R|R|Z|N
Regionales Rechenzentrum für Niedersachsen

Leibniz Universität Hannover

Excel – Automatisierung von Arbeitsschritten
Sparklines automatisiert erstellen

Sparklines ...

- sind kleine Diagramme als Hintergrund für eine Zelle.
- werden direkt in einer Zelle in einem Arbeitsblatt eingebettet.
- beziehen sich standardmäßig immer nur auf eine Datenreihe.
- bieten die Möglichkeit Trends einer Datenreihe abzubilden.
- visualisieren minimale und maximale Werte.
- werden immer direkt neben oder unter den dargestellten Daten eingebunden.
- beziehen sich immer auf eine Datenreihe in einer Spalte oder Zeile.
- können mit Excel 2010 erstellt werden.

Arbeitsschritte für die Aufzeichnung

- Das Menüband Einfügen ist geöffnet.
- Es wird eine Zelle ausgewählt, in dem die Sparkline eingebettet wird.
- Klick auf das Symbol *Linie* in der Gruppe Sparklines.
- In dem Dialog Sparklines erstellen wird der darzustellende Datenbereich angegeben. Für den Positionsbereich wird automatisch die markierte Zelle genutzt.
- Der Dialog wird mit *OK* geschlossen und die Sparkline wird in der gewählten Zelle dargestellt.

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

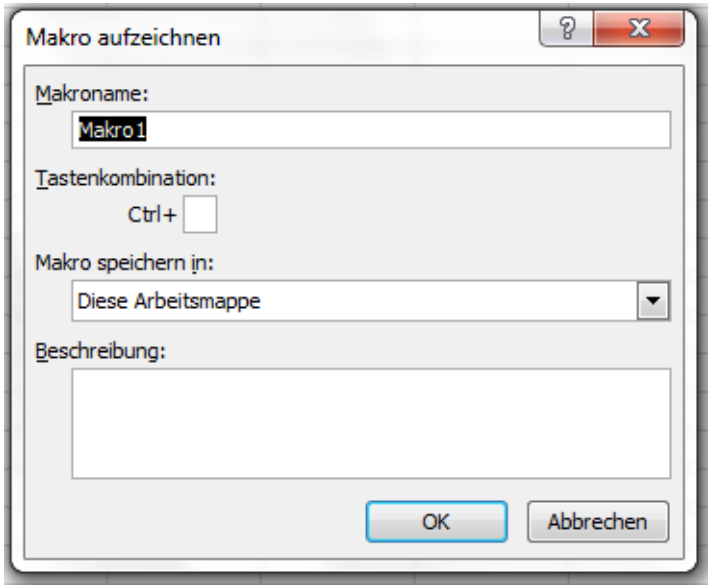
- *Datei – Optionen*.
- *Menüband anpassen*.
- In der rechten Liste Menüband anpassen werden alle Hauptregisterkarten angezeigt. Das Menüband Entwicklertools ist standardmäßig ausgeblendet (Kästchen ist leer).
- Durch einen Mausklick links von der Bezeichnung des Menübandes wird dieses aktiviert. In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

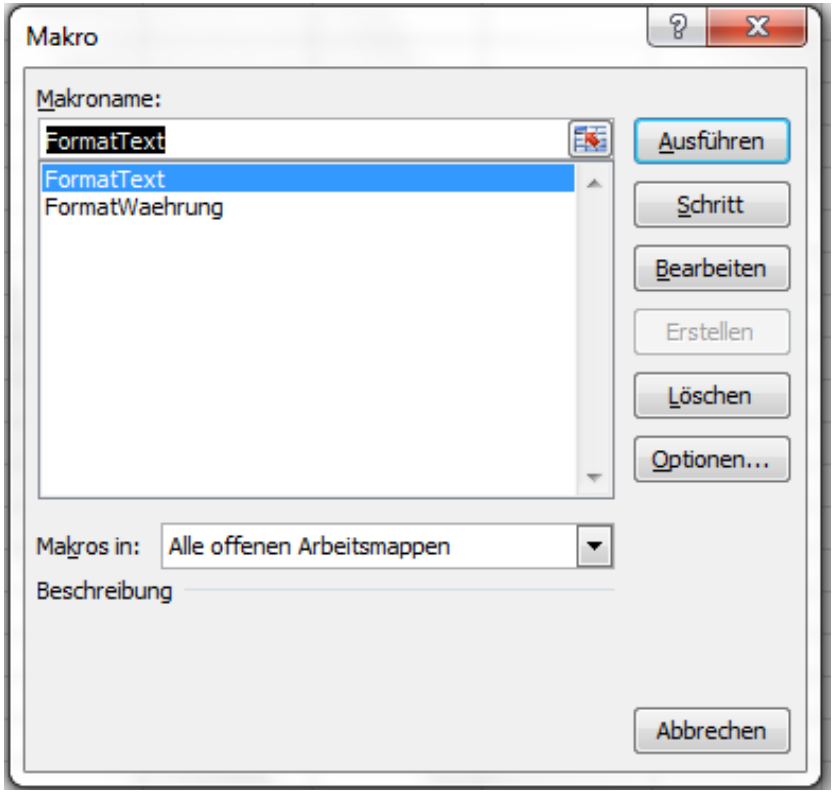
- automatisiert eine Folge von Arbeitsschritten.
- ist eine Programmiersprache, die in jeder Office-Anwendung eingebettet ist.
- erweitert die Funktionalität von Office-Anwendungen.
- passt eine Anwendung entsprechend der Wünsche des Benutzers an.

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.
- Hinweis: Die Sparkline wird immer für den markierten Zellbereich erstellt. Falls kein Zellbereich markiert ist, wird eine „leere“ Sparkline erzeugt.

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>+<gewählte Taste> startet das Makro.

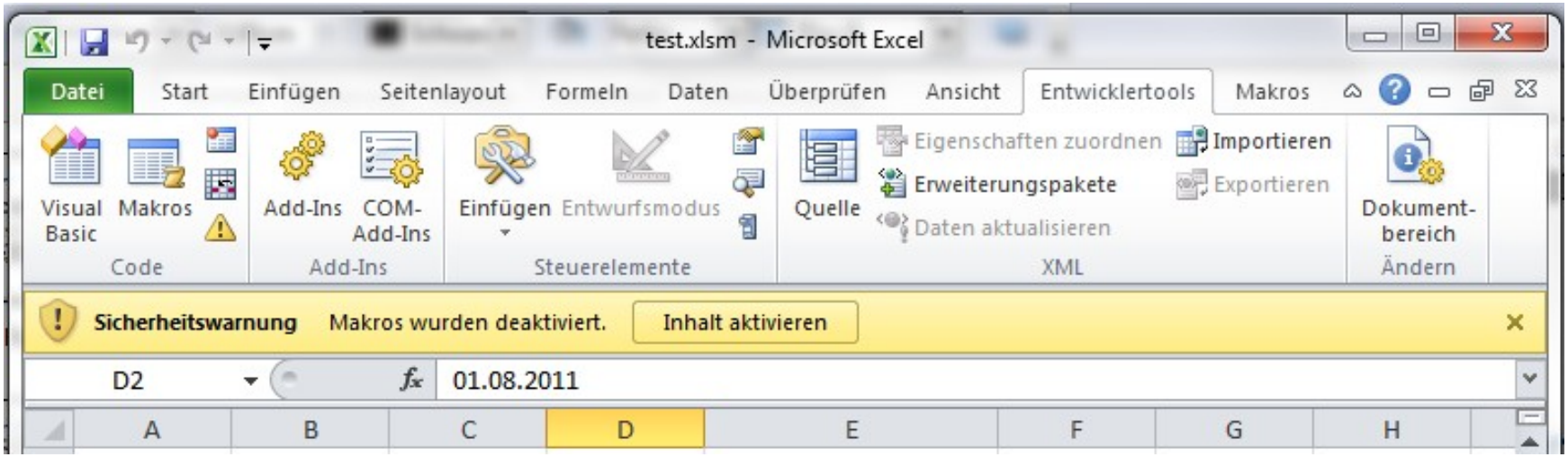
… über ein Menüband starten

- 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.

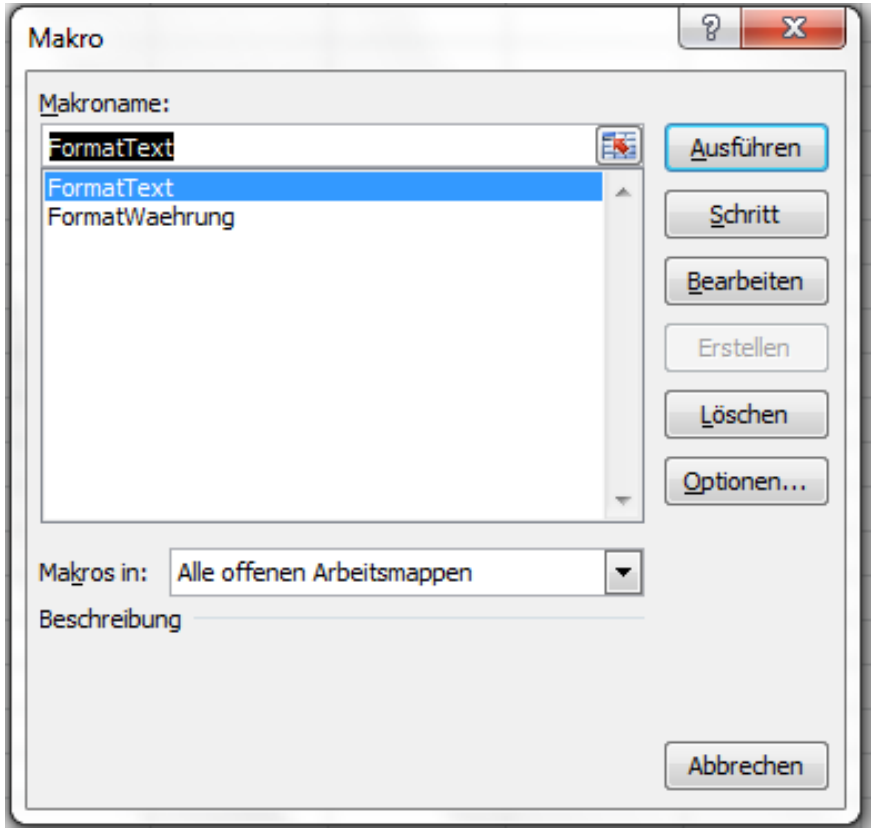
Arbeitsmappe mit einem Makro öffnen



- *Datei – Öffnen*.
- Beim erstmaligen Öffnen wird die Sicherheitswarnung angezeigt.
- Durch ein Klick auf die Schaltfläche *Inhalt aktivieren* werden die Makros aktiviert.

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 Sparklines()
    Range("N5").Select

    Range("$N$5").SparklineGroups.Add Type:=xlSparkLine, SourceData:="B5:M5"
    Selection.SparklineGroups.Item(1).SeriesColor.ThemeColor = 5
    Selection.SparklineGroups.Item(1).Points.Negative.Color.ThemeColor = 6
    Selection.SparklineGroups.Item(1).Points.Markers.Color.ThemeColor = 5
    Selection.SparklineGroups.Item(1).Points.Highpoint.Color.ThemeColor = 5
    Selection.SparklineGroups.Item(1).Points.Lowpoint.Color.ThemeColor = 5
    Selection.SparklineGroups.Item(1).Points.Firstpoint.Color.ThemeColor = 5
    Selection.SparklineGroups.Item(1).Points.Lastpoint.Color.ThemeColor = 5
End Sub
```

Objekte in Excel...

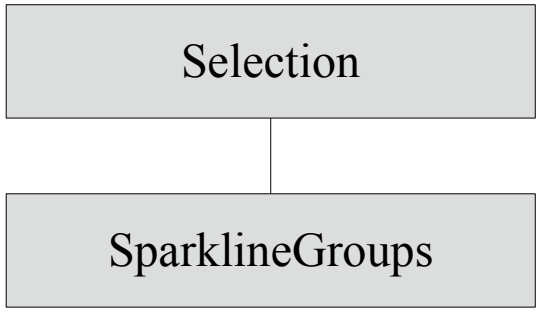
- sind zum Beispiel ein Zellbereich auf ein Tabellenblatt, eine Sparkline in einer Zelle.
- haben Eigenschaften (Attribute). Attribute beschreiben das Aussehen und das Verhalten eines Objekts.
- haben Methoden. Methoden sind vordefinierte Funktionen, die die Eigenschaften beeinflussen.
- reagieren mit Hilfe von Ereignisprozeduren auf Benutzeraktionen. Zum Beispiel: Der Benutzer ändert den Inhalt einer Zelle. Es wird das Ereignis „Arbeitsmappe geändert“ ausgelöst. In der dazugehörigen Ereignisprozedur kann überprüft werden, ob die Änderung korrekt ist.

... in dem Code-Beispiel

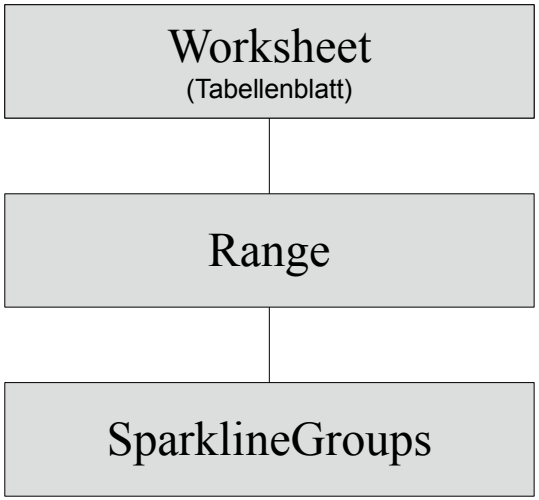
- Range: Zelle, Spalte, Zeile oder Zellbereich in einem Tabellenblatt.
- Selection: Das gewählte Objekt in der Excel-Anwendung.
- SparklineGroups: Auflistung von Gruppen von Sparklines.

Hierarchie von Objekten

Selection.SparklineGroups



Range("O5").SparklineGroups



- Selection oder Range sind Eltern-Objekte von SparklinesGroups.
- Die Aneinanderreihung bildet die Hierarchie der Objekte untereinander ab.
- Der Punkt verbindet das Kind-Objekt mit dem Eltern-Objekt.

Range() ...

- beschreibt eine Zelle, Zellbereich, Spalte oder Zeile auf einem Tabellenblatt.
- bekommt in runden Klammern ein Zellbezug übergeben. Dieser Zellbezug wird als Text angegeben. Zum Beispiel:
  - Range("A1") bezieht sich auf die Zelle A1.
  - Range("A1:D5") bezieht sich auf einen zusammen-hängenden Zellbereich.
  - ActiveCell verweist auf die momentan aktive Zelle.

… auf dem Tabellenblatt xyz

- Range("A1") nutzt die erste Zelle im aktiven Tabellenblatt.
- ActiveSheet.Range("A1:D5"). Die Nutzung des aktiven Tabellenblattes wird explizit angegeben.
- Worksheet("Tabelle1").Range("A1:D5"). In diesem Beispiel wird der Zellbereich „A1:D5“ auf dem Tabellenblatt Tabelle1 genutzt.

Methoden für den Zellbereich

- Range("A1").Select markiert den angegebenen Bereich. In diesem Beispiel wird die Zelle A1 ausgewählt.
- Range("A1").Clear löscht den Inhalt des angegebenen Bereichs.

Hinweise zu Methoden

- Methoden verändern die Attribute eines Objekts.
- Der Punkt verbindet die Methode mit dem dazugehörigen Objekt.
- Sobald der Methodenname vollständig ausgeschrieben ist, werden in einem Erklärfenster die Argumente der Methode angezeigt. Argumente sind Eingabeparameter für eine Methode.

Objekte für Sparklines

- beziehen sich immer auf ein Zellbereich Range.
- Sparkline enthält Methoden und Eigenschaften einer Sparkline.
- SparklineGroup enthält die Methoden und Eigenschaften einer Gruppe von Sparklines. Mit Hilfe des Befehls *Gruppieren* im Kontextmenü einer Sparkline werden verschiedene Sparklines zu eine Gruppe zusammengefasst.
- SparklineGroups listet alle Gruppen auf.

Neue Gruppe von Sparklines hinzufügen

```
SparklineGroups.Add Type:=xlSparkLine, _
                    SourceData:="B5:M5"
```

- Die Methode .Add() fügt der Auflistung SparklineGroup eine neue Gruppe von Sparklines hinzu. Eine Gruppe besteht mindestens aus einer Sparkline.
- In diesem Beispiel werden neue Sparklines als Liniendiagramm erstellt.
- Die Eingabeparameter werden der Methode als benannte Argumente übergeben. Die Position der Parameter in der Liste spielt durch die Benennung keine Rolle.

Benannte Argumente ...

```
SparklineGroups.Add Type:=xlSparkLine, _
                    SourceData:="B5:M5"
```

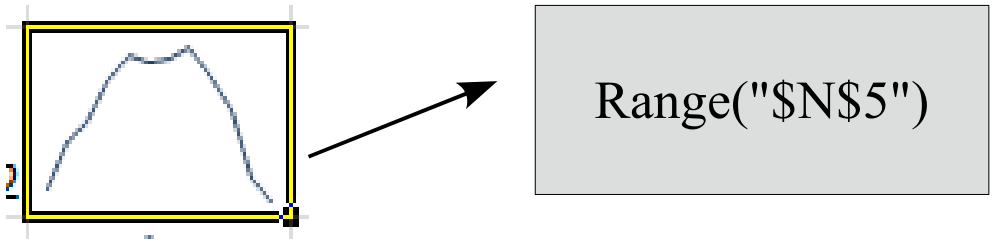
- weisen einem Platzhalter einem Wert zu.
- weisen mit Hilfe des zusammengesetzten Operators := einem Argument ein Wert zu.
- Die Namen der Argumente sind durch die Definition des Methodenkopfes vorgegeben. In der VBA-Hilfe werden die verschiedenen Methoden mit ihren Argumenten aufgelistet.

… für SparklineGroup.Add

- SourceData legt die Datenquelle für die Sparkline fest.
- Type legt mit Hilfe einer vordefinierten Konstante die Art der Sparkline fest.
- Durch die Nutzung der benannten Argumente kann die Angabe auf die benötigten Parameter beschränkt werden.

… in das Tabellenblatt einfügen

```
Range("$N$5").SparklineGroups.Add Type:=xlSparkLine, _
                                  SourceData:="B5:M5"
```



Elemente in der Auflistung

```
Selection.SparklineGroups.Item(1).SeriesColor.ThemeColor = 5
```

- SparklinesGroups.Item() bezieht sich auf ein Element in der Liste „Sparkline-Gruppe“.
- Mit Hilfe des Index in den runden Klammern wird exakt eine Gruppe identifiziert.
- In diesem Beispiel wird die erste Gruppe in der Auflistung angepasst.

Eigenschaften der Gruppe

- SeriesColor formatiert die Sparkline.
- Points legt das Aussehen der verschiedenen Datenpunkte fest. (siehe Kontrollkästchen *Sparklinetools – Entwurf*; Kategorie Anzeigen)

Nachteile des aufgezeichneten Codes

- Die Gruppe von Sparklines wird immer in der Zelle N5 eingebettet (Range("N5")).
- Die Darstellung der Sparkline-Gruppe wird immer in der Zelle O5 angepasst. Die Anweisung Selection.SparklinesGroups... bezieht sich immer auf den gewählten Zellbereich (Range("O5").Select).
- Die Datenquelle der Sparkline (SourceData:="B5:M5") kann nicht verändert werden.

Aktive Zelle nutzen

```
Sub Sparklines()

    ActiveCell.Select
    Selection.SparklineGroups.Add Type:=xlSparkLine, SourceData:="B6:M6"

End Sub
```

- Vor dem Start des Makros wird eine Zelle ausgewählt.
- In die ausgewählte Zelle wird das Marko eingefügt.

ActiveCell ...

- ist ein Synonym für die aktive Zelle in dem momentan aktiven Tabellenblatt.
- wird mit Hilfe der Methode .Select ausgewählt
- hat das Attribut .Address. Es wird die relative Position der aktiven Zelle zurückgegeben.

Selection ...

- bezieht sich immer auf den gewählten Bereich in der Anwendung.
- kann ein Zellbereich sein, muss aber nicht.

Platzhalter für Objekte nutzen

```
Sub Sparklines()
    Dim quelle As String
    Dim blatt As Worksheet
    Dim zelle As Range

    quelle = "B5:M5"
    ActiveCell.Select

    Set blatt = ThisWorkbook.ActiveSheet
    Set zelle = Application.ActiveCell
```

Objektvariablen ...

- sind Platzhalter für ein bestimmtes Excel-Objekt.
- können auf einen bestimmten Objekttyp verweisen.
- vom Typ
  - Workbook können einen Verweis auf eine Arbeitsmappe speichern.
  - Worksheet sind Platzhalter für eine bestimmtes Tabellenblatt in der angegebenen Arbeitsmappe.
  - Range verweisen auf einen definierten Zellbereich.

Objektvariablen definieren

```
Dim blatt As Worksheet
Dim variablenname As Objekttyp
```

- Der Name (blatt) der Objektvariablen ist frei wählbar.
- Mit Hilfe des Schlüsselwortes As wird der Objektvariablen ein bestimmter Objekttyp (hier: Worksheet) zugewiesen.
- Die Variable ist nur in dem Bereich bekannt, wo sie definiert ist. Das Schlüsselwort Dim regelt den Zugriff auf die Variable. In diesem Beispiel ist nur ein lokaler Zugriff möglich.

Hinweise zum Namen der Variablen

- Der Variablenname beginnt immer mit einem Buchstaben.
- Der Variablenname sollte nur aus den Zeichen A..Z, a..z, 0..9 und den Unterstrich zusammengesetzt werden.
- Der Name sollte sprechen. Das heißt, der Name spiegelt den Typ oder die Nutzung des gespeicherten Wertes / Verweises wieder.

Zuweisung an die Objektvariablen

```
Set blatt = ThisWorkbook.ActiveSheet
```

- Die Zuweisung muss mit dem Schlüsselwort `Set` eingeleitet werden.
- Mit Hilfe des Gleichheitszeichens wird der Variablen ein Verweis auf ein bestimmtes Objekt zugewiesen.
- Das zugewiesene Objekt und die Variable sollten vom gleichen Objekttyp sein.

Welches Tabellenblatt wird genutzt?

```
Sub Sparklines()
    Dim blatt As Worksheet

    Set blatt = ThisWorkbook.ActiveSheet
```

- ThisWorkbook ist ein Platzhalter für die aktuelle, aktive Arbeitsmappe in Excel. An dieser Arbeitsmappe hängt der Code.
- ActiveSheet ist ein Synonym für das momentan aktive Tabellenblatt.
- Jedes Tabellenblatt befindet sich in einer Arbeitsmappe. Arbeitsmappe und Tabellenblatt werden mit einem Punkt verbunden.

Nutzung eines Zellbereich

```
Sub Sparklines()
    Dim zelle As Range

    Set zelle = Application.ActiveCell
    zelle.Select
```

- Ein Verweis auf ein Zellbereich / eine Zelle kann in einer Variablen vom Typ Range gespeichert werden.
- In diesem Beispiel wird ein Verweis auf die aktive Zelle (ActiveCell) in der Anwendung (Application) übergeben.

Ersetzung der Variablenquelle durch eine Variable

```
Sub Sparklines()
    Dim quelle As String
    Dim blatt As Worksheet
    Dim zelle As Range

    quelle = "B5:M5"
    ActiveCell.Select

    Selection.SparklineGroups.Add Type:=xlSparkLine, SourceData:=quelle
```

Variablen ...

- sind Platzhalter für Zahlen und Zeichen im Speicher.
- können einen beliebigen Standard-Datentyp annehmen.
- haben einen eindeutigen Namen. Der Name sollte selbsterklärend sein.
- bekommen mit Hilfe des Gleichheitszeichen einen Wert zugewiesen.

... definieren

Dim quelle As String

Dim variablenname As Datentyp

- Jede Variable hat einen eindeutigen Namen (quelle), der mit einem Buchstaben beginnt.
- Jede Variable ist von einem bestimmten Typ. Der Typ wird mit Hilfe des Schlüsselwortes As zugewiesen. In diesem Beispiel kann die Variable Texte speichern. Für den angegebenen Datentyp wird Speicher bereit gestellt. Die Standarddatentypen werden in der Hilfe unter *Visual Basic-Sprachverzeichnis - Datentypen* aufgelistet.
- Mit Hilfe des Schlüsselwortes Dim wird der Nutzer der Variablen festgelegt. In diesem Beispiel kann die Variable nur in der Prozedur Sparklines genutzt werden.

Datentypen ...

- legen das Format des zu speichernden Wertes fest.
- legen Regeln für die Verwendung der Variablen fest.
- beschreiben die maximale Größe des Platzhalters.
- legen den Speicherbedarf fest.

... für Ganzzahlen

	Speicherbedarf in Bytes	Datenbereich
As Byte	1	0 - 255
As Integer	2	-32.768 - +32.767
As Long	4	-2.147.483.648 - +2.147.483.647
As Boolean	2	0 (falsch, false) <> 0 (wahr, true)

… für Dezimalzahlen

	Speicher-bedarf in Bytes	Datenbereich
As Single	4	-3,402823E38 bis -1,401298E-45 für negative Werte; 1,401298E-45 bis 3,402823E38 für positive Werte
As Double	8	-1,79769313486232E308 bis -4,94065645841247E-324 für negative Werte; 4,94065645841247E-324 bis 1,79769313486232E308 für positive Werte

… für Darstellung von Währungen

	Speicher-bedarf in Bytes	Datenbereich
As Currency	8	-922.337.203.685.477,5808 bis 922.337.203.685.477,5807

Genauigkeit von Dezimalzahlen

- As Single beschreibt ein Wert von einfacher Genauigkeit. Der Platzhalter enthält eine Näherung an eine reelle Zahl.
- As Double beschreibt ein Wert von doppelter Genauigkeit. Der Platzhalter enthält eine Näherung an eine reelle Zahl.
- As Currency besitzt 15 Stellen vor dem Dezimalkomma und vier Stellen nach dem Dezimalkomma.

Hinweise

- Als Dezimaltrennzeichen wird ein Punkt genutzt.
- Führende Nullen werden entfernt.
- Eine Null als Nachkommastelle wird durch das #-Zeichen ersetzt.
- Boolsche Variablen können die Schlüsselwörter True (Wahr, > 0) oder False (Falsch, = 0) zugewiesen werden.
- Das Programm kennt keine Maßeinheiten, Längenangaben etc.

… für Datums und Zeitwerte

	Speicherbedarf	Datenbereich
As Date	8 Bytes	1.Januar 100 – 31. Dezember 9999 00:00:00 - 23:59:59

Hinweise

datum = #2/23/2011#

uhrzeit = #1:20:00 PM#

- Konstante Datums- und Zeitwerte werden durch das #-Zeichen begrenzt.
- Datumswerte werden in dem Format Monat/Tag/Jahr angegeben. Es wird das kurze Datumsformat genutzt. Wenn möglich, sollten keine zweistelligen Jahresangaben genutzt werden.
- Zeitwerte werden in dem Format Stunde : Minute : Sekunde eingegeben. Zeitangaben werden in einem 12- oder 24-Stunden-Format in Abhängigkeit des Computers dargestellt.

… für Zeichenfolgen

	Länge
As String	Variable Länge.
As String * 5	Die Länge des Strings wird auf eine bestimmte Anzahl eingeschränkt. In diesem Beispiel besteht die Zeichenfolge aus fünf Zeichen.

Hinweise

ausgabeText = "Guten Tag"

- Zeichenfolgen werden durch Anführungsstriche begrenzt.
- Telefonnummern, Postleitzahlen werden in diesem Format gespeichert.
- Zeichenfolgen können Zahlen enthalten, mit denen nicht gerechnet wird.

Zuweisung an eine Variable

quelle = "B5:M5"

SourceData:=quelle

- Mit Hilfe des Gleichheitszeichen wird der definierten Variablen ein Wert zugewiesen.
- Dieser Wert kann ein konstanter Wert (quelle = "B5:M5") oder ein variabler Wert (quelle = aktiveZelle) sein.
- Der Wert rechts vom Gleichheitszeichen sollte den gleichen Datentyp wie die Variable links vom Gleichheitszeichen besitzen.

Einlesen der Datenquelle ...

- mit Hilfe eines vordefinierten Dialogs (InputBox). In dem Textfeld der InputBox gibt der Nutzer den Zellbereich ein.
- mit Hilfe eines benutzerdefinierten Dialogs (UserForm). Es wird der Einfüge-Dialog einer Sparkline nachgebildet.

Eingabefeld nutzen

```
Sub Sparklines()
    Dim quelle As String

    quelle = InputBox("Datenbereich: ")

    ActiveCell.Select
    Selection.SparklineGroups.Add Type:=xlSparkLine, _
                                  SourceData:=quelle

    Selection.SparklineGroups.Item(1).SeriesColor.ThemeColor = 5
End Sub
```

Datenquelle vom Bildschirm einlesen

quelle = InputBox("Datenbereich: ")

- Der Nutzer gibt in das Eingabefeld ein Text ein.
- Dieser eingegebene Text wird von der Funktion InputBox() zurückgegeben.
- Der Rückgabewert wird in der Variablen quelle gespeichert.

InputBox() ...

result = InputBox(prompt [, title] [, default]
[, xPos] [, yPos]
[, helpfile, context])

- blendet ein Fenster mit einem Textfeld zur Eingabe ein. Der Benutzer schreibt in das Feld mit Hilfe der Tastatur beliebigen Text.
- ist eine Funktion, der verschiedene Argumente übergeben werden können. Die Argumentliste wird durch die runden Klammern begrenzt. Die Argumente in der Liste werden durch Kommata getrennt.

Argumente der Funktion

- Der anzuzeigende Text prompt. Der Text informiert den Benutzer über die Eingabemöglichkeiten.
- Ein Text title für die Titelleiste des Fensters.
- Einen Standardwert default für das Textfeld festlegen. Dieser Wert wird nach dem Öffnen des Dialogs angezeigt, kann aber vom Nutzer überschrieben werden.

Rückgabewert der Funktion

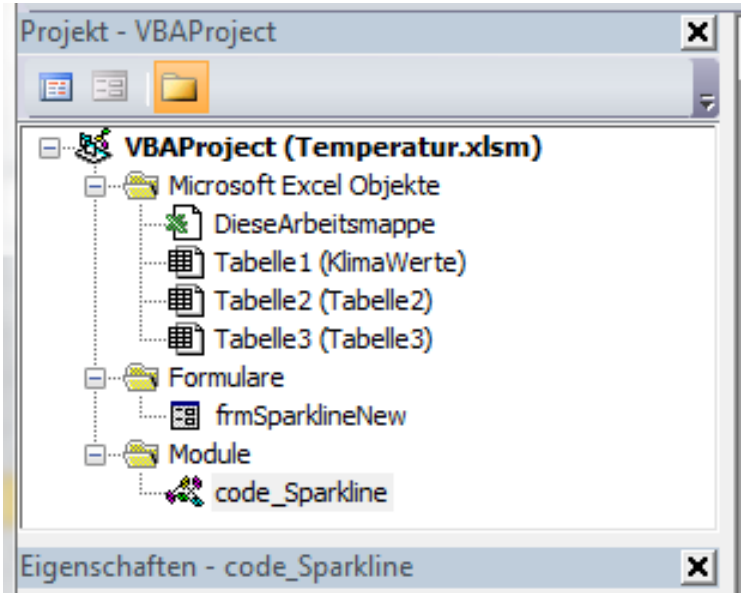
- Jede Funktion kann exakt einen Wert an den Aufrufer zurückgeben.
- Die InputBox() gibt den Inhalt des Textfeldes zurück.
- Der Rückgabewert ist vom Datentyp String (Text).

Benutzerdefinierte Dialoge ...

- können statt einer InputBox() genutzt werden.
- werden als UserForm bezeichnet.
- können über den VBA-Editor erstellt werden.

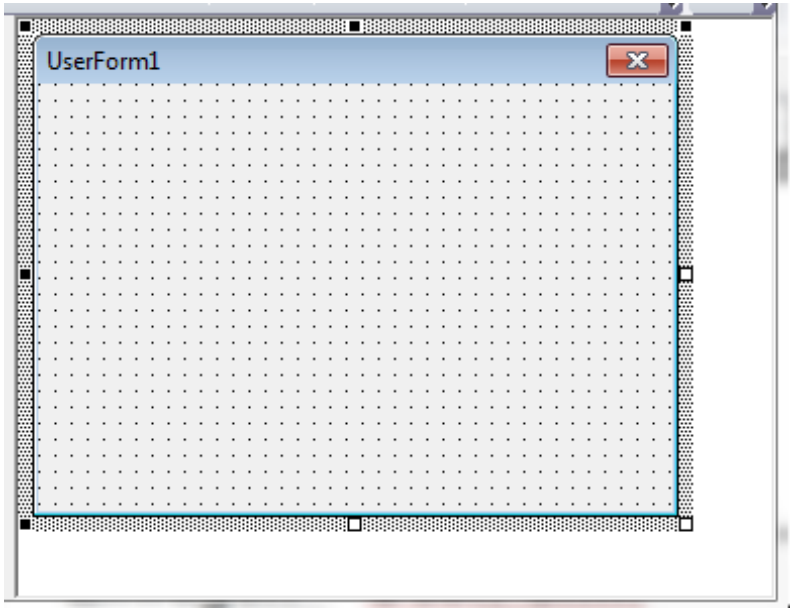
… im Projektexplorer des VBA-Editors

- Ordner Formulare zeigt alle benutzerdefinierten Dialoge (UserForms) an.
- Mit einem Doppelklick auf den Namen des Dialogs wird die Entwurfsansicht im VBA-Editor angezeigt.



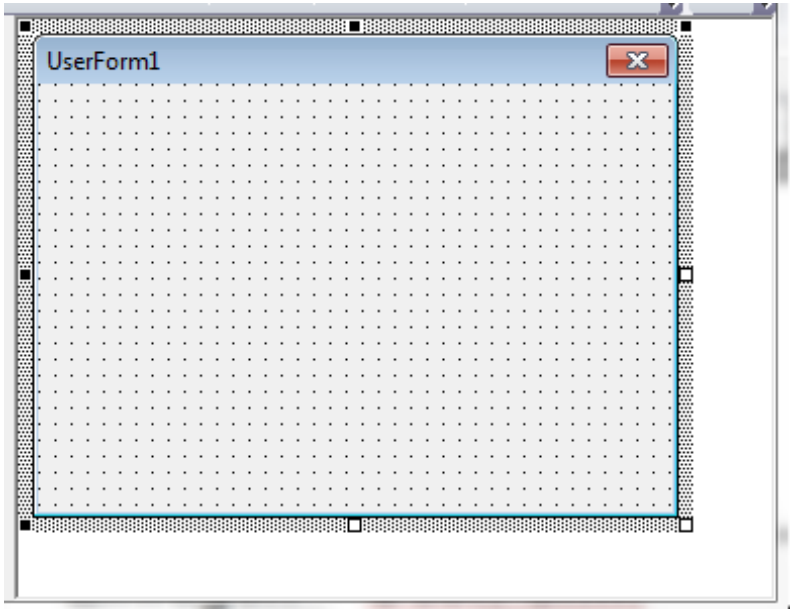
… neu erstellen

- *Einfügen – UserForm.*
- Statt dem Codefenster wird die Entwurfsansicht des Dialogs angezeigt. Es wird das Grundgerüst des Dialogs angezeigt.
- Das Eigenschaften-Fenster zeigt die Einstellungsmöglichkeiten für den Dialog an.
- Die Werkzeugsammlung zeigt Elemente, die in einem Dialog abgelegt werden können, an.



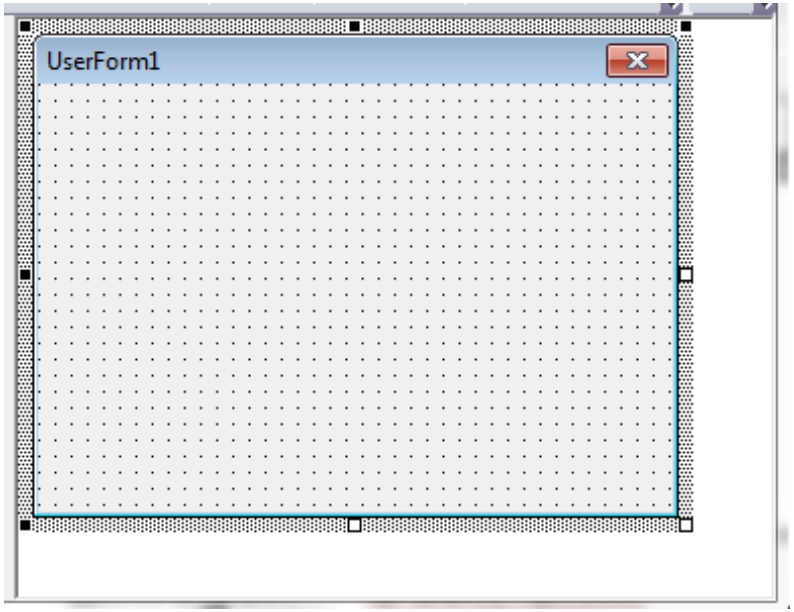
… in der Entwurfsansicht.

- Durch ein Mausklick auf die Titelleiste des Fenster wird der Dialog aktiviert
- Das aktive Element wird immer durch die dicke, gestrichelte Linie gekennzeichnet.
- Auf dieser Linie sowie in den Ecken befinden sich schwarze Quadrate (sogenannte Anfasser).



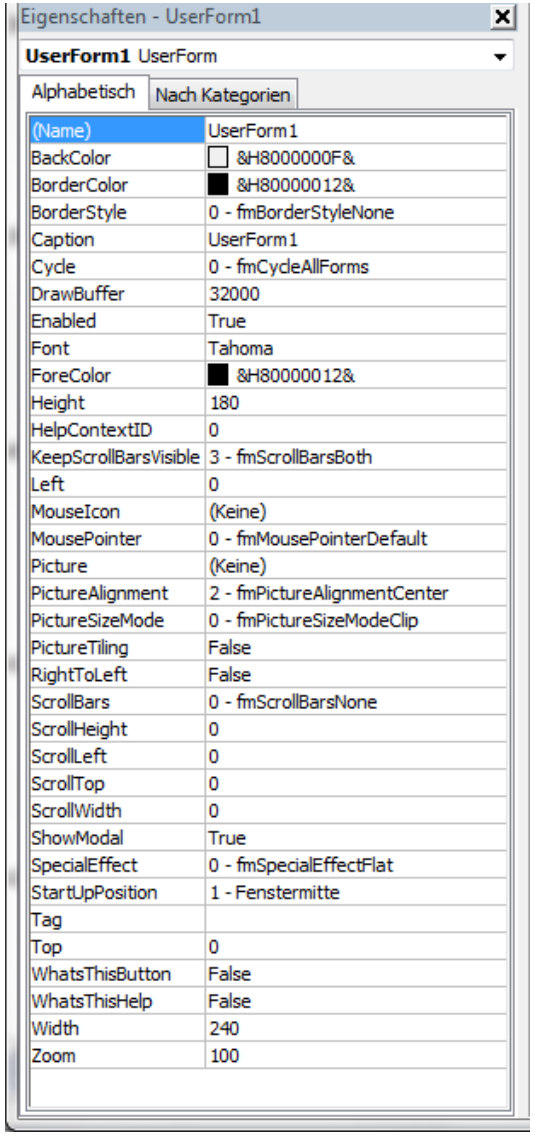
Größe des Dialogs ändern

- Der Dialog ist aktiv.
- Der Mauszeiger befindet sich über einen der schwarzen Quadrate.
- Mit Hilfe der gedrückt gehaltenen Maustaste wird das Fenster größer oder kleiner gezogen.
- Sobald die Maustaste losgelassen wird, wird die Größe des Dialogs entsprechend angepasst.



Eigenschaften (Attribute) ...

- beeinflussen das Aussehen und Verhalten des Dialogs.
- haben einen englischsprachigen Namen.
- haben einen bestimmten Wert. Für einige Eigenschaften gibt es Standardwerte.
- werden im Eigenschaftenfenster alphabetisch oder nach Kategorien sortiert, angezeigt.



Attributwerte werden ...

- mit Hilfe der Tastatur in ein Textfeld eingegeben.
- aus einer Liste eines Kombinationsfeldes ausgewählt. Die Liste wird durch den schwarzen Pfeil nach unten am rechten Rand des Feldes geöffnet. Das gewählte Element wird im Textfeld des Kombinationsfeldes angezeigt.
- mit Hilfe eines Assistenten eingestellt. Durch einen Mausklick auf Schaltfläche *3 Punkte* wird der passende Assistent geöffnet.

Eigenschaft „Name“ ...

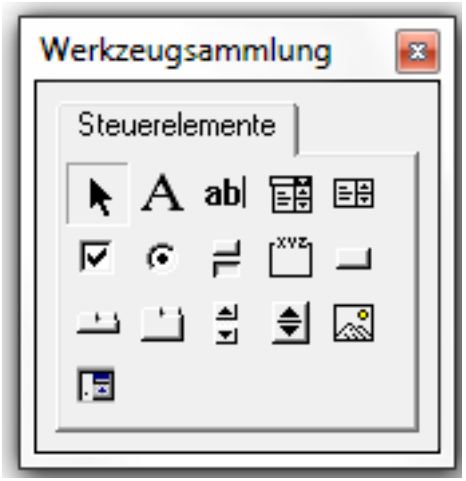
- identifiziert eindeutig ein Dialog oder ein Element, welches sich im Dialog befindet.
- hat einen Standardwert, der überschrieben werden sollte.
- ist ein selbsterklärender Namen. Die Aufgabe des Dialogs wird in dem Namen beschrieben.

… setzen

- Mausklick in das leere Feld rechts von der Bezeichnung (Name). Die Einfügemarke wird eingeblendet.
- Mit Hilfe der Tastatur wird ein passender Name für das gewählte Element eingegeben.
- Durch Wechsel in ein anderes Feld, wird die Eingabe gespeichert.

Werkzeugsammlung ...

- zeigt alle Elemente an, die in einem Dialog abgelegt werden können.
- enthält Werkzeuge zur Eingabe von Text.
- enthält Werkzeuge zur Auswahl von möglichen Werten.
- bietet Werkzeuge zur Gestaltung des Dialogs.



Beispiele für Werkzeuge

- Beschriftungsfelder geben Informationen zur Bedienung des Dialogs oder zum Ausfüllen von Feldern.
- In Textfeldern kann beliebiger Text mit Hilfe der Tastatur eingegeben werden.
- Listenfelder und Optionsfelder bieten eine Auswahl an Möglichkeiten.
- Kontrollkästchen bieten eine Mehrfachauswahl aus einer Vielzahl von Möglichkeiten.
- Befehlsschaltflächen starten eine Aktion.

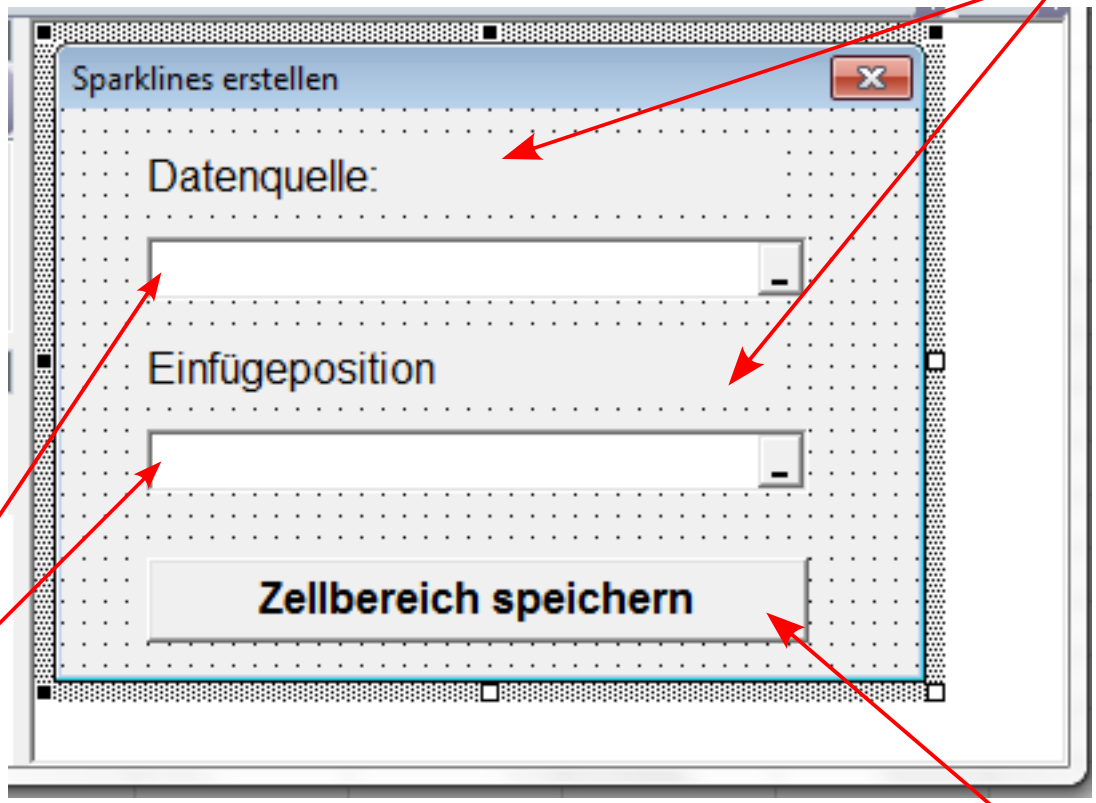
… auf einer UserForm ablegen

- Mausklick auf das gewünschte Werkzeugsymbol in der Werkzeugsammlung.
- Anschließend schwebt der Mauszeiger über den Entwurf des Dialogs.
- Mit Hilfe der gedrückt gehaltenen Maustaste wird ein Auswahlrahmen in der gewünschten Größe aufgezogen.
- Sobald die Maustaste losgelassen wird, wird das Werkzeug entsprechend der Größe des Auswahlrahmens eingefügt.

Dialog für das Einfügen von Sparklines

- Mit Hilfe eines Beschriftungsfeldes wird die Nutzung der Eingabefelder erläutert.
- Das Werkzeug RefEdit ...
  - wird zur Eingabe der Datenquelle und der Einfügeposition genutzt.
  - bietet die Möglichkeit Verweise auf Zellen zu bearbeiten.
  - übernimmt automatisch die Adresse des markierten Zellbereichs.
- Mit Hilfe einer Befehlsschaltfläche werden die Angaben in den Code übernommen und gespeichert.

Dialog


Beschriftungsfelder
Sparklines erstellen
Datenquelle:
Einfügeposition
Zellbereich speichern
Werkzeug „RefEdit“
Befehlsschaltfläche

Eigenschaften der UserForm

- Name legt die Bezeichnung für den Dialog in dem Projekt-Explorer fest.
- Caption legt den Text für die Titelleiste des Dialogs fest.
- BackColor verändert die Hintergrundfarbe des Dialogs.

Eigenschaften des Beschriftungsfeldes

- Name enthält eine eindeutige Bezeichnung für das Beschriftungsfeld.
- Caption legt den, im Beschriftungsfeld, angezeigten Text fest.
- Font legt die Schriftart fest.
- ForeColor beeinflusst die Schriftfarbe.

Eigenschaften für das RefEdit

- Name enthält eine eindeutige Bezeichnung für das Steuerelement RefEdit.
- BackColor legt die Hintergrundfarbe in Abhängigkeit von BackStyle fest. Bei einem Wert 0 wird ein transparenter Hintergrund genutzt.
- Font und ForeColor beeinflussen die genutzte Schrift.
- BorderStyle und BorderColor beeinflusst das Aussehen des Rahmens. Wenn aber die Eigenschaft SpecialEffect gesetzt ist, werden die Rahmeneigenschaften nicht beachtet.

Eigenschaften der Befehlsschaltfläche

- Name enthält eine eindeutige Bezeichnung für die Befehlsschaltfläche. Dieser Name wird in VBA zur Identifizierung des Steuerelements genutzt.
- BackColor legt die Hintergrundfarbe in Abhängigkeit von BackStyle fest. Bei einem Wert 0 wird ein transparenter Hintergrund genutzt.
- Font und ForeColor beeinflussen die genutzte Schrift.
- Caption legt die Beschriftung der Befehlsschaltfläche fest. Die Beschriftung gibt über die zu startende Aktion Auskunft.

Mausklick ...

- ist ein Ereignis für eine Befehlsschaltfläche.
- ist das Standardereignis einer Befehlsschaltfläche.
- wird vom Nutzer der UserForm ausgelöst.

Ereignisse

- sind immer mit einem bestimmten Objekt verbunden.
- repräsentierten eine Aktivität des Benutzers.
- werden im Kombinationsfeld Prozedur des Codefensters dargestellt.
- werden in Ereignisprozeduren verarbeitet.

Ereignisprozeduren ...

- enthalten Code, der nur in Abhängigkeit eines Ereignisses gestartet werden kann.
- sind die Reaktion auf die Auslösung eines Ereignisses.
- müssen nicht für alle möglichen Ereignisse eines Objekts geschrieben werden.

Beispiel: Mausklick auf die Befehlsschaltfläche

```
Private Sub cmdSaveZellbereich_Click()
    code_Sparkline.datenbereich = RefDatenbereich.Value
    code_Sparkline.positionsbereich = RefPositionsbereich.Value

    Unload Me
End Sub
```

Erläuterung

- Auf ein Ereignis wird mit Hilfe einer Prozedur reagiert. Die Prozedur beginnt mit Sub und endet mit End Sub.
- Jede Prozedur hat einen eindeutigen Namen. Der Name einer Ereignisprozedur (cmdSaveZellbereich_Click) setzt sich aus dem auslösenden Objekt und dem Ereignis zusammen.
- Die Prozedur ist privat. Die Prozedur kann nur von der dazugehörigen Befehlsschaltfläche aufgerufen werden.

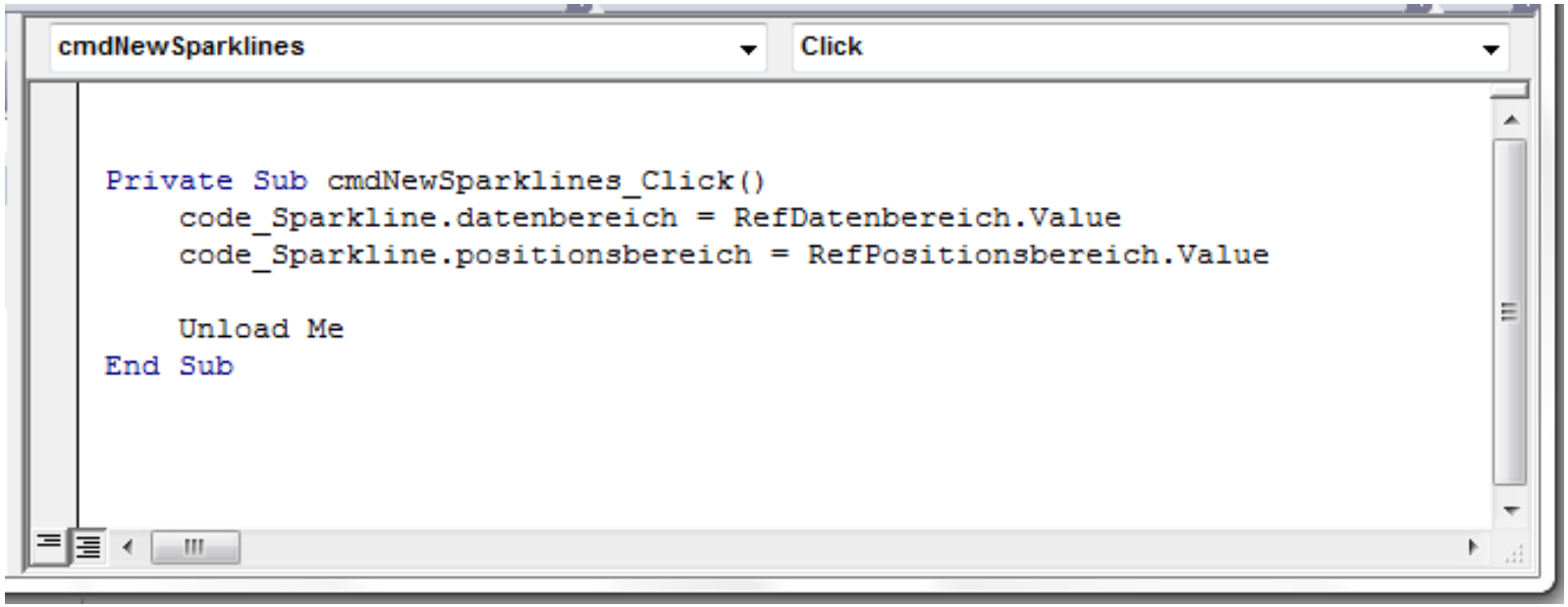
„Grundgerüst“ erstellen

```
Private Sub cmdSaveZellbereich_Click()


End Sub
```

- In dem Dialog wird eine Befehlsschaltfläche eingefügt.
- Durch ein Doppelklick auf die Befehlsschaltfläche wird das Codefenster geöffnet. Für das gewählte Werkzeug wird das Grundgerüst der Standard-Ereignisprozedur angelegt oder falls vorhanden, angezeigt.
- Durch einen Mausklick zwischen Sub und End Sub wird die Einfügemarke in die Ereignisprozedur gesetzt. Vorhandener Code kann verändert oder neuer eingegeben werden.

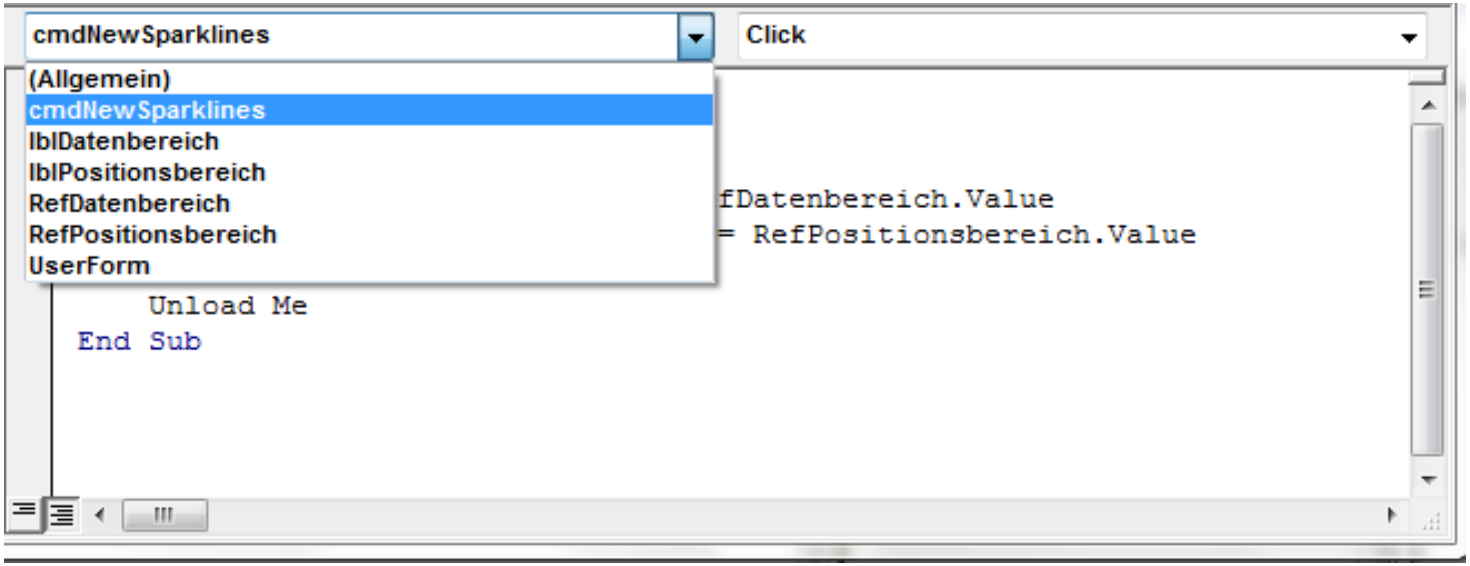
Andere Möglichkeit

- Das Codefenster ist geöffnet.
- Im Kombinationsfeld Objekt wird der Name des gewünschten Werkzeugs ausgewählt.
- Im Kombinationsfeld Prozedur wird ein Ereignis ausgewählt.



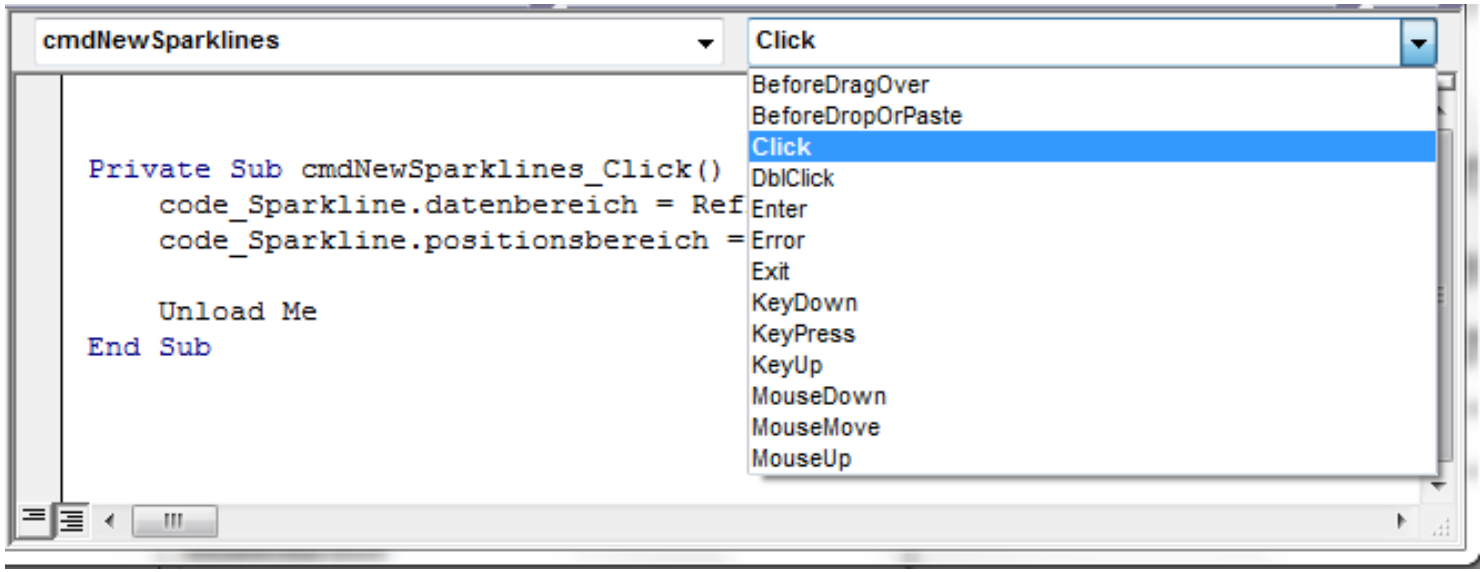
Kombinationsfeld „Objekt“ im Codefenster ...

- hat den Eintrag UserForm für den dazugehörigen Dialog.
- listet alle in dem Dialog abgelegten Werkzeuge namentlich auf.
- hat immer den Eintrag Allgemein. In diesem Bereich befinden sich Definitionen, die das gesamte Modul betreffen.



Kombinationsfeld „Prozedur“ im Codefenster ...

- listet alle Ereignisse des gewählten Objekts auf.
- Mit Hilfe eines Mausklicks wird ein Element ausgewählt und im Codefenster angezeigt. Falls die dazugehörige Ereignisprozedur nicht vorhanden ist, wird ein „Grundgerüst“ erstellt.



Anweisungen im Code-Beispiel

```
Private Sub cmdNewSparklines_Click()
    code_Sparkline.datenbereich = RefDatenbereich.Value
    code_Sparkline.positionsbereich = RefPositionsbereich.Value

    Unload Me
End Sub
```

- Sobald der Benutzer auf die Befehlsschaltfläche klickt, werden die Angaben im Werkzeug RefEdit in Variablen gespeichert und der Dialog geschlossen.

Wert eines Werkzeugs ...

RefDatenbereich.Value

RefPositionsbereich.Value

- wird in der Eigenschaft .Value abgelegt.
- wird zum Beispiel durch die Eingabe in ein Textfeld oder durch Auswahl eines Listenelements festgelegt.
- wird mit dem dazugehörigen Objekt durch ein Punkt verbunden.
- kann mit Hilfe von Code verändert oder gelesen werden.

... speichern

```
datenbereich = RefDatenbereich.Value
positionsbereich = RefPositionsbereich.Value
```

- Der Wert der Eigenschaft .Value wird einer Variablen zugewiesen.
- Die Eigenschaft steht rechts und die Variable links vom Gleichheitszeichen.
- Die Variable und der gespeicherte Wert der Eigenschaft sollten den gleichen Datentyp besitzen. Andernfalls versucht VBA den Wert automatisch in Abhängigkeit des Typs der Variablen zu konvertieren.

Variablen aus einen anderen Modul nutzen

Modul code_Sparkline

```
Public datenbereich As String
Public positionsbereich As String
```

UserForm ...

```
code_Sparkline.datenbereich = RefDatenbereich.Value
code_Sparkline.positionsbereich = RefPositionsbereich.Value
```

Öffentlichen Variablen

Public datenbereich As String

Public positionsbereich As String

- haben einen Zugriff Public. Die Variable kann von jeder Methode / Funktion / Prozedur in dem aktuellen Projekt genutzt werden.
- werden am Anfang eines Moduls definiert.
- haben einen eindeutigen Namen.
- bekommen mit Hilfe von As einen bestimmten Datentyp zugewiesen.

… nutzen

```
code_Sparkline.datenbereich = RefDatenbereich.Value
code_Sparkline.positionsbereich = RefPositionsbereich.Value
```

- Modul.Variable = Ausdruck.
- Jede öffentliche Variable wird in einem bestimmten Modul definiert. Das Modul und die darin definierte Variable werden mit einem Punkt verbunden.
- Zuerst wird der Modulname angegeben. Dann wird der Punkt gesetzt. VBA zeigt automatisch alle in dem Modul definierten öffentlichen Variablen in einer Liste an.

Dialog schließen

Unload Me

- `Me` ist ein Platzhalter für den, zum Code gehörenden Dialog (frmSparklineNew).
- Die Funktion Unload entfernt den angegebenen Dialog aus dem Speicher. Das Formular wird geschlossen.

Dialog öffnen

```
Sub SparklinesRelativ()
    Dim blatt As Worksheet
    Dim zelle As Range

    frmSparklineNew.Show

End Sub
```

- Die Methode Show des Dialogs lädt ein Dialog in den Speicher. Der Dialog wird am Bildschirm angezeigt.
- Die Methode und der Name des zu öffnenden Dialogs werden mit einem Punkt verbunden.

Nutzung einer Prozedur

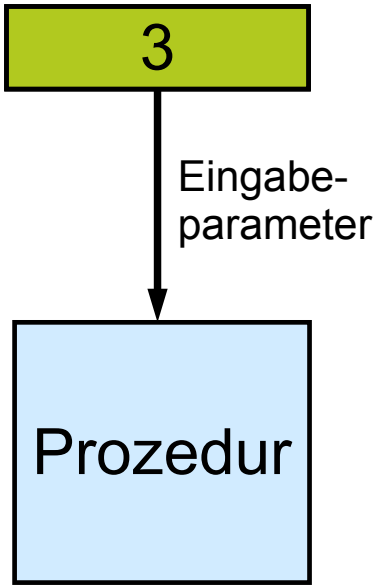
```
Sub SparklinesHinzufuegen(positionsbereich As String, datenbereich As String)
    Dim blatt As Worksheet
    Dim zelle As Range



    Set blatt = ThisWorkbook.ActiveSheet
    Set zelle = blatt.Range(positionsbereich)
    zelle.Select

    Selection.SparklineGroups.Add Type:=xlSparkLine, SourceData:=datenbereich
End Sub
```

Prozeduren ...

- beginnen in VBA mit Sub und enden mit End Sub.
- können auf ein Ereignis reagieren.
- werden vom Entwickler selber geschrieben.
- können mit Hilfe Ihres Namens aufgerufen werden.
- geben keinen Wert an den Aufrufer zurück.
- können mit Hilfe von Exit Sub vorzeitig beendet werden.

Arbeitsweise

- Der Prozedur können Eingabeparameter (Argumente) übergeben werden.
- Diese Eingabeparameter werden in der Prozedur verarbeitet.
- Die Verarbeitung selber ist dem Aufrufer aber nicht bekannt. Der Nutzer kennt nur die Schnittstelle (den Prozedurkopf) nach außen.



Prozedurkopf

```
Sub SparklinesHinzufuegen(positionsbereich As String, datenbereich As String)
Sub procName(arg01 As Typ, arg02 As Typ, ...)
```

- Der Prozedurkopf beginnt mit dem Schlüsselwort Sub.
- Dem Schlüsselwort folgt der Name der Prozedur. Der Name ist frei wählbar.
- In runden Klammern steht die Argumentliste. Die Argumentliste beschreibt die Anzahl und Art der Eingabeparameter. Die Argumente werden in der Liste durch Kommata getrennt.

Argumentliste ...

Sub SparklinesHinzufuegen(**positionsbereich As String,**
datenbereich As String)

- beginnt und endet mit den runden Klammern.
- steht direkt hinter dem Prozedur-Namen.
- kann aus beliebig vielen Argumenten bestehen. Die Argumente werden durch Kommata getrennt.
- kann leer sein. Der Prozedur werden keine Argumente übergeben.

Argumente ...

Sub Sparklines(**positionsbereich As String**,
datenbereich As String)

- symbolisieren einen Platzhalter für einen Wert, der der Prozedur übergeben wird.
- haben einen eindeutigen Namen.
- sind in der Prozedur gekapselt.
- bekommen mit Hilfe von As einen Datentyp zugewiesen. Es kann jeder beliebige Datentyp genutzt werden. Unter dem Menüpunkt *Visual Basic-Sprachverzeichnis – Datentypen* in der Hilfe im VBA-Editor werden alle vorhandenen Standardtypen aufgelistet.

Benutzerdefinierte Prozedur aufrufen

```
Private Sub cmdSaveZellbezug_Click()
    Call code_Sparkline.SparklinesHinzufuegen(RefPositionsbereich.Value,
                                              RefDatenbereich.Value)

    Unload Me
End Sub
```

Prozedur aufrufen

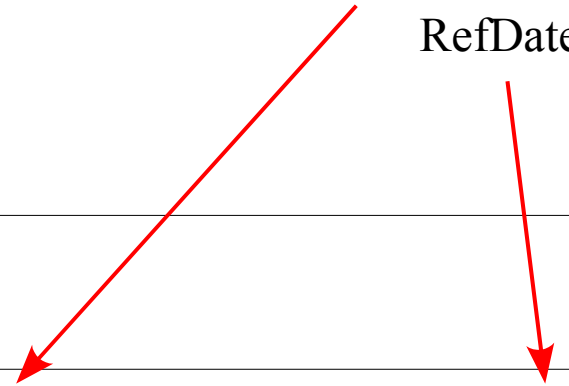
```
Call code_Sparkline.SparklinesHinzufuegen(RefPositionsbereich.Value,
                                          RefDatenbereich.Value)
```

- Mit Hilfe des Schlüsselwortes Call wird ein Prozedur-Aufruf gestartet.
- Die Prozedur wird mit ihren Namen aufgerufen (SparklinesHinzufügen).
- Die Prozedur SparklinesHinzufügen wird in dem Modul code_Sparkline definiert. Es muss exakt die Speicherposition der aufzurufenden Position angegeben.
- In den runden Klammern folgt eine Parameterliste.

Zuordnung der Parameter zu den Argumenten

UserForm ...

```
Private Sub cmdSaveZellbezug_Click()
    Call code_Sparkline.SparklinesHinzufuegen(RefPositionsbereich.Value,
                                              RefDatenbereich.Value)

End Sub
```



Modul code_Sparkline

```
Sub SparklinesHinzufuegen(positionsbereich As String, datenbereich As String)
    Dim blatt As Worksheet
    Dim zelle As Range

End Sub
```

Erläuterung

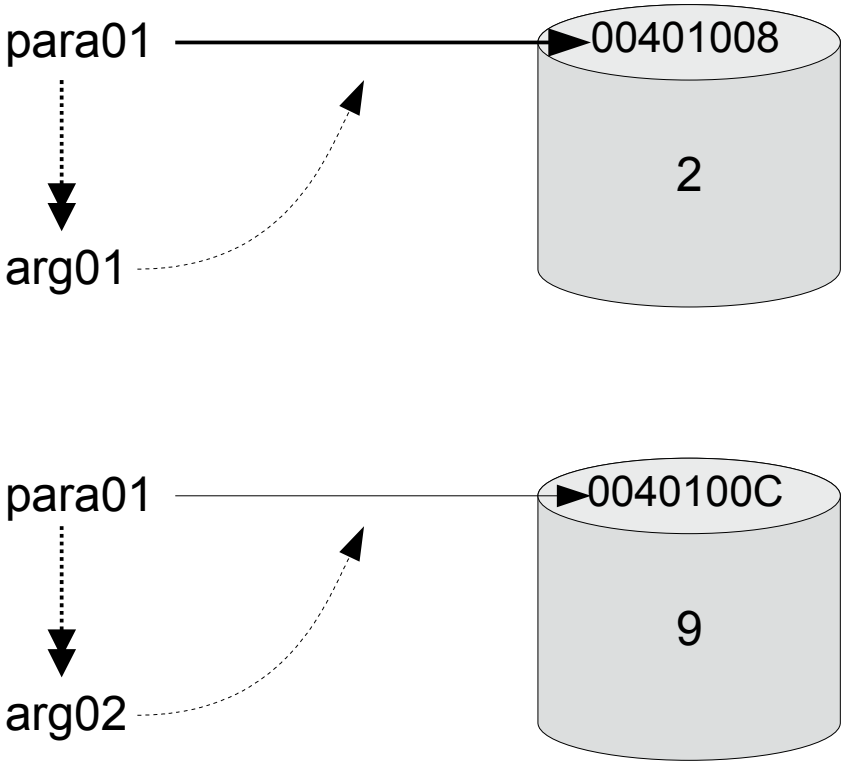
- Die Anzahl der Parameter im Aufruf entspricht der Anzahl der Elemente in der Argumentliste.
- Die Parameter werden den Argumenten von links nach rechts zugeordnet.
- Die Parameter werden den Argumenten in Abhängigkeit ihrer Position zugeordnet. Das heißt, der erste Parameter wird dem ersten Argument zugeordnet und so weiter.
- Um eventuelle Fehler auszuschließen, sollten der Parameter sowie das Argument den gleichen Datentyp besitzen.

Gleichen Speicherbereich für die Ablage nutzen

```
Sub SparklinesHinzufuegen(RefPositionsbereich.Value,
                          RefDatenbereich.Value)
Sub SparklinesHinzufuegen(ByRef RefPositionsbereich.Value,
                          ByRef RefDatenbereich.Value)
```

- Die Standardübergabe von Argumenten arbeitet mit Verweisen auf Speicherplätze. Die Position des Speicherplatzes wird durch den Parameter festgelegt.
- Das Argument bekommt die Position des Speicherplatzes übergeben.
- Der Wert des Parameters kann durch eine Zuweisung an das Argument ohne Prüfung verändert werden.

Grafische Darstellung

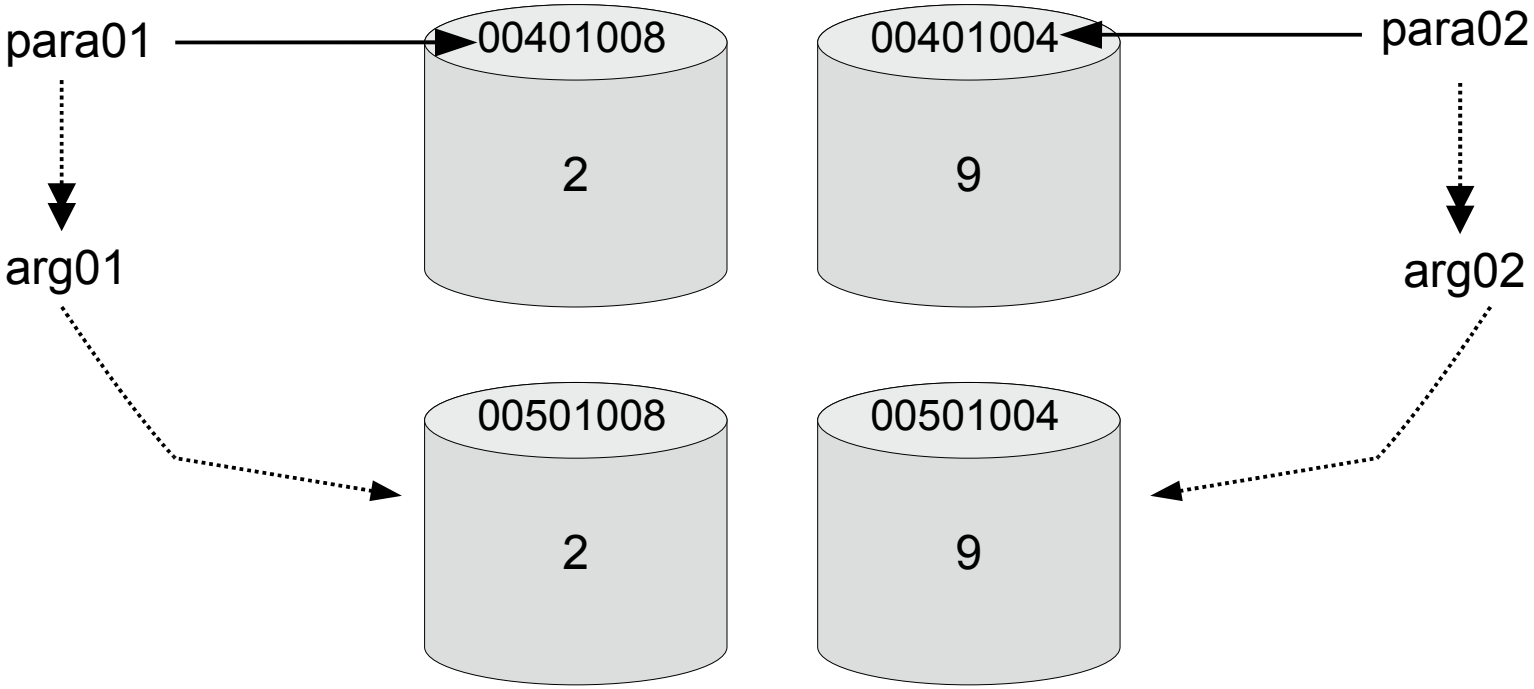


Schreibschutz für Parameter

```
Sub SparklinesHinzufuegen(ByVal RefPositionsbereich.Value,
                          ByVal RefDatenbereich.Value)
```

- Der zu übergebene Parameter verweist auf einen Speicherplatz, an dem ein bestimmter Wert abgelegt ist.
- Das Argument bekommt eine Kopie des Wertes übergeben. Der Platzhalter „Parameter“ wird kopiert.
- Vorteil: Die Parameter können nicht durch die aufgerufene Prozedur verändert werden. Der Wert des Parameters ist bei der Übergabe schreibgeschützt.

Grafische Darstellung



Nutzung von Argumenten

```
Sub SparklinesHinzufuegen(positionsbereich As String, datenbereich As String)
    Dim blatt As Worksheet
    Dim zelle As Range


    Set blatt = ThisWorkbook.ActiveSheet
    Set zelle = blatt.Range(positionsbereich)
    zelle.Select

    Selection.SparklineGroups.Add Type:=xlSparkLine, SourceData:=datenbereich
End Sub
```