

R | R | Z | N |

Regionales Rechenzentrum für Niedersachsen

11
102
1004

Leibniz
Universität
Hannover

Access 2010 – Programmierung Objekte und Klassenmodule

Objekte ...

- sind zum Beispiel Formulare, Berichte und Steuerelemente in Access.
- beschreiben eindeutig ein bestimmtes Element und deren Ausprägungen.
- kapseln Prozeduren (Methoden) und Variablen (Eigenschaften) in einem Container.
- belegen Platz im Arbeitsspeicher.
- basieren auf Klassen.
- werden in der Hilfe unter *Access 2010-Entwicklerreferenz – Access-Objektmodellreferenz* abgebildet.

Klassen ...

- definieren eine Struktur. In Access definiert zum Beispiel die Klasse „Formulare“ die Eigenschaften und Methoden eines Formulars.
- sind standardisierte Baupläne für Objekte.
- stellen Schablonen für Objekte bereit.
- sind Rezepte für die Erstellung eines bestimmten Objekts. In diesem Rezept werden allgemein die Eigenschaften und das Verhalten eines Objekts beschrieben.
- können die Basis für beliebig viele Objekt sein.

Die Klasse „Textbox“ ...

- hat die Eigenschaften ForeColor, Value, etc.
- hat die Methoden SetFocus, Requery etc, um die Eigenschaften zu verändern.
- kann auf die Ereignisse KeyPress, Undo etc reagieren.

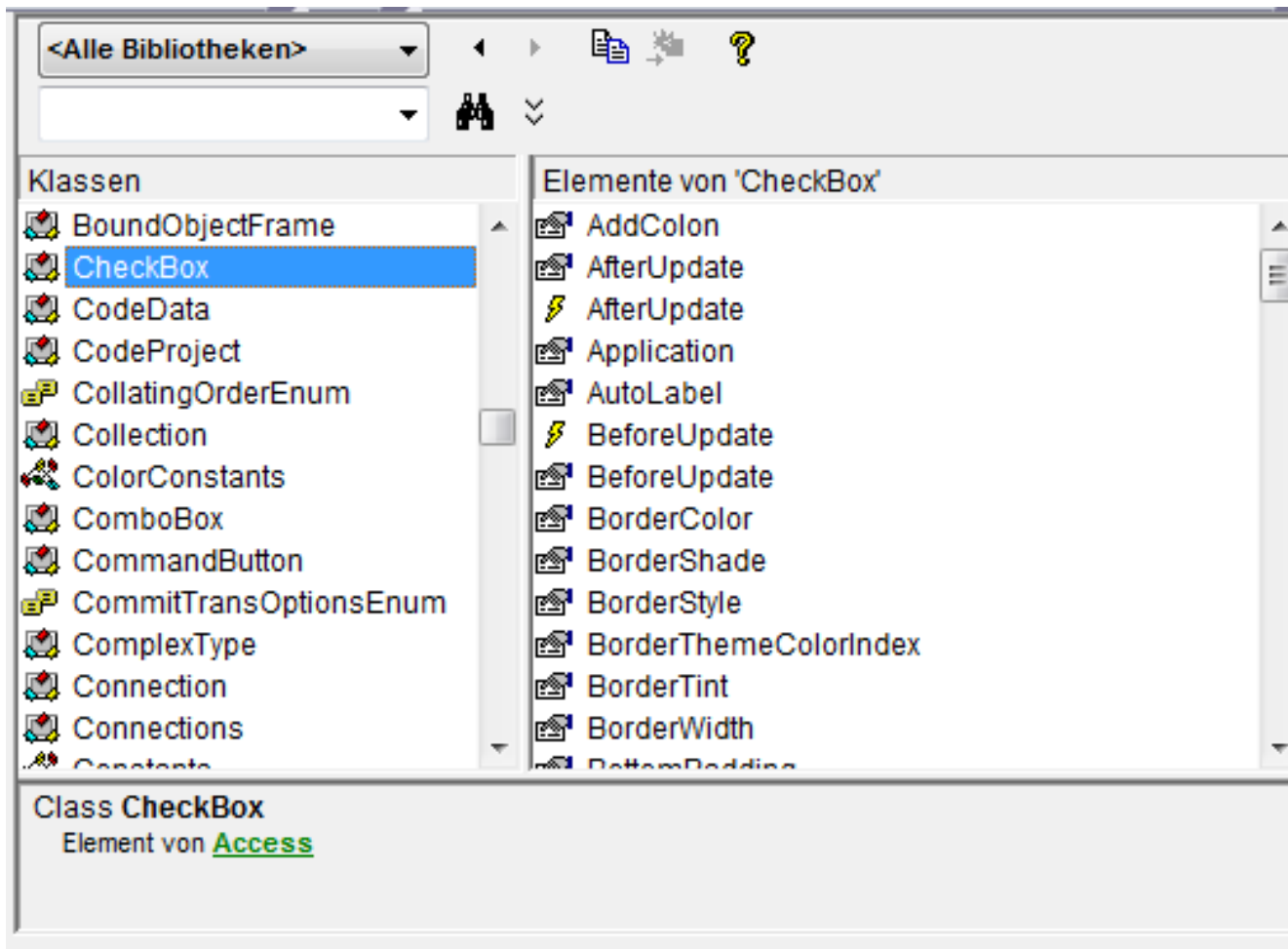
Das Objekt „Anzahl“ der Klasse „Textbox“ ...

- hat für folgende Eigenschaften die Ausprägung:
 - ForeColor = vbRed
 - Value = 15
- kann die Methoden der Klasse nutzen.
- kann auf die Ereignisse der Klasse reagieren.

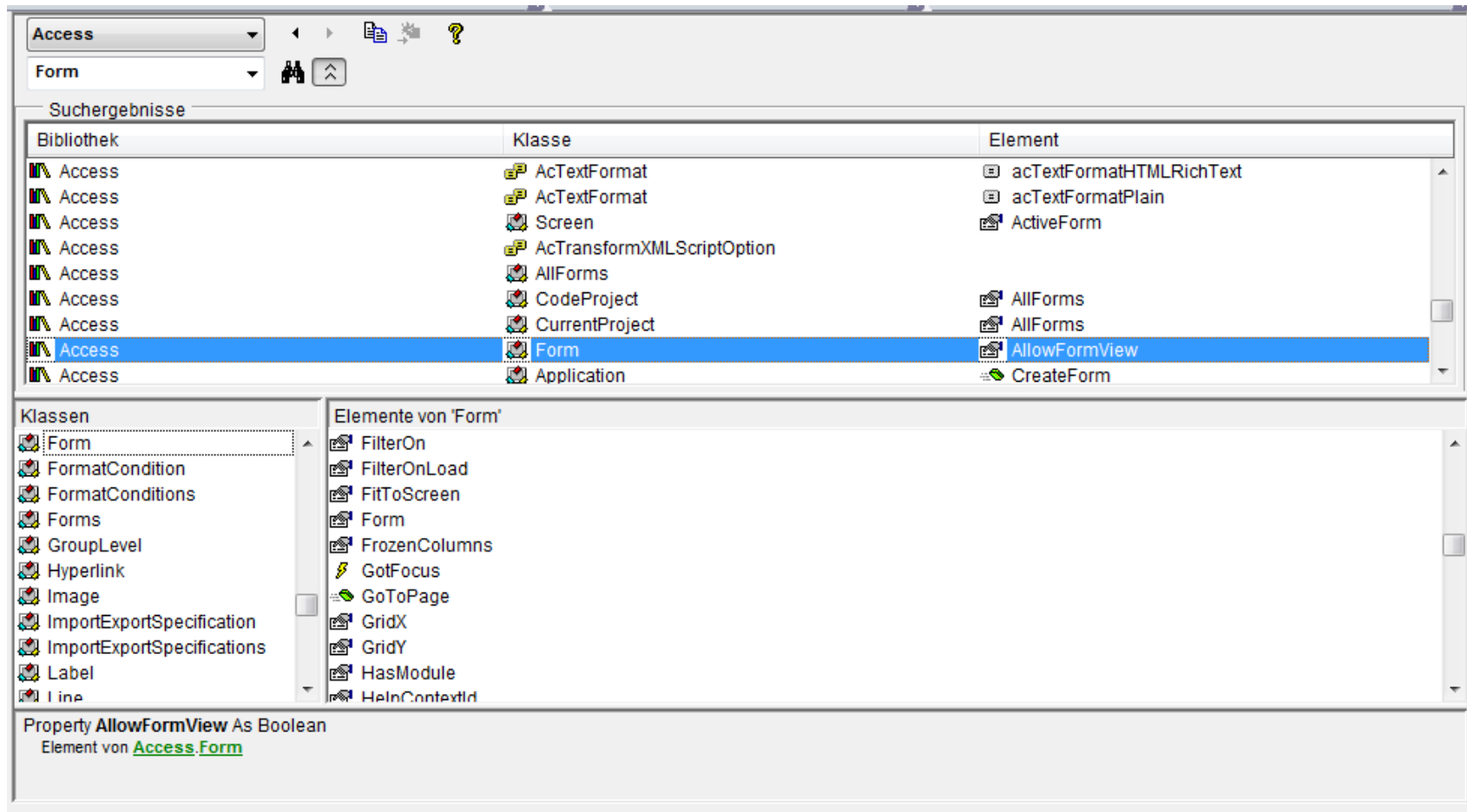
Objektkatalog ...

- ist eine Nachschlage-Hilfe für vordefinierte und in einem Projekt vorhandenen Objekte.
- ist ein Verzeichnis aller Objekte und deren Eigenschaften sowie Methoden.
- bildet die Hierarchie der Objekte untereinander ab.
- enthält Objekte, die mit Hilfe von Bibliotheken zusammengefasst werden. Für jede Office-Anwendung existiert eine eigene Bibliothek.
- wird mit Hilfe von *Ansicht – Objektkatalog* geöffnet.

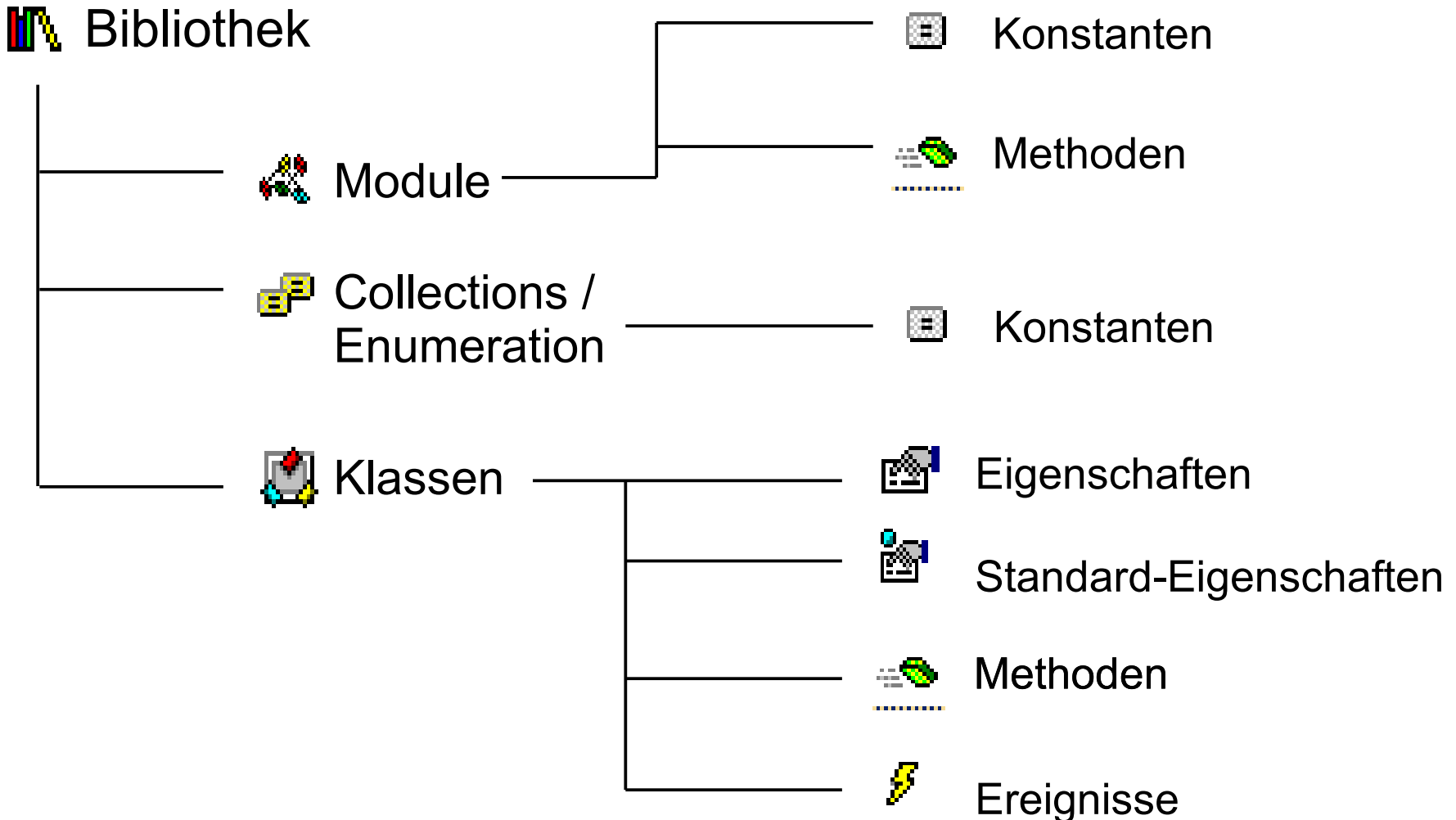
Objektkatalog (Nach dem ersten Öffnen)



Suche im Objektkatalog



Symbole der Elemente in dem Katalog

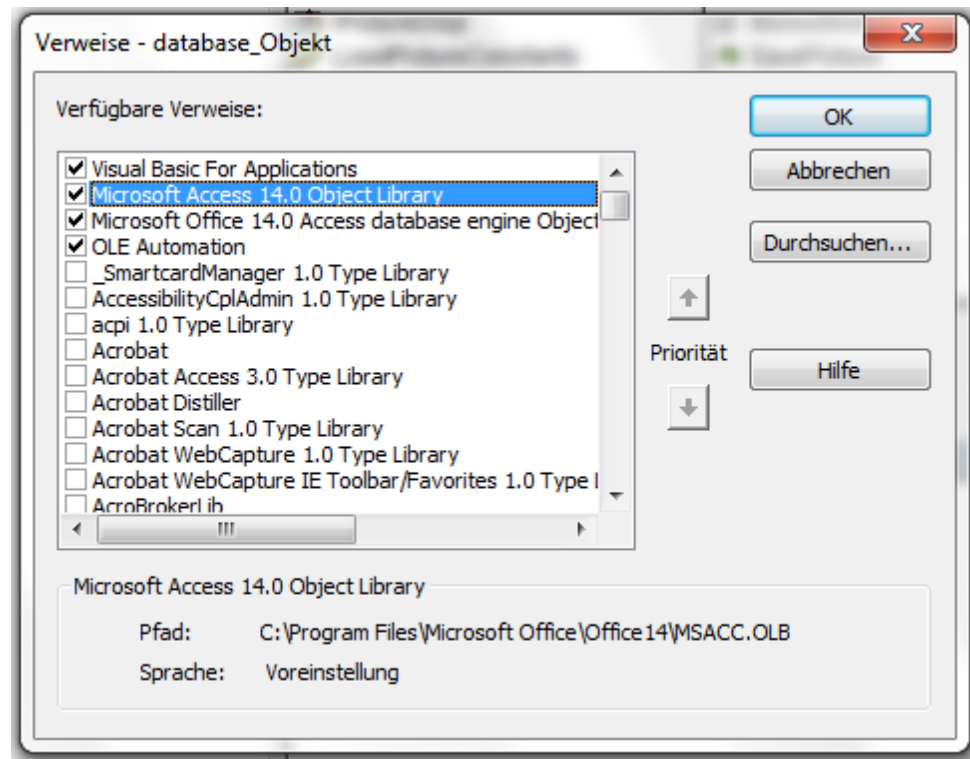


Standardmäßig eingebundene Bibliotheken

- Access beschreibt alle Objekte von Access wie zum Beispiel ein Formular etc.
- *DAO* (Data Access Object) enthält Objekte, die die Daten in einer Datenbank lesen oder schreiben können. Daten können mit Hilfe von SQL gefiltert werden. Mit Hilfe der Objekte ist ein Zugriff auf die Daten in einer Datenbank möglich.
- *database_Object* beschreibt alle, im aktuellen Projekt vorhandenen Objekte.
- *stdole* beschreibt das Einbetten und Verlinken von Objekten. Standard-OLE-Automation. Mit Hilfe der Bibliothek kann zum Beispiel ein Bild eingebunden werden.
- *VBA* beschreibt alle Objekte, die zu der Programmiersprache VBA gehören.

Verweise auf Bibliotheken ...

- werden mit Hilfe der Menüs *Extras* – *Verweise* im VBA-Editor angelegt.
- Mit Hilfe des Kontrollkästchens werden Verweise aktiviert (Häkchen) oder deaktiviert (leeres Feld).
- Der Speicherort des ausgewählten Verweises wird am unteren Rand angezeigt.
- haben meist die Dateiendung „.OLB“.



Objekte ...

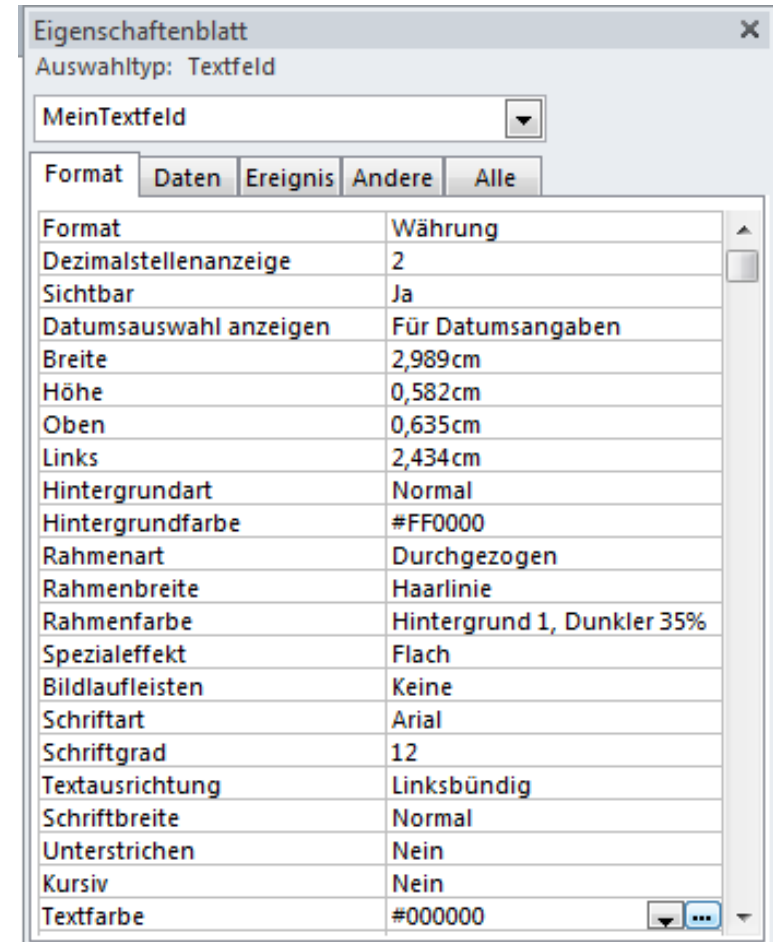
- werden mit Hilfe von Eigenschaften beschrieben.
- haben Methoden, um ihre Eigenschaften anzupassen.
- reagieren auf Ereignisse wie zum Beispiel ein Mausklick oder das Vergrößern eines Fensters.

Eigenschaften ...

- beschreiben ein Objekt eindeutig.
- werden für Formulare, Berichte und Steuerelemente in Access im Eigenschaftenblatt auf den Registerkarten Format, Daten und Andere angezeigt. Jedes der dort aufgeführten Eigenschaften ist auch in VBA vorhanden. VBA nutzt aber die englischsprachige Bezeichnung der Eigenschaften.
- können mit Hilfe von VBA in den meisten Fällen verändert werden.
- Objekte einer Klasse haben die gleichen Eigenschaften, aber unterschiedliche Ausprägungen für die Eigenschaften.

Beispiel

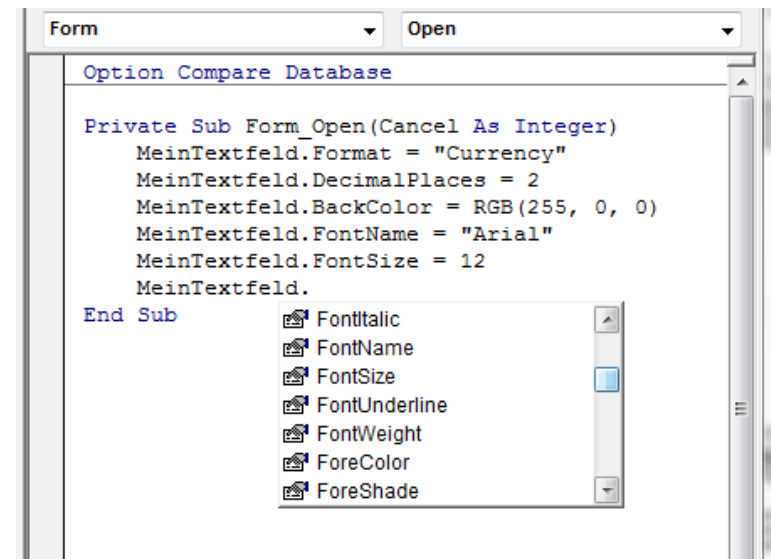
```
Private Sub Form_Open(Cancel As Integer)
    MeinTextfeld.Format = "Currency"
    MeinTextfeld.DecimalPlaces = 2
    MeinTextfeld.BackColor = RGB(255, 0, 0)
    MeinTextfeld.FontName = "Arial"
    MeinTextfeld.FontSize = 12
    MeinTextfeld.ForeColor = RGB(0, 0, 0)
End Sub
```



Syntax in VBA

« Objektname.Eigenschaft = Ausdruck »

- Der Objektname und die Eigenschaft werden durch ein Punkt getrennt. Nach dem Setzen des Punkts im Codebereich werden die vorhandenen Eigenschaften automatisch aufgelistet.
- Jede Eigenschaft hat einen bestimmten Datentyp. Zum Beispiel hat die Eigenschaft « .Format » den Datentyp String und « .FontSize » den Datentyp Integer.

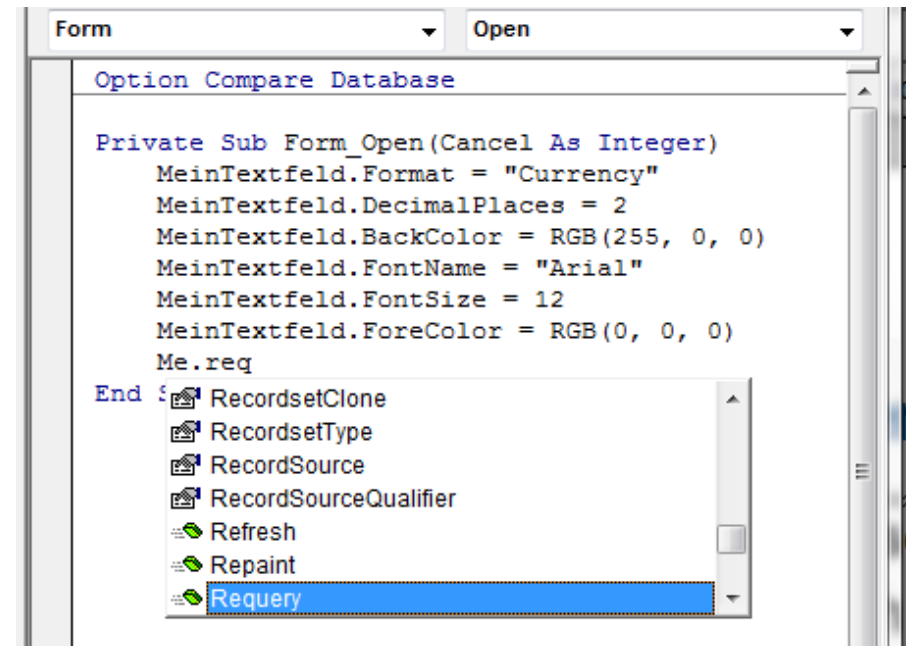


Methoden von Objekten ...

- können gekapselte Eigenschaften von Objekten verändern.
- lösen ein bestimmtes Verhalten aus.
- können von außen aufgerufen werden.
- sind für alle Objekte einer bestimmten Klasse gleich.
- sind mit Prozeduren vergleichbar.

Beispiel

```
Private Sub Form_Open(Cancel As Integer)
    Me.Requery
End Sub
```



Syntax in VBA

« Objektname.Methode »

- Der Objektname und die Methode werden durch ein Punkt getrennt. Nach dem Setzen des Punktes im Codebereich werden die vorhandenen Methoden automatisch aufgelistet.
- Methoden können Argumente haben. Häufig sind die Argumente einer Methode optional.
- Methoden können Rückgabewerte besitzen.

Zusammenfassung von Anweisungen für ein Objekt

```
Private Sub Form_Open(Cancel As Integer)
```

```
    With Me.Textfeld
```

```
        .Format = "Currency"
```

```
        .DecimalPlaces = 2
```

```
        .BackColor = RGB(255, 0, 0)
```

```
        .FontName = "Arial"
```

```
        .FontSize = 12
```

```
        .ForeColor = RGB(0, 0, 0)
```

```
    End With
```

```
    Me.Refresh
```

```
End Sub
```

With-Anweisung ...

- beginnt mit dem Schlüsselwort « With ».
- bezieht sich auf ein bestimmtes Objekt « With MeinTextfeld ».
- enden mit « End With ».
- Alle Anweisungen zwischen « With » und « End With » beziehen sich auf das Objekt „MeinTextfeld“. Es werden Anweisungen für ein bestimmtes Objekt zusammengefasst.

Hinweise

- « .Format = "Currency" »
 - Vor Eigenschaften und Methoden, die sich auf das angegebene Objekt (hier: „MeinTextfeld“) beziehen, muss immer das Trennzeichen „Punkt“ gesetzt werden.
 - Der Name des dazugehörigen Objekts ist im Kopf der With-Anweisung angegeben.
- « MeinWaehrung.Format = .Format »
 - Eigenschaften und Methoden von anderen Objekten können genutzt werden.
 - Es muss aber der Objektname angegeben werden (siehe « MeinWaehrung.Format »).

Ereignisse ...

- ist eine Aktion, die in Verbindung mit einem Objekt steht.
- werden durch die Tastatur, der Maus oder einem Zeitgeber ausgelöst.
- werden durch Veränderungen an einem Objekt ausgelöst.
- werden auf der Registerkarte Ereignis im Eigenschaftenblatt eines Objekts aufgelistet.
- Auf Ereignisse kann mit Hilfe sogenannter Ereignisprozeduren reagiert.

Ereignisprozeduren ...

- fassen Anweisungen zusammen, mit denen auf ein bestimmtes Ereignis reagiert wird.
- enthalten Anweisungen, die das Objekt verändern.
- können manchmal vorzeitig gestoppt werden.
- beginnen immer mit « Sub » und enden mit « End Sub ».

Prozedur-Kopf

« Private Sub txtOperandLinks_KeyPress(KeyAscii As Integer) »

« Private Sub Form_Load() »

- Das Schlüsselwort « Sub » kennzeichnet eine Prozedur. Jede Prozedur endet mit « End Sub»
- Der Name einer Ereignisprozedur hat folgenden Aufbau: Objekt_Ereignis.
- In den runden Klammern werden der Prozedur Argumente übergeben.
- Jede Ereignisprozedur kann nur von dem Objekt angestoßen werden. Die Prozedur ist nicht öffentlich (« Private »).

Name einer Ereignisprozedur

« Private Sub txtOperandLinks_KeyPress(KeyAscii As Integer) »

« Private Sub Form_Load() »

- Der Name beginnt mit dem Auslöser der Prozedur. Die erste Prozedur kann nur von dem Objekt txtOperandLinks und die zweite Prozedur vom Formular selber ausgelöst werden. Das erste Wort definiert den Eigentümer der Prozedur.
- Dem Namen des Auslösers folgt ein Unterstrich.
- Anschließend wird das ausgelöste Ereignis beschrieben. In diesem Beispiel ist das Ereignis KeyPress (Taste gedrückt) und Load (Beim Laden). Die Anzahl der Argumente sowie deren Typ sind abhängig vom ausgelösten Ereignis.

Argumentliste ...

- wird durch die runden Klammern begrenzt.
- kann beliebig viele Argumente enthalten. Die Argumente werden durch Kommata getrennt.
- ist eine Auflistung von Werten, die der Prozedur übergeben werden.

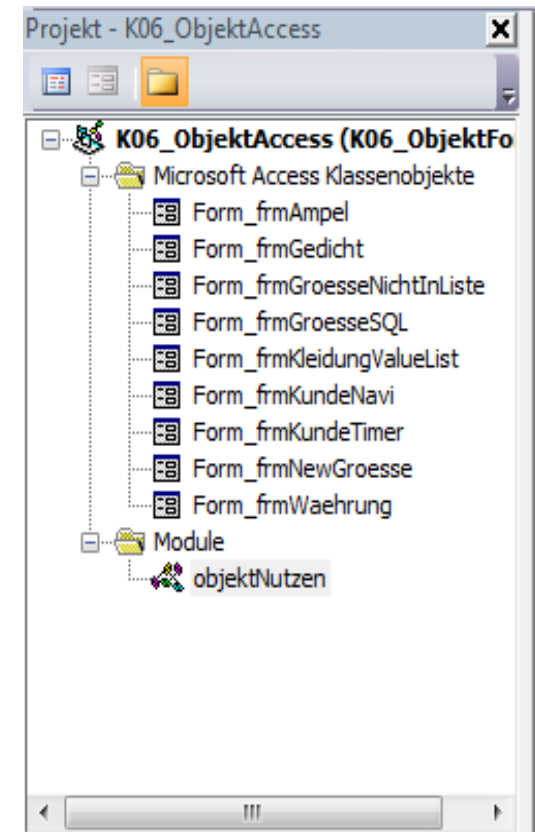
Ereignis vorzeitig stoppen

« Private Sub Report_Unload(Cancel As Integer) »

- Mit Hilfe des Arguments « Cancel » kann ein Ereignis und die Reaktion darauf vorzeitig abgebrochen werden.
- Die Anweisung « Cancel = True » bricht die Ausführung vorzeitig ab.

Ereignisprozeduren im VBA-Editor ...

- werden im Ordner Microsoft Access Klassenobjekte gespeichert.
- Für jedes Bericht oder Formular, das auf ein Ereignis reagiert, werden Klassenmodule angelegt.
- sind immer an ein bestimmtes Objekt gebunden. Das daran gebundene Objekt, spiegelt sich im Namen des Klassenmoduls wieder.
- enthalten Klassenmodule, die an ein Formular oder Bericht gebunden sind.



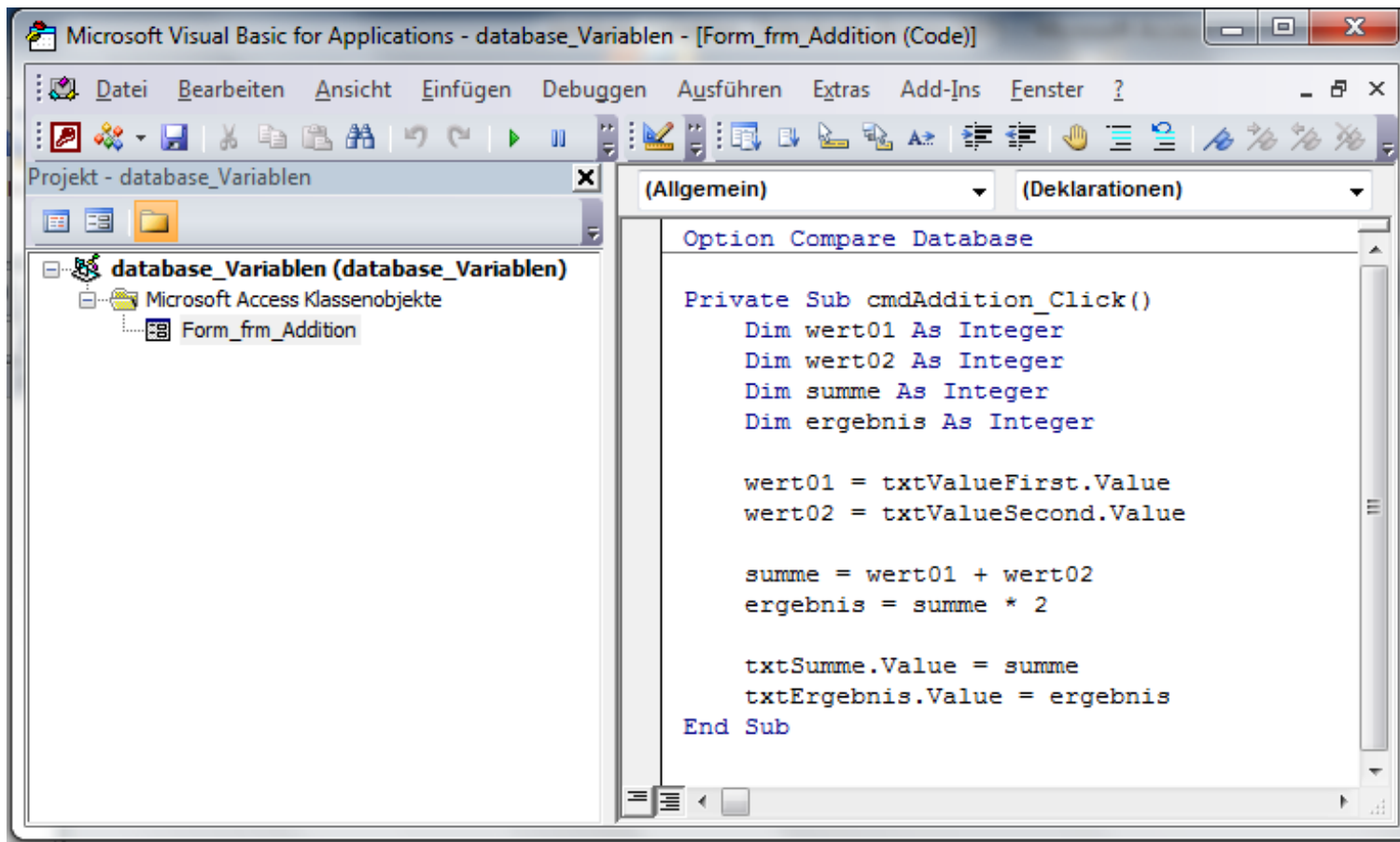
Klassenmodule ...

- sind Container für Code, die das Verhalten des Formulars oder Berichts festlegen.
- enthalten Code, der auf Benutzeraktionen reagieren kann.
- können Code enthalten, der das Formular oder Bericht allgemein betrifft.
- beginnen mit dem Präfix Form oder Report. Dem Präfix folgt der Name des dazugehörigen Formulars (Form) oder Berichts (Report).
- haben die Dateiendung „.cls“.

... anlegen und entfernen

- Klassenmodule, die an ein Formular oder Bericht gebunden sind, werden automatisch angelegt und entfernt.
- Durch die Anlage einer Ereignisprozedur auf der Registerkarte *Ereignis* im Eigenschaftenblatt des Objekts (Formular, Bericht, Steuerelement) wird ein Klassenmodule automatisch erstellt oder, falls vorhanden, aufgerufen.
- Durch die Löschung des dazugehörigen Formulars oder Berichts wird automatisch das Modul gelöscht.

Beispiel



Microsoft Visual Basic for Applications - database_Variablen - [Form_frm_Addition (Code)]

Projekt - database_Variablen

database_Variablen (database_Variablen)

- Microsoft Access Klassenobjekte
 - Form_frm_Addition

(Allgemein) (Deklarationen)

```
Option Compare Database

Private Sub cmdAddition_Click()
    Dim wert01 As Integer
    Dim wert02 As Integer
    Dim summe As Integer
    Dim ergebnis As Integer

    wert01 = txtValueFirst.Value
    wert02 = txtValueSecond.Value

    summe = wert01 + wert02
    ergebnis = summe * 2

    txtSumme.Value = summe
    txtErgebnis.Value = ergebnis
End Sub
```

... importieren und exportieren

- Der VBA-Editor ist geöffnet.
- *Datei – Datei importieren*. Ein gespeichertes Klassenmodul wird in das aktuelle Projekt geladen.
- *Datei – Datei exportieren* speichert eine Kopie des aktuellen Klassenmoduls.