Excel + VBA

Application Events

Autor & Copyright: Dipl.-Ing. Harald Nahrstedt

Version: 2016 / 2019 / 2021 / 365

Erstellungsdatum: 25.03.2015

Überarbeitung: 01.12.2023

Beschreibung:

Dieses Kapitel erklärt noch einmal die Nutzung von Application-Events durch Objektvariable und ihre Klasse.

Anwendungs-Datei: AE-020_ApplicationEvents.xlsm

1 Das Application-Objekt

In einem Code-Fenster können die vorhandenen Objekte (Auswahlliste links) und ihre Events (Auswahlliste rechts) ausgewählt werden (Bild 1).

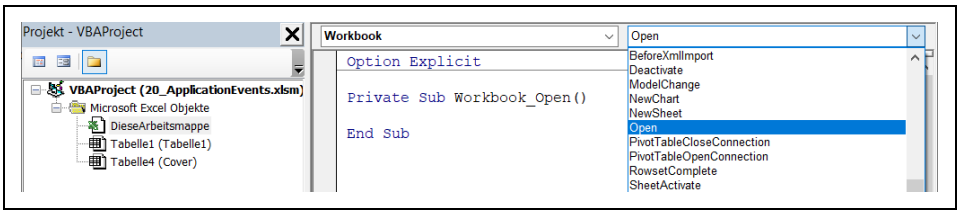

Bild 1. Das Workbook-Objekt und seine Ereignisse

Worksheet und Workbook verfügen über ein solches Code-Fenster, dass Application-Objekt jedoch nicht. Um auch an die Events des Application-Objekts zu kommen, bedient man sich eines Tricks. Man definiert eine Variable über die Anweisung *WithEvents*, sodass sie auch die Events des Application-Objekts bekommt, zu sehen im Code-Fenster des Workbooks (Bild 2).

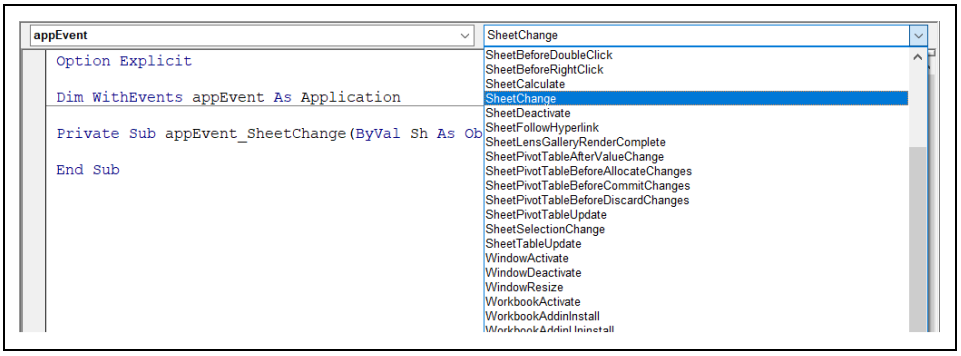

Bild 2. Die Ereignis-Prozedur SheetChange

Danach ist dieses Objekt anwählbar und besitzt die Ereignisse des Application-Objekts (Bild 3).

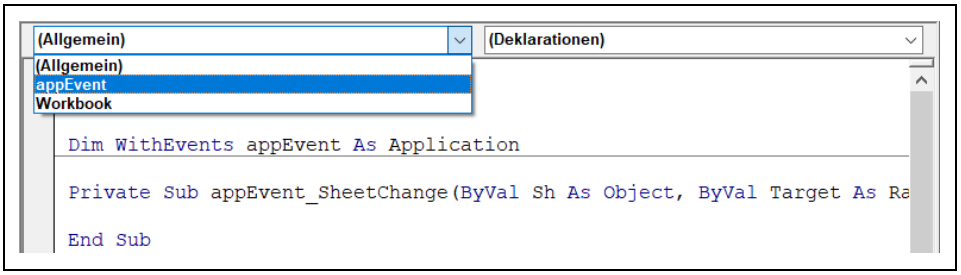

Bild 3. Das neue appEvent-Objekt

Diese Event-Prozedur lässt sich nun mit Funktionalität füllen, z. B. die Meldung, welches Sheet und welche Zelle geändert wurde.

Codeliste 1. Definition des Change-Ereignisses im Codefenster von DieseArbeitsmappe

```
Open Explicit

Dim WithEvents appEvent As Application

Private Sub appEvent_SheetChange _
   (ByVal Sh As Object, ByVal Target As Range)
   MsgBox "Sheet: " & Sh.Name & _
      " Adresse: " & Target.Address
End Sub
```

Die Konstruktion ist abgeschlossen und das Objekt deklariert. Das Event wirkt allerdings erst wenn es das Objekt *appEvent* auch gibt. Die Instanziierung des Objekts *appEvent* legt man üblicherweise in das *Workbook_Open* –Event. Damit werden mit dem Öffnen des Workbooks auch das *Application*-Objekt instanziiert und damit seine Events wirksam.

Codeliste 2. Erweiterung des Codes in DieseArbeitsmappe mit der Instanziierung des appEvent-Objekts

```
Option Explicit

Dim WithEvents appEvent As Application

Private Sub appEvent_SheetChange _
   (ByVal Sh As Object, ByVal Target As Range)
   MsgBox "Sheet: " & Sh.Name & _
      " Adresse: " & Target.Address
End Sub

Private Sub Workbook_Open()
   Set appEvent = Application
End Sub
```

Nach einem Neustart der Excel-Mappe werden alle Änderungen auf den Worksheets gemeldet (Bild 4).

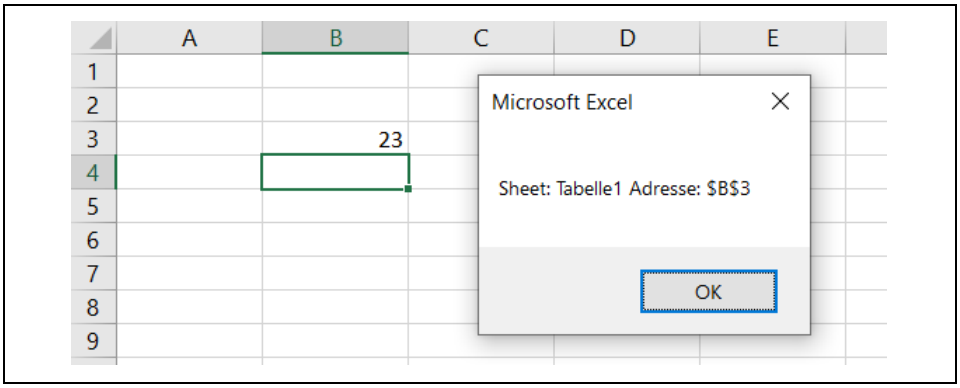

Bild 4. Änderungsmeldung

2 Die Application-Event-Klasse

Mitunter möchte man mehrere Application-Objekte verwenden und ihre Funktionalität auch ausweiten, also nicht nur eine Änderungsmeldung. Für gleiche Objekte erstellt man eine Konstruktionsvorschrift – die Klasse *clsAppEvent* (Bild 5).

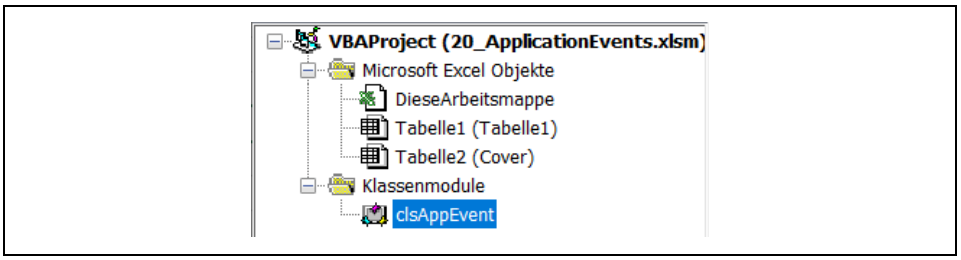

Bild 5. Das Klassenmodul clsAppEvent im VBA-Projekt

Nun bekommt die Klasse die Funktionalität, die zuvor im Code-Fenster des Workbooks stand. Die Instanziierung übernimmt der Konstruktor der Klasse, das *Class_Initialize*-Ereignis (Bild 6).

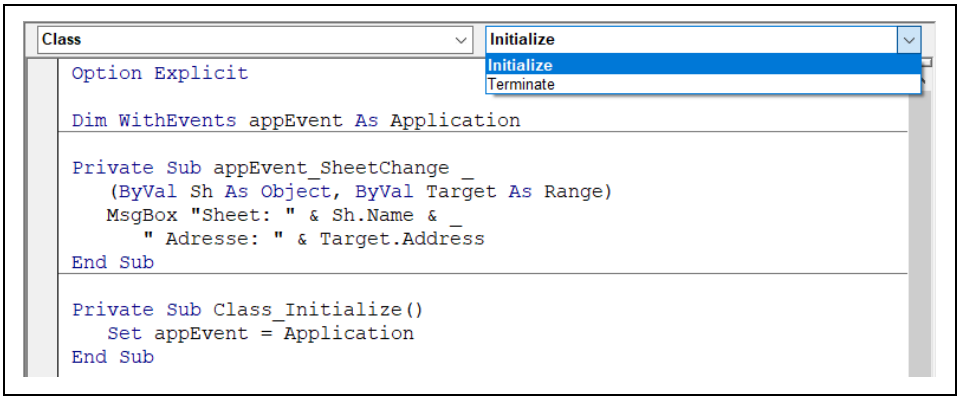

Bild 6. Die Klasse und ihre Ereignisse im Klassenmodul clsAppEvent

Genutzt werden die Klasse und ihre Objekte dann in irgendeinem Code-Fenster, wie nachfolgend in einem Code-Modul der Tabelle1 (Bild 7).

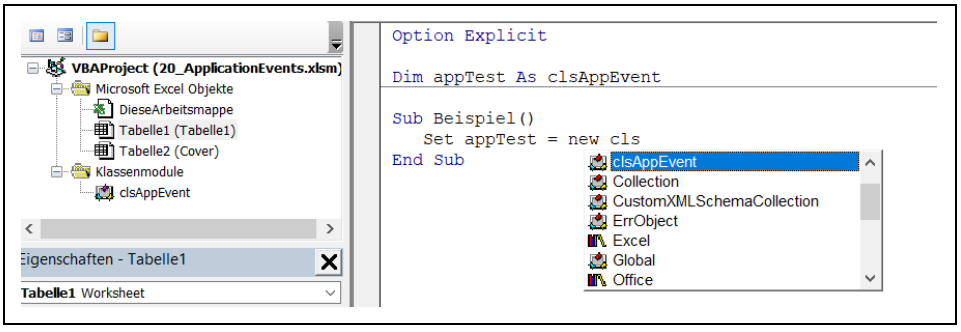

Bild 7. Beispielanwendung der Event-Klasse

Das Ergebnis ist das Gleiche und wieder wird eine Meldung ausgegeben. Der Unterschied ist, dass sich mit einer Klasse beliebig viele Application-Objekte instanziieren lassen. *Change-Event* ist jetzt eine Methode der Klasse. Die Meldung erscheint erst, wenn das Application-Objekt durch die Prozedur *Beispiel* instanziiert wurde (Bild 8). Natürlich auch mit jedem weiteren Objekt der Klasse.

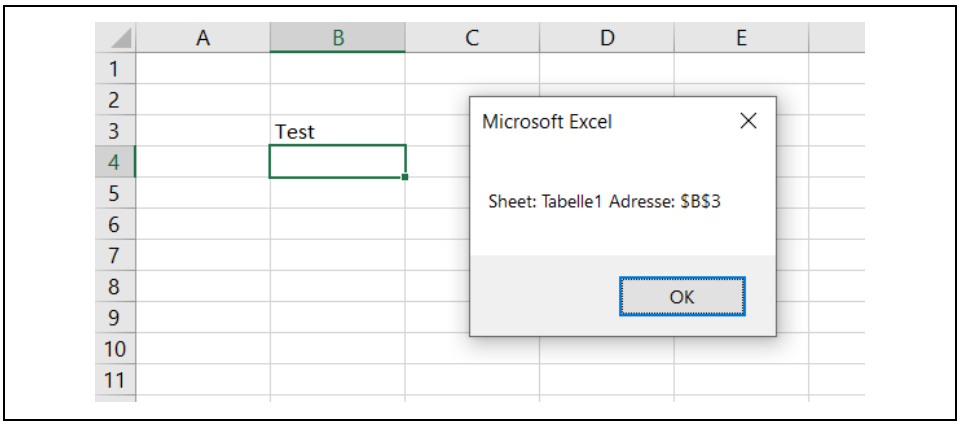

Bild 8. Änderungsmeldung