OLE-Objekte

Autor & Copyright: Dipl.-Ing. Harald Nahrstedt

Version: 2016 / 2019 / 2021 / 365

Erstellungsdatum: 01.12.2011

Überarbeitung: 01.12.2023

Beschreibung:

Die Abkürzung OLE steht für *Object Linking and Embedding* und ist eine Bezeichnung für die Methoden, Objekte in eine Anwendung zu integrieren. Durch Verknüpfung mit der objekterzeugenden Anwendung (Linking) oder durch Einbettung (Embedding).

Bei der Verlinkung wird das integrierte Objekt nicht Bestandteil der Anwendung, sondern es existiert in der Anwendung nur ein Verweis auf das Original. Bei der Einbettung wird das Objekt durch Kopieren zum festen Bestandteil der Anwendung. Selbst wenn dann das Original verloren geht, ist das Objekt noch integriert.

Anwendungs-Datei: AE-016_OLEObjekte.xlsm

1 Strukturen

Die Strukturen von Verweis und Einbindung zeigt die nachfolgende Anwendung (Bild 1).

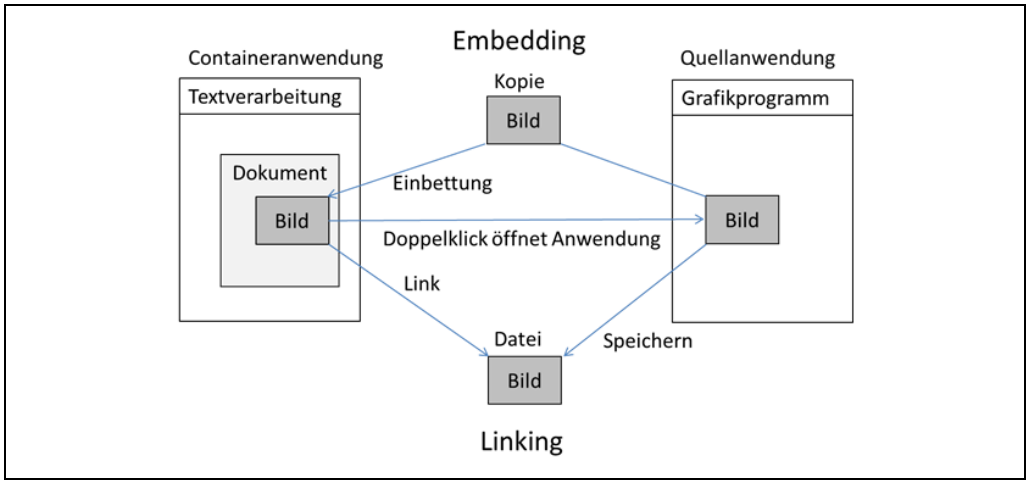

Bild 1. Schema zur Einbettung von Objekten

Zum OLE gehört die Methode *Drag & Drop*, bei der Objekte mit der Maus anwendungsübergreifend verschoben und kopiert werden können. Dazu gehört auch, dass die Attribute und Methoden der Objekte übertragen werden, so dass Objekte aus anderen Anwendungen gesteuert werden können. Verlinkte Quell-Objekte benötigen einen Namen zur Bindung.

In diesem Kapitel liegt der Focus auf die Anwendung unter Excel. Ein OLE-Objekt kann hier ein ActiveX-Steuerelement, ein verlinktes oder ein eingebettetes Objekt innerhalb eines Worksheets sein. Ein OLE-Objekt, als OLEObject bezeichnet, ist ein Element der OLEObjects-Auflistung. Mit Auflistungen hatten wir schon mehrfach zu tun und so wissen wir, dass sie über ihren Listenindex oder über ihre Namen angesprochen werden können.

2 ActiveX OLE-Objekte

Die nachfolgende Prozedur erzeugt eine Textbox auf dem Sheet *Tabelle1*. Verwendet wird dabei die Add-Methode.

Codeliste 1. Die Prozedur erzeugt eine Textbox auf Tabelle1

```
Sub OLE_TextBox_Neu()
   Worksheets("Tabelle1").OLEObjects.Add _
      ClassType:="Forms.TextBox.1"
End Sub
```

OLE-Objekte werden bezogen auf ein Worksheet, da sie ja eine Darstellungsfläche benötigen, als Objektliste verwaltet. So liest die nachfolgende Prozedur alle OLE-Objekte des Worksheets Tabelle1.

Codeliste 2. Die Prozedur zeigt alle OLE-Objekte in Tabelle1

```
Sub OLE_Textbox_Show()
   Dim oleBox As Object

   For Each oleBox In Worksheets("Tabelle1").OLEObjects
      MsgBox oleBox.Name
   Next
End Sub
```

Zu sehen sind die Namen in einer MessageBox (Bild 2).

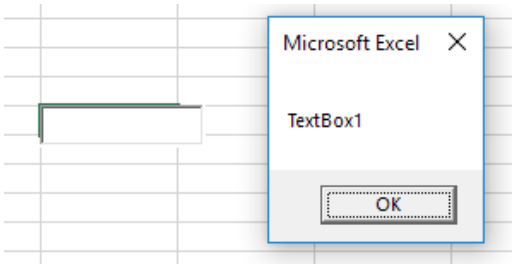

Bild 2. Anzeige der OLE-Objekte in Tabelle1

Die folgende Anweisung löscht die *Testbox1* wieder.

Codeliste 3. Die Prozedur zeigt alle OLE-Objekte in Tabelle1

```
Sub OLE_Textbox_Delete()
   Worksheets("Tabelle1").OLEObjects("TextBox1").Delete
End Sub
```

Die folgende Prozedur erzeugt eine Listbox und gibt ihr den Namen Datenliste.

Codeliste 4. Die Prozedur erzeugt eine Listbox

```
Sub OLE_Listbox()
   Dim oleListbox As Object

   Set oleListbox = Worksheets("Tabelle1").OLEObjects.Add _
      (classtype:="Forms.Listbox.1")
   With oleListbox
      .Name = "Datenliste"
   End With
End Sub
```

Die ActiveX-Steuerelemente werden im Register Entwicklertools, in der Gruppe Steuerelemente verwaltet. Dort lässt sich durch Klicken der Entwurfsmodus einschalten. Nach einem Klick auf die erstellte Listbox, steht in der Funktionszeile die Anweisung =EINBETTEN(„Forms.Listbox.1“;““). Im Namensfeld steht der Name *Datenliste* (Bild 3).

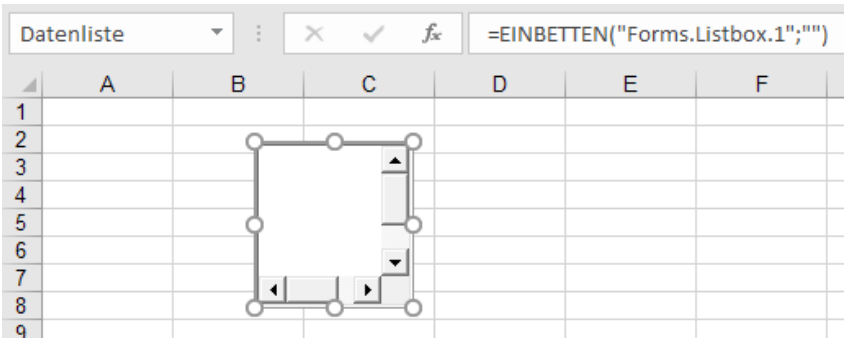

Bild 3. Datenliste auf Tabelle1

Ein ActiveX-Steuerelement verfügt über eine Menge von Eigenschaften, die durch den Aufruf Eigenschaften im Menüband Entwicklertools in der Gruppe Steuerelemente in einem Fenster angezeigt werden. Voraussetzung ist die Darstellung im Entwurfsmodus, in der dann das Steuerelement markiert wird (Bild 4).

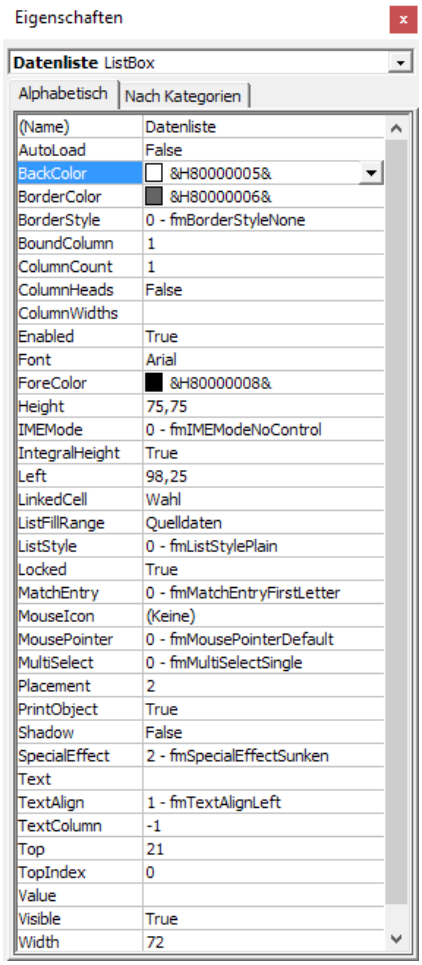

Bild 4. Eigenschaften der Datenliste

Wollen wir die Liste bei der Erzeugung bereits füllen, zum Beispiel mit Wochentags-Namen, so kann der Zellbereich unter dem Attribut *ListFillRange* angegeben werden. In diesem Beispiel schreiben wir die Wochentags-Namen in den Zellbereich A1:A7, und ergänzen die *With*-Anweisung um dieses Attribut.

Codeliste 5. Die Prozedur füllt die Datenliste mit einem Tabellenbereich

```
Sub
   With oleListbox
      .Name = "Datenliste"
      .ListFillRange = "A1:A7"
   End With
End Sub
```

Nach Ausführung stehen die Daten in der Liste (Bild 5).

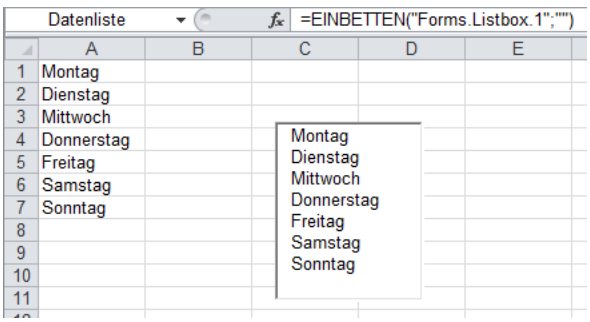

Bild 5. Datenliste mit Inhalt bereits beim Start

Ist der Entwurfsmodus ausgeschaltet, lässt sich jeder Eintrag in der Liste markieren. Nun wollen wir erreichen, dass ein markiertes Listenelement in einer Zelle (hier A10) eingetragen wird. Dazu besitzt die Listbox das Attribut *LinkedCell*, also eine Verlinkung. Die with-Anweisung erhält den Eintrag *LinkedCell*. Nach der Ausführung und einem Klick auf einen Listeneintrag, steht dieser dann in der besagten Zelle.

Noch ein wichtiger Tipp. Wenn Sie dem Zellbereich der Quelldaten und Zelle für das Ergebnis einen Namen geben, lassen sich die Anweisungen besser lesen, und die Bereiche dynamisch gestalten. Im Beispiel die Namen *Datenliste* und *Wahl*.

Codeliste 6. Die Prozedur nutzt Bereichsnamen zum Füllen der Datenliste

```
Sub
   With oleListbox
      .Name = "Datenliste"
      .ListFillRange = "Datenliste"
      .LinkedCell = "Wahl"
   End With
End Sub
```

Und ein solches Objekt besitzt natürlich noch besondere Methoden, die als Events bezeichnet werden. Eins davon ist der Mausklick. Dieser Event liefert in der nachfolgenden Weise, über eine Mitteilungsbox, das angeklickte Element der Liste.

Codeliste 7. Die Ereignis-Prozedur reagiert bei einem Klick auf die Datenliste

```
Private Sub Datenliste_Click()
   MsgBox "Auswahl " & Datenliste.List(Datenliste.ListIndex)
End Sub
```

Beispiel einer Auswahl aus der Datenliste (Bild 6).

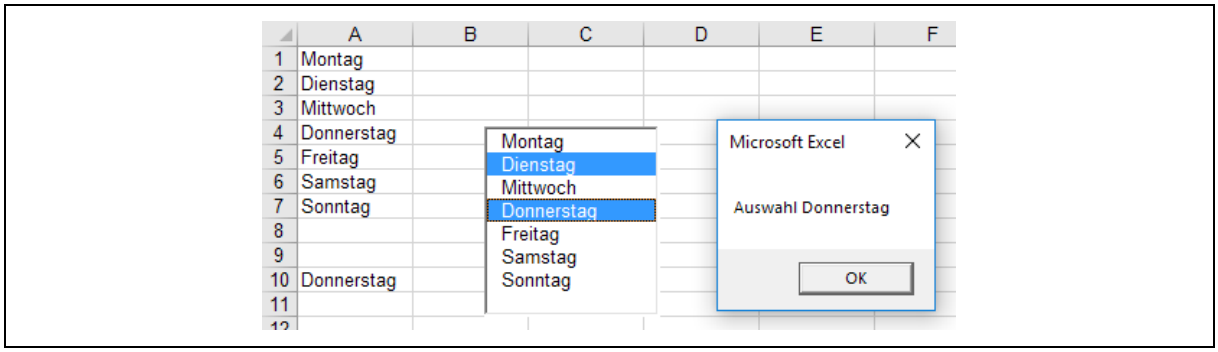

Bild 6. Beispiel zur Auswahl aus der Datenliste

3 Verlinkte OLE-Objekte

Die nachfolgende Prozedur erzeugt eine Einbettung von einem Bild auf dem Worksheet *Tabelle3*. Das Bild *Koala.png* befindet sich im Verzeichnis *C:\Temp*.

Codeliste 8. Die Prozedur bindet ein Bild ein

```
Sub BildEinbindung()
   Dim oleBild As OLEObject

   Set oleBild = Worksheets("Tabelle1").OLEObjects.Add _
      (Filename:="C:\Temp\Koala.bmp")
   oleBild.Name = "Koala"
End Sub
```

Die Einbindung des Bildes wird als Bitmap-Bild vorgenommen (Bild 7).

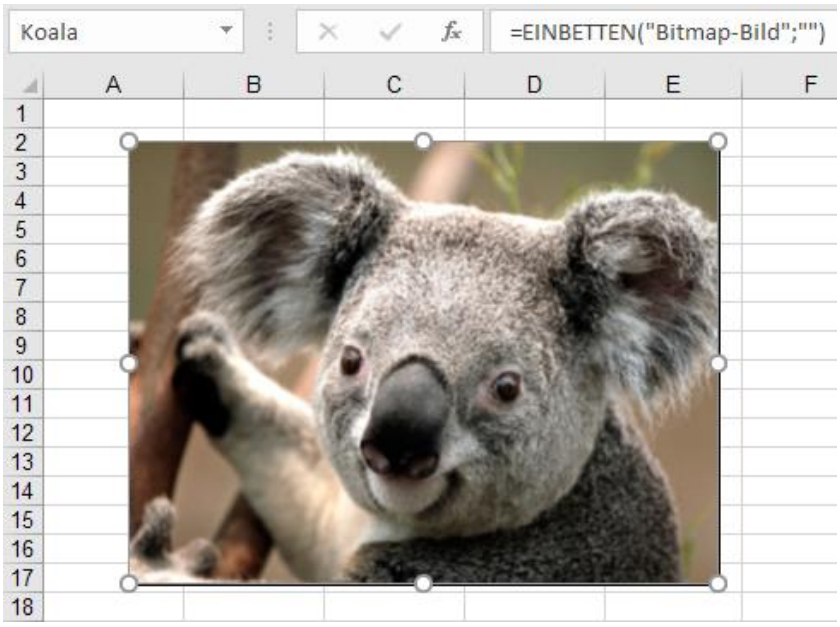

Bild 7. Einbettung als Bitmap-Bild

Die folgende Prozedur erzeugt einen Link zu demselben Bild. Dabei wird das Attribut Link auf *True* gesetzt. Das, wenn es nicht ausdrücklich gesetzt ist, auf *False* steht.

Codeliste 9. Die Prozedur bindet ein Bild als Link ein

```
Sub BildVerlinkung()
   Dim oleBild As OLEObject

   Set oleBild = Worksheets("Tabelle1").OLEObjects.Add _
      (Filename:="C:\Temp\Koala.bmp", link:=True)
   oleBild.Name = "Koala-Link"
End Sub
```

Damit stehen zwei Bilder auf der Tabelle1 (Bild 8).

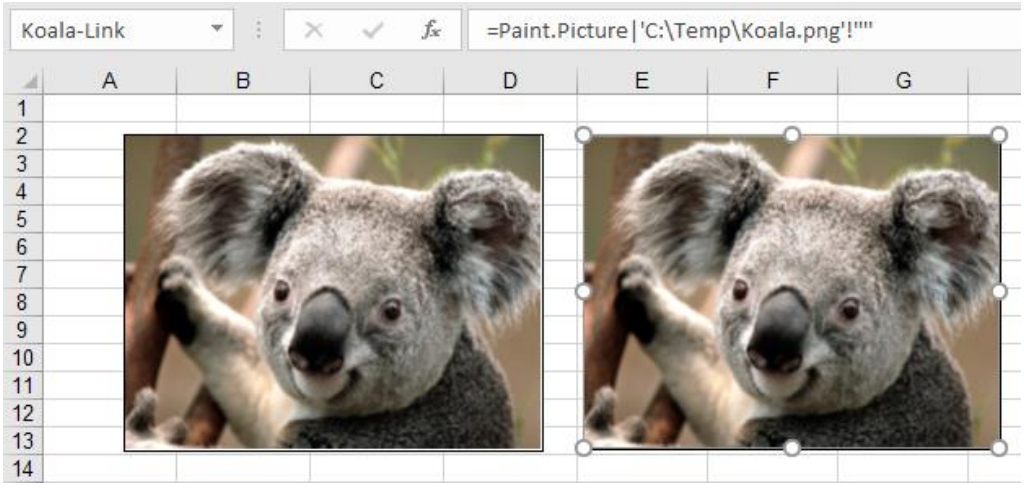

Bild 8. Beide Bilder auf Tabelle1

Ein Doppelklick auf beide Bilder öffnet die assoziierte Anwendung, meist ist es *Paint*.

4 Methoden der OLE-Objekte

Nachfolgend eine Übersicht aller Methoden, die die Klasse OLEObjects besitzt.

Tabelle 1. Methoden der Klasse OLEObjects

Name	Beschreibung
Add	Fügt einem Blatt ein neues OLE-Objekt hinzu.
BringToFront	Bringt das Objekt in den Vordergrund der Z-Reihenfolge.
Copy	Kopiert das Objekt in die Zwischenablage.
CopyPicture	Kopiert das ausgewählte Objekt als Bild in die Zwischenablage. (Variant)
Cut	Schneidet das Objekt aus und speichert es in der Zwischenablage.
Delete	Löscht das Objekt.
Duplicate	Dupliziert das Objekt und gibt einen Bezug auf die neue Kopie zurück.
Item	Gibt ein einzelnes Objekt aus einer Auflistung zurück.
Select	Markiert das Objekt.
SendToBack	Verschiebt das Objekt in den Hintergrund der Z-Reihenfolge.

5 Eigenschaften der OLE-Objekte

Nachfolgend eine Übersicht aller Attribute, die die Klasse OLEObjects besitzt.

Tabelle 2. Eigenschaften der Klasse OLEObjects

Name	Beschreibung
Application	Ohne Objektkennzeichner gibt diese Eigenschaft ein Application-Objekt zurück, das die Anwendung Microsoft Excel darstellt. Bei Verwendung mit einem Objektbezeichner gibt die Eigenschaft ein Application-Objekt zurück, das den Ersteller des angegebenen Objekts darstellt (diese Eigenschaft kann zusammen mit einem OLE-Automatisierungsobjekt verwendet werden, um die Anwendung des Objekts zurückzugeben). (Schreibgeschützt)
AutoLoad	True, wenn das OLE-Objekt beim Öffnen der Arbeitsmappe, in der es enthalten ist, automatisch geladen wird. (Boolean-Wert mit Lese-/Schreibzugriff)
Border	Gibt ein Border-Objekt zurück, das den Rahmen des Objekts darstellt.
Count	Gibt einen Long-Wert zurück, der die Anzahl der Objekte in der Auflistung darstellt.
Creator	Gibt den ganzzahligen 32-Bitwert zurück, der die Anwendung angibt, in der dieses Objekt erstellt wurde. (Schreibgeschützter Long-Wert)
Enabled	True, wenn das Objekt aktiviert ist. (Boolean-Wert mit Lese-/Schreibzugriff)
Height	Gibt einen Double-Wert zurück, der die Höhe des Objekts in Punkt darstellt, oder legt diesen fest.
Interior	Gibt ein Interior-Objekt zurück, das den Innenbereich des angegebenen Objekts darstellt.
Left	Gibt einen Double-Wert zurück, der den Abstand (in Punkt) von der linken Seite des Objekts zum linken Rand der Spalte A (in einem Arbeitsblatt) oder zur linken Seite des Diagrammbereichs (in einem Diagramm) darstellt, oder legt diesen fest.
Locked	Gibt einen Boolean-Wert zurück, der angibt, ob das Objekt gesperrt ist, oder legt diesen fest.
Parent	Gibt das übergeordnete Objekt für das angegebene Objekt zurück. Schreibgeschützt.
Placement	Gibt einen Variant-Wert zurück oder legt diesen fest, der eine XlPlacement-Konstante enthält, die darstellt, wie das Objekt an die Zellen darunter angefügt ist.
PrintObject	True, wenn das Objekt zusammen mit dem Dokument gedruckt wird. Boolean-Wert mit Lese-/Schreibzugriff.
Shadow	Gibt einen Boolean-Wert zurück, der bestimmt, ob das Objekt einen Schatten aufweist, oder legt diesen Wert fest.
ShapeRange	Gibt ein ShapeRange-Objekt zurück, das ein oder mehrere angegebene Objekte darstellt. Schreibgeschützt.

Name	Beschreibung
SourceName	Gibt einen String-Wert zurück, der den Namen der Verknüpfungsquelle des angegebenen Objekts darstellt, oder legt diesen fest.
Top	Gibt einen Double-Wert zurück, der den Abstand (in Punkt) vom oberen Rand des Objekts zum oberen Rand der Zeile 1 (auf einem Arbeitsblatt) oder bis zum oberen Rand des Diagrammbereichs (in einem Diagramm) darstellt, oder legt diesen Wert fest.
Visible	Gibt einen Boolean-Wert zurück, der bestimmt, ob das Objekt sichtbar ist, oder legt diesen Wert fest. (Lese-/Schreibzugriff)
Width	Gibt einen Double-Wert zurück, der die Breite des Objekts darstellt, oder legt diesen Wert fest.
ZOrder	Gibt die Position des Objekts in der Z-Reihenfolge zurück. Schreibgeschützter Long-Wert.