Programmieren in Anwendungen

Annette Bieniusa

Technische Universität Kaiserslautern

bieniusa@cs.uni-kl.de

21.05.2015

Überblick

Integrierte Office-Automatisierung

Outlook

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 Funktionalität 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, um Speicherplatz wieder freizugeben

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!

Gründe für Error 429

Aus der Microsoft Dokumentation
`http://support.microsoft.com/kb/828550`

- There is a mistake in the application.
- There is a mistake in the system configuration.
- There is a missing component.
- There is a damaged component.

Beispiel: Daten aus Excel auslesen und in Word einfügen

- Gegeben: Excel-Datei `''Mappe.xlsx''`, in der verschiedene Daten gespeichert sind
- In der Word-Datei, die das Makro enthält, soll an die Textmarke `WERT` der Eintrag aus Zelle `"A1"` eingetragen werden.

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

- Gegeben: Excel-Datei `''Namen.xlsx''` mit Datenblatt `''Mitarbeiter''`. In der ersten Spalte steht `"Frau"/"Mann"`, in der zweiten Spalte der Vorname und in der dritten Spalte der Nachname
- In der Wort-Datei gibt es eine UserForm namens `frmBriefanrede` mit folgenden Komponenten:
  - Label `lblBriefanrede` und `lblAnzahl`
  - Schaltflächen `cmdAbbrechen` und `cmdEinfuegen`
  - Listbox `lstNamen`

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
  'Iteration ueber die Zeilen
  For Zaehler = 1 To appXL.Range("A1").CurrentRegion.Rows.
      Count
    'Einfuegen der Nachnamen in Listbox
    Me.lstNamen.AddItem appXL.Cells(Zaehler, 3)
  Next
    Me.lblAnzahl.Caption = Me.lstNamen.ListCount & " Namen
        geladen"
End Sub

Private Sub cmdAbbrechen_Click()
    'Freigabe nicht benoetigter Resourcen
    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
  ' ListBox nummeriert ab 0!
  Zeile = Me.lstNamen.ListIndex + 1

  'Trim entfernt Leerzeichen am Anfand und Ende eines
      Strings
  If Trim(appXL.Cells(Zeile, 1).Text) = "Herr" Then
      Anrede = "Lieber Herr " & appXL.Cells(Zeile, 3)
  Else
      Anrede = "Liebe Frau " & appXL.Cells(Zeile, 3)
  End If
  Me.lblBriefanrede.Caption = Anrede
End Sub

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

Verwendung von Me

- Das Schlüsselwort `Me` kann innerhalb von Klassenmodulen verwendet werden, um auf die Klasseninstanz zu verweisen.
- Beispiel: Im ThisWorkbook-Modul, referenziert `Me` ThisWorkbook; in einem UserForm-Modul, bezieht es sich auf diese Form.
- Im vorigen Beispiel ist es einfacher `Me` zu werden als `frmBriefanrede`, da kürzer und einfacher anzupassen, falls das Klassenmodul umbenannt wird.

Beispiel: Excel-Diagramme in Word nutzen

- Word unterstützt Tabellen, allerdings bietet es keine Möglichkeit Diagramme daraus zu generieren.
- Ziel: Lese Werte aus einer Word-Tabelle und erstelle mittels Excel ein Diagramm, das in die Zwischenablage kopiert wird.
- Das Diagramm kann dann an gewünschter Stelle eingefügt werden.

Beispiel: Excel-Diagramme in Word nutzen

```
Sub DiagrammErstellen()
  'hier ohne Fehlerbehandlung!
  Dim appXL As New Excel.Application
  If Selection.Information(wdWithInTable) Then
    Selection.Tables(1).Select
    Selection.Copy          'Kopie in Zwischenablage
    With appXL
      .Workbooks.Add        'neue leere Mappe
      .ActiveWorkbook.Sheets(1).Paste 'Daten einfuegen
      .Charts.Add           'leeres Diagramm
      With .ActiveChart
        .ChartType = xl3DPie
        .SetSourceData appXL.Sheets("Tabelle1").UsedRange
        .Location xlLocationAsNewSheet 'neues Blatt
        .ChartArea.Copy  'Kopie in Zwischenablage
      End With
      .ActiveWorkbook.Close False  'Schliessen ohne
          Speichern
      .Quit
    End With
    Set appXL = Nothing
  Else
    MsgBox "Bitte Cursor in Tabelle setzen!"
  End If
End Sub
```

Beispiel: Outlook-Notizen aus Word

- Ziel: Erstelle aus markiertem Text in Word eine neue Notiz, die sogleich am Bildschirm angezeigt wird

Beispiel: Outlook-Notizen aus Word

```
Sub NotizErstellen()
  On Error GoTo Fehlerbehandlung
  Dim appOL As Outlook.Application
  Dim NotizOL As Outlook.NoteItem

  Set appOL = GetObject(,"Outlook.Application")
  Set NotizOL = appOL.CreteItem(olNoteItem)
  If Selection.Type = wdNoSelection Or Selection.Type =
      wdSelectionIP Then
    Selection.WholeStory 'gesamter Text
  End If
  NotizOL.Body = Selection.Range.Text
  NotizOL.Display 'Anzeige auf dem Bildschirm
Fehlerbehandlung:
  If Err = 429 Then
    Set appOL = CreateObject("Outlook.Application")
  End If
  Resume Next
End Sub
```

Das Outlook-Objektmodell

- Outlook bietet Adressbücher, Ordner mit Emails, Notizen, Aufgaben
- VBA-Code für Outlook kann nicht mit einzelnen Outlook-Elementen weitergegeben werden, sondern muss als *.bas-Datei exportiert und später importiert werden.
- Zugriff auf Outlook-Elemente immer über ein `NameSpace`-Objekt
- Windows unterstützt als einzige Datenquelle MAPI (Messaging Application Project Interface)
- Dies erlaubt z.B. das direkte Versenden von Emails

Objekte in Outlook

- NameSpace-Objekt

```
Set NamespaceVerweis =
    Application.GetNameSpace("MAPI")
```

- Eigenschaften: `AdressListe`, `Folders`,

`CurrentProfileName`,`ExchangeMailboxServerName`,...

- Ordner-Objete (z.B. Standard-Ordner)

```
Set OrdnerVerweis =
    NamespaceVerweis.GetDefaultFolder(Typkonstante)
```

- Neue Elemente erstellen

```
Set ObjektVerweis =
    Application.CreateItem(Typkonstante)
```

Objekte in Outlook

- NameSpace-Objekt

```
Set NamespaceVerweis =
    Application.GetNameSpace("MAPI")
```

- Eigenschaften: `AdressListe`, `Folders`,

`CurrentProfileName`,`ExchangeMailboxServerName`,...

- Ordner-Objete (z.B. Standard-Ordner)

```
Set OrdnerVerweis =
    NamespaceVerweis.GetDefaultFolder(Typkonstante)
```

- Neue Elemente erstellen

```
Set ObjektVerweis =
    Application.CreateItem(Typkonstante)
```

Objekttypen

Konstante	Beschreibung
`olMailItem`	EMail
`olNoteItem`	Notiz
`olContactItem`	Kontakt
`olTaskItem`	Aufgabe

Zugriff auf vorhandene Elemente

- Über die `Items`-Auflistung eines `Folder`-Objekts

Element	Beschreibung
`MailItem`	Email im Posteingangs-Ordner
`NoteItem`	Notiz im Notizen-Ordner
`Contact`	Kontakt im Kontakte-Ordner
`AppointmentItem`	Termin im Kalender-Ordner

- Weiter dann über objektspezifische Eigenschaften (z.B. `Recipients` oder `Body` bei `MailItem`)

Beispiel: Zuletzt erhaltene Nachricht

```
Sub LetzteNachricht()
  Dim MeinNS As NameSpace, Posteingang As Folder
  Dim Email As MailItem, Anzahl As Integer

  Set MeinNS = Application.GetNamespace("MAPI")
  Set Posteingang = MeinNS.GetDefaultFolder(olFolderInbox)
  Anzahl = Posteingang.Items.Count
  Set Email = Posteingang.Items(Anzahl)
  MsgBox "Die letzte Nachricht wurde empfangen " & Email.
      CreationTime
End Sub
```

Das Email-Objekt

- E-Mail erstellen

```
Set MailObjekt = Application.CreateItem(olMailItem)
```

- Empfänger bestimmen

```
MailObjekt.Recipients.Add "maier@mail.com"
```

- Betreff und Text der Nachricht

```
MailObjekt.Subject = "Wichtige Info"
MailObjekt.Body = "Lieber Herr Maier, ..."
```

- Anhang, Position bestimmt die Position des Symbols innerhalb der EMail (z.B. `Len(Text)+20`)

```
MailObjekt.Attachments.Add Dateipfad,Position
```

- Email versenden

```
MailObjekt.Send
```

Hinweis: Verwendung von ActiveX-Steuerelementen

- Weitere spezialisierte Softwarekomponenten für die Makroprogrammierungen
- Sie können in den Anwendungen selbst, aber auch UserForms verwendet werden
- Beispiele: Kalender, PDF-Reader, Media-Player, Smiley-Auswahl Menü, spezialisierte Diagrammtypen
- Auswahl der installierten ActiveX-Steuerelemente unter Entwicklertools $\rightarrow$ Steuerelemente / Einfügen $\rightarrow$ ActiveX-Steuerelemente
- Eventuell müssen die Elemente zuerst registriert werden

Zusammenfassung

- Informationsaustausch zwischen Anwendungen in VBA
- Life cycle eines Objekts
  - Deklaration
  - Instanzerzeugung (evtl. Fehlerbehandlung)
  - Verwenden des Objekts
  - Beenden/Schliessen + Freigeben
- Objektmodell in Outlook
  - Zugriff über `Namespace` - Objekt