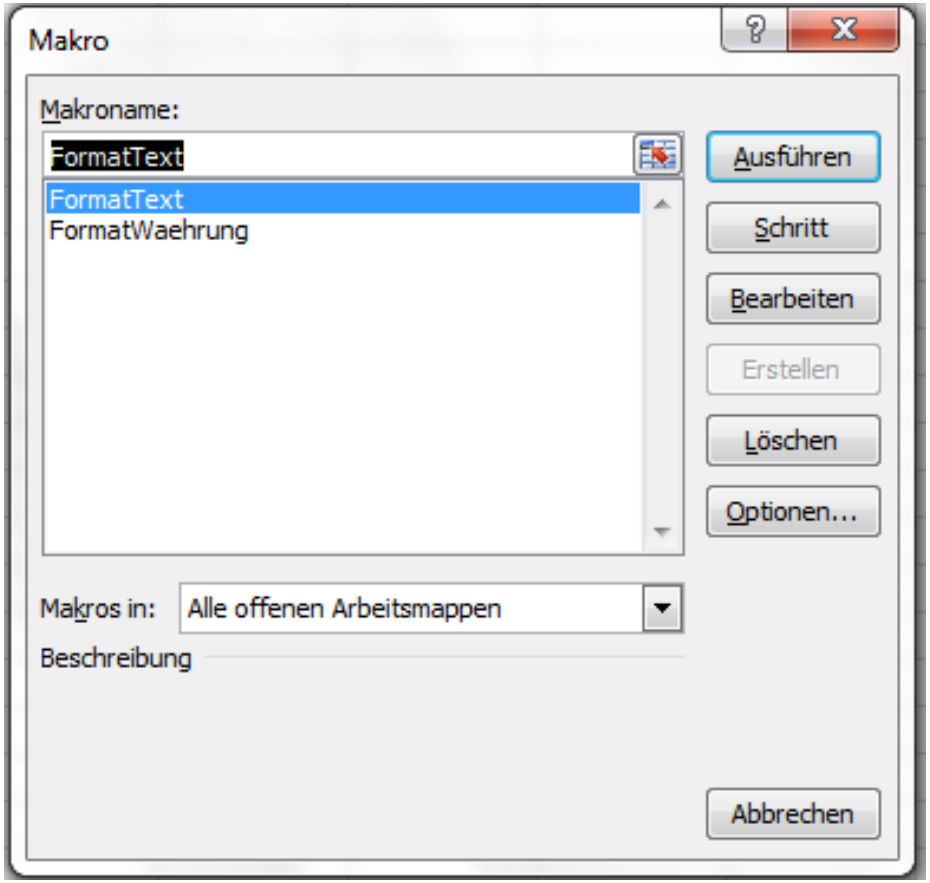
- automatisiert eine Folge von Arbeitsschritten.
- ist eine Programmiersprache, die in jeder Office-Anwendung eingebettet ist.
- erweitert die Funktionalität von Office-Anwendungen.
- passt eine Anwendung entsprechend der Wünsche des Benutzers an.

Start des Makros ...

- mit Hilfe des Symbols *Makros* im Bereich Code des Menübandes Entwicklertools.
- mit Hilfe von <CTRL> und der eingegebenen Taste.
- über ein Symbol in einem benutzerdefinierten Menüband.
- Nach dem Start werden die aufgezeichneten Arbeitsschritte abgearbeitet.

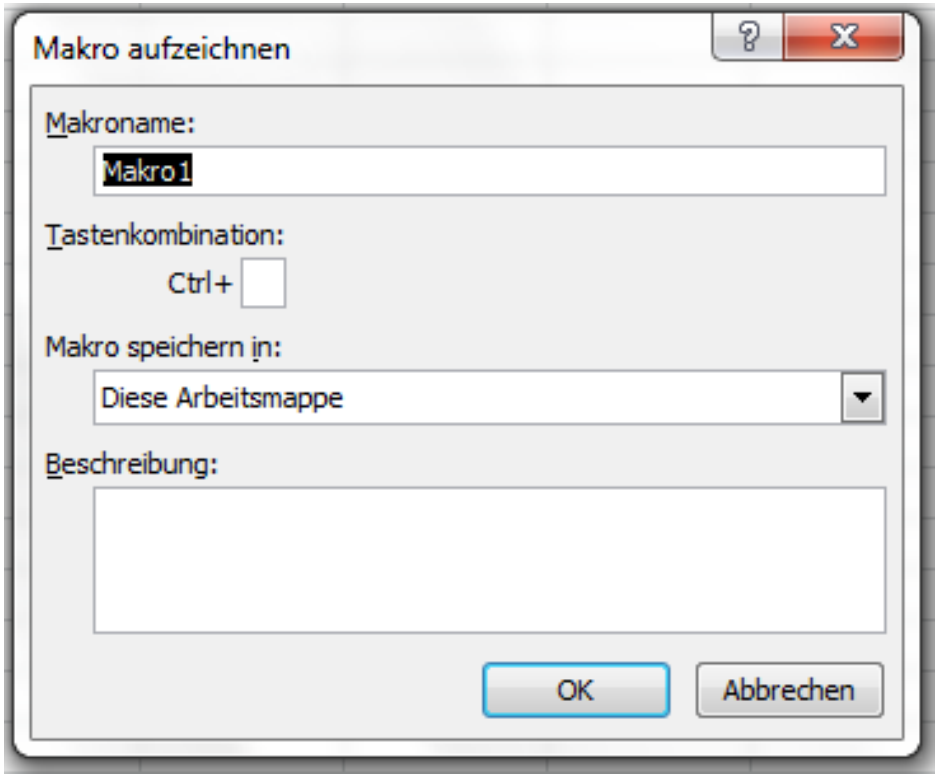
Entwicklertools - Makros

- Mit einem Mausklick wird ein Makro aus der Liste ausgewählt.
- Die Schaltfläche *Ausführen* startet das gewählte Makro.



… mit Hilfe einer Tastenkombination starten

- <CTRL>+<Taste> startet das Makro.



… über ein Menüband starten

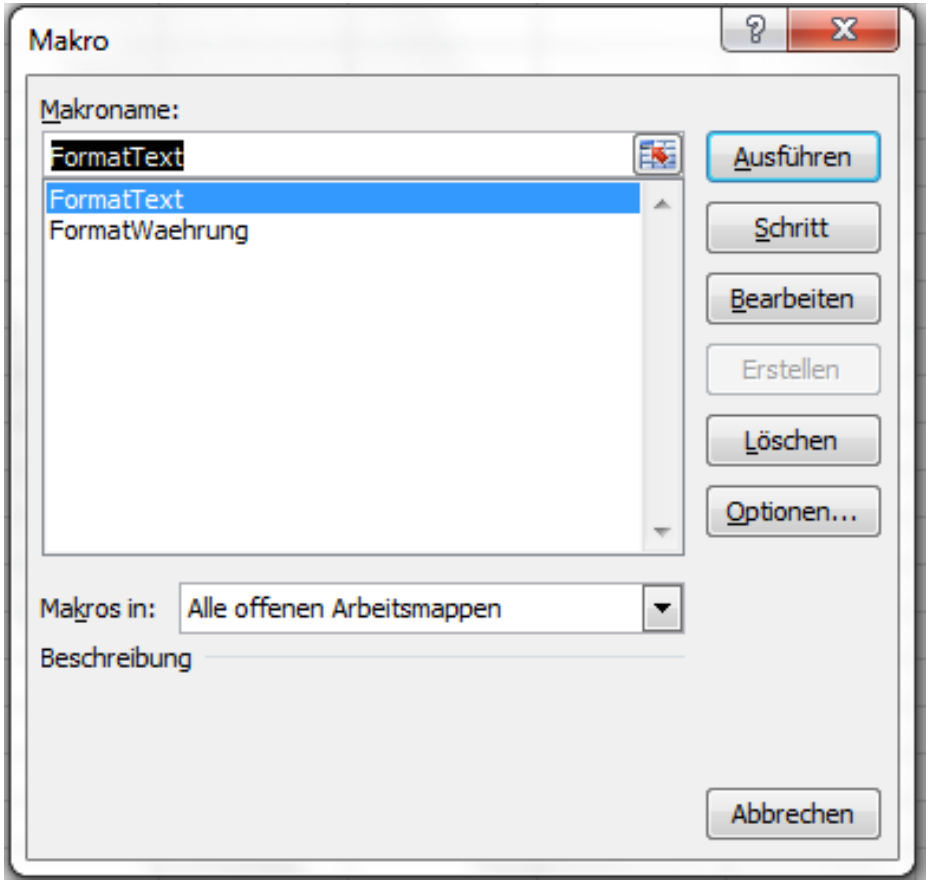
- Voraussetzungen: Das Makro ist als Icon in einem Menüband sichtbar.
- Mit einem Klick auf das passende Icon wird das Makro gestartet.

Arbeitsmappen mit einem Makro speichern

- *Datei – Speichern unter*.
- Als Dateityp wird der Eintrag Excel-Arbeitsmappe mit Makros (*.xlsm) genutzt.
- Für die Speicherung wird ein aussagekräftiger Name eingegeben.
- Ein Ordner wird als Speicherplatz ausgewählt.

Makros bearbeiten

- Das Menüband Entwicklertools ist aktiv.
- In der Gruppe Code wird auf das Symbol *Makros* geklickt.
- Die Schaltfläche *Bearbeiten* öffnet den VBA-Editor. Alle Schritte des Makros werden in VBA-Code dargestellt.
- Mit Hilfe der Schaltfläche *Optionen* werden die Voreinstellungen angezeigt und können verändert werden.



Aufgezeichnete Arbeitsschritte in VBA

```
Sub DiagrammErstellen()
    ActiveSheet.Shapes.AddChart.Select

    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.SetSourceData Source:=Sheets("KlimaWerte").Range("B5:M6")

    ActiveChart.SeriesCollection(1).Name = "=""Sonnenstunden"""
    ActiveChart.SeriesCollection(2).Name = "=""Niederschlag"""

    ActiveChart.ApplyLayout (1)
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text =  "Sonnenstunden und Niederschlag im Harz"

    Selection.Format.TextFrame2.TextRange.Characters.Text = _
                                                "Sonnenstunden und Niederschlag im Harz"
End Sub
```

Objekte in dem aufgezeichneten Code

- ActiveSheet beschreibt das aktive Arbeitsblatt.
- ActiveChart ist ein Synonym für das aktive Diagramm.
- Shapes ist eine Auflistung von gezeichneten Objekten und Bildern.

Neues eingebettetes Diagramm

ActiveSheet.Shapes.AddChart.Select

- Das aktive Arbeitsblatt ActiveSheet hat eine Auflistung Shapes. Diese Auflistung enthält auch alle eingebetteten Diagramme.
- Mit Hilfe der Methode .AddChart wird der Auflistung Shapes eine neues Element hinzugefügt. Es wird ein neues Diagramm erstellt.
- Das neu erstellte Diagramm wird ausgewählt (.Select).

Eigenschaften eines Diagramms

- Der Typ (ChartType) des Diagramms wird festgelegt. In diesem Beispiel wird ein Balkendiagramm (xlColumnClustered) erzeugt. Die Angabe xlLine erzeugt ein Liniendiagramm.
- Mit Hilfe der Methode SetSourceData wird die Datenquelle des Diagramms festgelegt.
- Mit Hilfe der Methode ApplyLayout(1) wird das erste Layout im Menüband angewendet.
- Eigenschaften und Methoden werden durch ein Punkt mit dem Objekt verbunden.

Elemente eines Diagramms

- Die Datenreihen in dem Diagramm werden in der SeriesCollection gesammelt.
- ChartTitle beschreibt den Diagrammtitel.

Nachteil des automatisch erzeugten Codes

- Die Aufzeichnung formatiert immer nur den selektierten Bereich eines Diagramms.
- Es werden Bestandteile des Diagramms abschnittweise formiert.
- Der Code ist sehr unübersichtlich.

Eigenen Code erzeugen

- Das Menüband Entwicklertools ist aktiv.
- Mit Hilfe des Symbols *Visual Basic* wird der VBA-Editor geöffnet.
- Mit Hilfe des Menüs *Einfügen* – *Modul* wird ein neue Datei angelegt.
- In diesem Modul wird der Code mit Hilfe der Tastatur eingegeben.

Prozedur erstellen

```
Sub DiagrammNeu()


End Sub
```

- Durch das Schlüsselwort Sub wird der Beginn einer Prozedur gekennzeichnet.
- Anschließend folgt ein selbsterklärender, benutzerdefinierter Name.
- Dem Namen folgen leere runde Klammern. Der Prozedur werden keine Werte übergeben.
- Die Prozedur endet mit den Schlüsselwörtern End Sub. Diese Schlüsselwörter werden automatisch von VBA gesetzt.

Arbeitsschritte eingeben

```
Sub DiagrammNeu()


End Sub
```

- Mit einem Mausklick wird die Einfügemarke zwischen den Schlüsselwörtern Sub und End Sub gesetzt.
- Mit Hilfe der Tastatur werden dort die benötigten Arbeitsschritte zeilenweise eingegeben.

Neues Diagramm einfügen

```
Sub DiagrammNeu()
   ThisWorkbook.Charts.Add After:=Worksheets("KlimaWerte")
```

- ThisWorkbook verweist auf die aktuelle Arbeitsmappe.
- Charts ist eine Auflistung aller Diagramme in Din A4-Größe. Die Diagramme werden auf eigenen Blättern dargestellt.
- Mit Hilfe der Methode Add wird ein Diagrammblatt an einer bestimmten Position in die Arbeitsmappe eingefügt und aktiviert. In diesem Beispiel wird die Arbeitsmappe nach der Mappe „KlimaWerte“ eingefügt.

Falls das Diagrammblatt vorhanden ist ...

```
Sub DiagrammNeu()
  ThisWorkbook.Charts.Add After:=Worksheets("KlimaWerte")

  For Each element In ThisWorkbook.Charts
      If element.Name = "Wetter" Then
         element.Delete
      End If
   Next
```

Für jedes Element in der Auflistung

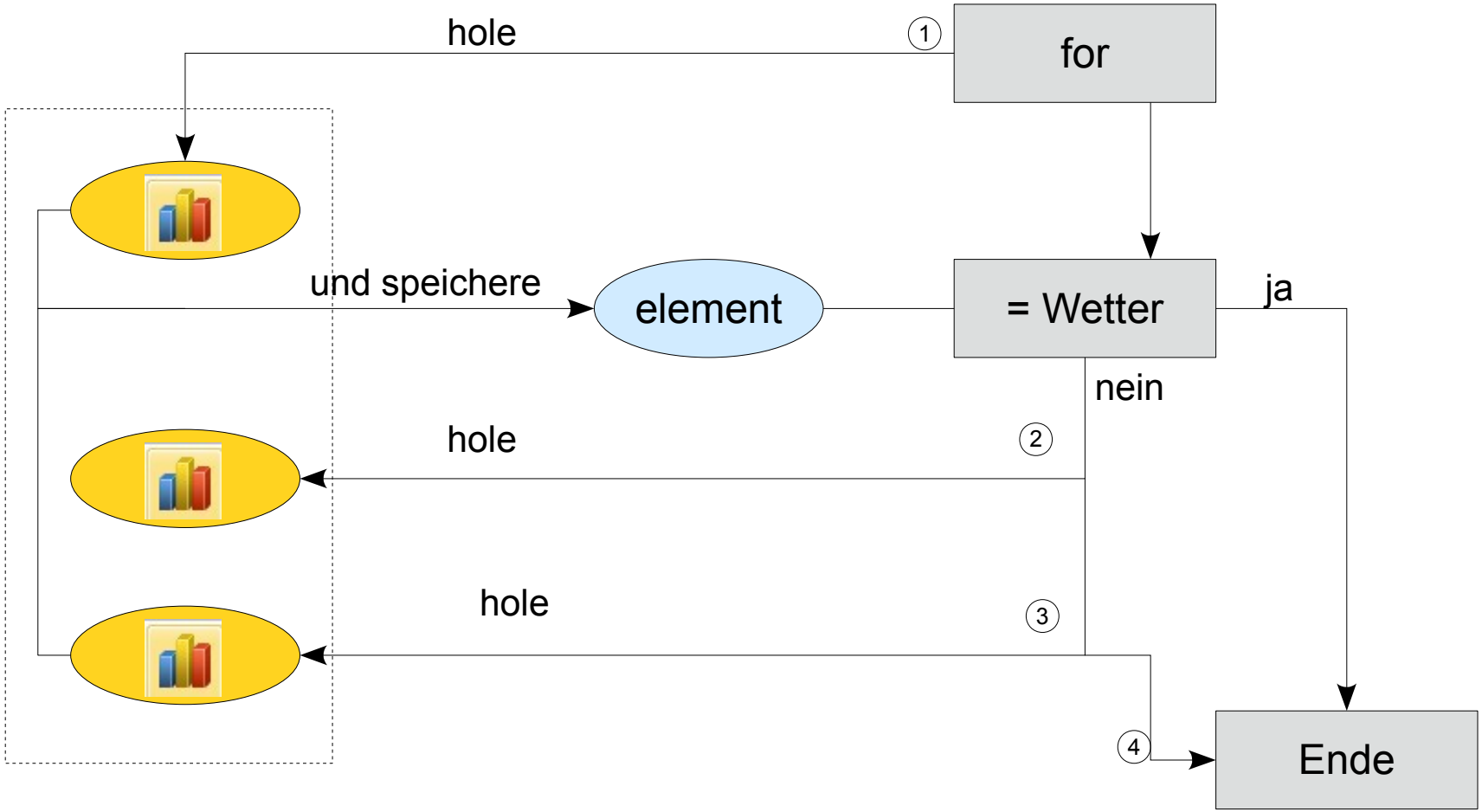
```
Dim element As Chart

For Each element In ThisWorkbook.Charts

Next
```

- Für jedes Element in der Auflistung „AktiveArbeitsmappe. Diagramme“.
- Die Auflistung wird vom ersten bis zum letzten Element vollständig durchlaufen. Der Durchlauf kann mit Hilfe von `Exit For` vorzeitig abgebrochen werden.
- Mit Hilfe von `Next` wird das nächste Element in der Auflistung geholt.

Ablauf


hole
1
for
und speichere
element
= Wetter
ja
nein
hole
2
hole
3
4
Ende

If … Then

```
If element.Name = "Wetter" Then
   element.Delete
End If
```

- Wenn die angegebene Bedingung erfüllt ist, dann ...
- Wenn das Diagrammblatt den Namen „Wetter“ besitzt, dann lösche das Blatt.
- Eine Bedingung ...
  - vergleicht immer zwei Ausdrücke.
  - gibt wahr zurück, wenn die Aussage zutrifft.

Diagrammblätter löschen

```
ThisWorkbook.Charts("Wetter").Delete
element.Delete
```

- Mit Hilfe der Methode Delete wird ein Diagrammblatt aus einer Arbeitsmappe gelöscht.
- Die Methode und das Objekt, worauf die Methode angewandt wird, werden mit einem Punkt verbunden.
- Der Objektname kann durch eine Variable ersetzt werden. Die Variable wird folgendermaßen definiert: Dim element As Chart.

Diagrammblatt formatieren

```
Sub DiagrammNeu()
    With ActiveChart
        .ChartType = xlColumnClustered
        .SetSourceData
                Source:=Sheets("KlimaWerte").Range("B5:M6")
        .Name = "Wetter"
        .HasTitle = True
        .ChartTitle.Characters.Text =
                "Niederschlag und Sonnenschein"
        .HasLegend = True
        .ChartGroups(1).Overlap = -30
    End With
```

Zusammenfassung von Eigenschaften etc.

```
With ActiveChart


End With
```

- Das Schlüsselwort With fasst Einstellung zu einem bestimmten Objekt zusammen. In diesem Beispiel werden Anweisungen zum aktiven Diagrammblatt (ActiveChart) zusammengefasst.
- Die Zusammenfassung endet mit End With.

Anweisungen in der With-Anweisung ...

```
With ActiveChart
   .ChartType = xlColumnClustered
   .SetSourceData
            Source:=Sheets("KlimaWerte").Range("B5:M6")
End With
```

- die mit einem Punkt beginnen, beziehen sich auf das angegebene Objekt.
- Der Punkt muss als Verbindung zu dem Objekt gesetzt werden.

Eigenschaften und Methoden des Diagramms

- Der Typ (ChartType) des Diagramms wird festgelegt. In diesem Beispiel wird ein Balkendiagramm erzeugt. xlLine erzeugt ein Liniendiagramm.
- Mit Hilfe der Methode SetSourceData wird die Datenquelle des Diagramms festgelegt.
- Die Eigenschaft Name legt eine Bezeichnung für das Diagrammblatt fest. Der Name wird im Register des Blattes angezeigt.
- Eigenschaften und Methoden werden durch ein Punkt mit dem Objekt verbunden.

Diagrammtitel einblenden und setzen

```
With ActiveChart
        .HasTitle = True
End With
```

- Die Anweisung .HasTitle = True blendet ein Textfeld für den Diagrammtitel ein.

Diagrammtitel formatieren

```
Sub DiagrammNeu()
    With ActiveChart.ChartTitle
        .Characters.Text = "Niederschlag und Sonnenschein"
        .Characters.Font.Name = "Arial"
        .Characters.Font.Color = RGB(0, 0, 0)
        .Characters.Font.Size = 18
        .Characters.Font.Bold = msoCTrue
        .Format.TextFrame2.TextRange.Font.Fill
        .BackColor.ObjectThemeColor = msoThemeColorAccent1
    End With
```

Erläuterung

- ActiveChart.ChartTitle.Characters definiert Formatierungen für einen Text innerhalb eines Objekts. In diesem Beispiel wird der Text des Diagrammtitels formatiert.
- ActiveChart.ChartTitle.Format verweist auf die Formatierung für einen Diagrammtitel in Office.
- ActiveChart.ChartTitle.Format.TextFrame2 ist ein Verweis auf ein Textrahmen innerhalb eines Diagramms.
- ChartTitle.Format.TextFrame2.TextRange ermöglicht Textformatierungen innerhalb eines Textrahmens.

Legende einblenden

```
With ActiveChart
        .HasLegend = True
End With
```

- Die Anweisung .HasLegend = True blendet eine Legende zur Erläuterung des Diagramms ein.
- Wenn mehr als eine Datenreihe in dem Diagramm angezeigt wird, sollte eine Legende zur Erläuterung eingeblendet werden.

Legende formatieren

```
Sub DiagrammNeu()
    With ActiveChart.Legend
        .Position = xlLegendPositionBottom

        .LegendEntries(1).Font.Name = "Arial"
        .LegendEntries(1).Font.Size = 10

        .LegendEntries(2).Font.Name = "Arial"
        .LegendEntries(2).Font.Size = 10

    End With
```

Legenden-Objekte

- ActiveChart.Legend beschreibt die Legende. Das Objekt bezieht sich auf alle Legenden-Einträge.
- ActiveChart.Legend.LegendEntries(1) bezieht sich auf den ersten Legenden-Eintrag. Das Objekt LegendEntries ist eine Auflistung von Legenden-Einträgen. In den runden Klammern wird ein Index für ein bestimmtes Element übergeben. Das erste Element hat den Index 1 und so weiter.

Schrift der Einträge

- Das Objekt Font beeinflusst die Schrift des vorangestellten Objekts.
- Die Eigenschaft .Name legt die Schriftart fest.
- Die Eigenschaft .Size legt die Schriftgröße fest.
- Die Eigenschaft .Color legt die Schriftfarbe fest.

Achsen formatieren

```
Sub DiagrammNeu()
    With ActiveChart.Axes(xlCategory)
      ' Formatierung des Hauptintervalls
      .TickLabels.NumberFormatLocal = "@"
   End With

   With ActiveChart.Axes(xlValue)
      .MinimumScale = 0  ' Beginn des Intervalls
      .MaximumScale = 100    ' Ende des Intervalls
      .TickLabels.NumberFormatLocal = "00"
   End With
```

Objekte

- .Axes(xlCategory) beschreibt die x-Achse (Rubrikenachse). Die Achse bezieht sich häufig auf die Spaltenüberschriften der Datenquelle.
- .Axes(xlValue) beschreibt die y-Achse (Größenachse). Die Achse stellt die möglichen Werte aus der Datenquelle dar. Welche Werte kann ein Element von der x-Achse annehmen?
- .Axes(...).Ticklabels beschreibt die Beschriftung der Teilstriche.

Beispiel für die Nutzung der Achsen

- Darstellung von Niederschlagsmengen pro Monat.
- Die Niederschlagsmengen werden durch die Monate zeitlich eingeteilt. Mit Hilfe der Monatsangaben werden die Niederschlagsmengen in Rubriken eingeteilt.
- Die Niederschlagsmengen sind Werte, die sich in der Datenreihe widerspiegeln sollen. Niederschlagsmengen sind Größenangaben. Die Werte werden auf der y-Achse aufgetragen.

Eigenschaften

- .MinimumScale gibt den Minimalwert des Intervalls an.
- .MaximumScale gibt den Maximalwert des Intervalls an.
- .TickLabels.NumberFormatLocal gibt ein Zahlenformat für die Beschriftung der Teilstriche vor. Das Zahlenformat bezieht sich auf den Sprachraum des Benutzers.

Datenreihen formatieren

```
Sub DiagrammNeu()
  With ActiveChart.SeriesCollection(1)
      .Interior.Color = RGB(220, 220, 220) 'Innenfläche
      .Border.Color = RGB(0, 0, 0)  ' Rahmen
      .Name = "Sonnenschein-Dauer"  ' Name der Datenreihe
      .XValues = "=KlimaWerte!$B$4:$M$4"
   End With

   With ActiveChart.SeriesCollection(2)
      .Interior.Color = RGB(190, 190, 190)
      .Border.Color = RGB(0, 0, 0)
      .Name = "Niederschlag"
      .XValues = "=KlimaWerte!$B$4:$M$4"
   End With
```

Objekte

- ActiveChart.SeriesCollection enthält alle Datenreihen eines Diagramms.
- ActiveChart.SeriesCollection(1) bezieht sich auf die erste Datenreihe in dem Diagramm. In den runden Klammern wird ein Index für die Auflistung übergeben. Der Index identifiziert eindeutig eine Datenreihe.

Eigenschaften, die die Achse beeinflussen

- Die Eigenschaft .XValues legt die Beschriftung für die Rubrikenachse fest.

Eigenschaften des Balken-Diagramms

- Die Eigenschaft .Interior.Color legt die Füllfarbe der Balken fest.
- Die Eigenschaft .Border.Color legt die Rahmenfarbe fest.
- Die Eigenschaft .Name legt den Legenden-Eintrag für die Datenreihe fest.

Abstand zwischen den Balken

```
With ActiveChart
    .ChartGroups(1).Overlap = -30
End With
```

- .ChartGroups(1).Overlap positioniert Balken und Säulen.
- Der Wert -100 positioniert die Balken in einem Abstand einer Balkenbreite voneinander.
- Der Wert +100 legt die Balken übereinander.
- Der Wert 0 positioniert die Balken direkt nebeneinander.