Programmieren in Anwendungen

Annette Bieniusa

Technische Universität Kaiserslautern

bieniusa@cs.uni-kl.de

24.05.2013

Überblick

Ereignisorientierte Programmierung

UserForm-Dialoge

Integrierte Office-Automatisierung

Sicherheitsaspekte

Ereignisorientierte Programmierung

Ereignisorientierte Programmierung

- *Ereignisse* treten z.B. beim Arbeiten mit Steuerelementen auf.
- Ereignisse können durch den Benutzer direkt (beispielsweise durch Anklicken von Buttons, Wechsel zwischen Dokumenten), aber auch durch das System selbst angestossen werden (z.B. Öffnen oder Speichern von Dokumenten).
- Ereignisprozeduren sind Makros, die als Reaktion auf bestimmte Ereignisse ausgeführt werden.

```
' Hebt bei Markieren einer Zelle die gesamte Zeile und
    Spalte hervor.
Sub Worksheet_SelectionChange(ByVal Target As Range)
    Cells.Interior.ColorIndex = xlColorIndexNone
    ActiveCell.EntireRow.Interior.ColorIndex = 15
    ActiveCell.EntireColumn.Interior.ColorIndex = 15
End Sub
```

Beispiel: Durchlauf von Arbeitsblättern

```
Sub Workbook_Open()
    Application.OnKey Key:="{PgUp}", Procedure:="SheetsUp"
    Application.OnKey Key:="{PgDn}", Procedure:="
        SheetsDown"
End Sub

Sub SheetsUp()
    Dim i As Integer
    i = ActiveSheet.Index + 1
    If i <= Sheets.Count Then Sheets(i).Select
End Sub

Sub SheetsDown()
    Dim i As Integer
    i = ActiveSheet.Index - 1
    If i >= 1 Then Sheets(i).Select
End Sub
```

Beispiel: Automatisierte Speicherabfrage

```
Sub Workbook_Open()
    Application.OnTime EarliestTime:=Now + TimeValue("
        00:10:00"), Procedure:="SaveWorkbook"
End Sub

Sub SaveWorkbook()
    If MsgBox("Save workbook?", vbYesNo) = vbYes Then
        ActiveWorkbook.Save
    Application.OnTime EarliestTime:=Now + TimeValue("
        00:10:00"), Procedure:="SaveWorkbook"
End Sub
```

UserForm-Dialoge

Dialoge und Formulare

Integrierte Dialoge für häufig verwendete Abfragen, z.B. Druck- oder Speicherdialog

Vordefinierte VBA-Dialoge für einfache Benutzerein- und ausgaben, z.B. `InputBox` und `MsgBox`

Benutzerdefinierte Dialoge (UserForm-Dialoge) für komplexere Interaktion

Dokumente als Formulare durch direktes Einbetten von Steuerelementen

Grundlagen zu UserForm-Dialogen

- Vielzahl von Steuerelementen, z.B. Listenfelder, Schaltflächen, Eingabefelder
- UserForm-Diloge werden durch Drag&Drop in der VBA-Entwicklungsansicht zusammengestellt.
- Flexible Reaktion auf Benutzereingaben durch Ereignisprozeduren, z.B. Hinzufügen von Werten

```
'Anzeigen eines Formulars
UserFormName.Show

'Beim Aktivieren eines Formulars
Sub UserFormName_Activate()
...
End Sub

'Ereignisprozeduren erstellen
Sub UserFormName_Ereignisname
Sub SteuerelementName_Ereignisname

'Ressourcenfreigabe
Unload UserFormName
```

Integrierte Office-Automatisierung

Kommunikation zwischen Office-Anwendungen

- Austausch von Daten kann auf verschiedenen Wegen erfolgen
  - Zwischenablage (bei bestimmten VBA-Objekten durch die Prozeduren `Copy` und `Paste`)
  - COM-Automation (Component Object Model) (Interprozesskommunikation unter Windows)
  - DAO/ADO (Data Access Objects/ActiveX Data Objects) (VBA-angepasste Objektmodelle zum Datenaustausch unter COM)
- Integration von Funktionalitä durch COM möglich

Allgemeine Schritte

1. Erstellen eines Verweises auf die Objektbibliothek der Office-Anwendung, die integriert werden soll (VBA-Entwicklungsumgebung, Extras→Verweise...)
2. Erzeugung des zu integrierenden Objekts
3. Programmierung der erwünschten Funktionalität
4. Freigabe des integrierten Objekts

Erzeugung des zu integrierenden Objekts

1. Deklaration der Objektvariablen

```
Dim appX As Objekttyp 'fruehes Binden

Dim appWord As Word.Application
Dim appExcel as Excel.Application
Dim sheetExcel as Excel.Sheet

Dim appX As Object 'spaetes Binden
```

2. Objektinstanzen erzeugen
   - Die `CreateObject`-Funktion startet die zugrunde liegende Anweisung und liefert einen Objektverweis auf eine neue Objektinstanz zurück.
   - Mit dem Schüsselwort `New` wird die zugrunde liegende Anweisung gestartet und wird eine neue Objektinstanz erzeugt (nur bei frühem Binden).
   - Die `GetObject`-Funktion liefert einen Objektverweis auf eine bereits gestartete Anwendung (häufig schneller).

Beispiele: Objekterzeugung mit Fehlerbehandlung

- Verwendung von `CreateObjekt`

```
On Error Resume Next
Set AppWord = CreateObject("Word.Application")
If Err = 429 Then
   MsgBox "Anwendung Word nicht installiert"
End If
```

- Verwendung von `GetObject`

```
On Error Resume Next
Set AppWord = GetObject(,"Word.Application")
If Err = 429 Then
   MsgBox "Anwendung Word noch nicht gestartet"
End If
```

- Optionale Pfadangaben bei `GetObject`

```
Set appExcel = GetObject(,"Excel.Application")
Set appExcel = GetObject("","Excel.Application") '
    startet Anwendung wie bei CreateObject
Set worksheetExcel = GetObject("C:\Daten\Statistic.
    xlsx")
```

Objekte freigeben

- Objekte schliessen bzw. beenden, danach stehen sie nicht mehr zur Verfügung

```
ObjektVariable.Close
ObjektVariable.Quit
```

- Arbeitsspeicher freigeben

```
Set ObjektVariable = Nothing
```

- Nicht freigegebener Arbeitsspeicher kann zu Programmabstürzen durch Speichermangel führen.

Beispiel: Daten aus Excel auslesen und in Word einfügen

```
Sub DatenAusExcel()
  On Error GoTo FehlerSub
  Dim app As Excel.Application
  Set app = GetObject(,"Excel.Application")
  app.Workbooks.Open ("Mappe.xlsx")
  If app.Visible = False Then
    app.Visible = True
  End If
  ActiveDocument.Bookmarks("WERT").Select
  Selection.InsertAfter app.ActiveSheet.Range("A1")
  app.Quit
  Set app = Nothing
  Exit Sub
FehlerSub:
  If Err = 429 Then
    Set app = CreateObject("Excel.Application")
  Else
    Err.Raise Err.Number
  End If
  Resume Next
End Sub
```

Beispiel: Einfügen einer Briefanrede

```
Dim appXL As Excel.Application

Private Sub UserForm_Activate()
  Dim Zaehler As Integer
  Set appXL = CreateObject("Excel.Application")
  appXL.Workbooks.Open "Namen.xlsx"
  appXL.Sheets("Mitarbeiter").Activate
  For Zaehler =1 To appXL.Range("A2").CurrentRegion.Rows.
      Count
    Me.lstName.AddItem appXL.Cells(Zaehler, 2)
  Next
    Me.lblAnzahl.Caption = Me.lstNamen.ListCount & " Namen
        geladen"
End Sub

Private Sub cmdAbbrechen_Click()
    appXL.Quit
    Set appXL = Nothing
    Unload frmBriefanrede
End Sub
```

Beispiel: Einfügen einer Briefanrede - Teil 2

```
Private Sub lstNamen_Change()
  Dim Zeile As Integer
  Dim Anrede As String

  Zeile = Me.lstNamen.ListIndex + 1
  If appXL.Cells(Zeile, 3) = 1 Then
  Else
      Anrede = "Liebe Frau " & appXL.Cells(Zeile, 1)
  End If
  Me.txtBriefanrede = Anrede
End Sub

Private Sub cmdEinfuegen_Click()
    Selection.TypeText Me.txtBriefanrede
    cmdAbbrechen_Click
End Sub
```

Sicherheitsaspekte

Sicherheitseinstellungen für Makros

- Ab Office 356: Authentifizierung von vertrauenswürdigen Quellen/Programmierern
- Standardmässig werden dann nur Makros ausgeführt, die aus vertrauenswürdigen Quellen stammen (Überprüfung durch digitale Signaturen).
- Die Freigabe beliebiger Makros ist nicht empfehlenswert, da hierbei grosse Sicherheitslücken geschaffen werden!