FAQ *ACCESS-Trainer-Edition:*

Formular schließen

```
DoCmd.Close acForm, "Filme", acSavePrompt
```

`DoCmd.Close` ist die Anweisung: den Befehl *Schließen* auszuführen.
Die *ACCESS*-Konstante `acForm` definiert, dass es sich um ein *Formular* handelt.
Der *Name* des *Formulars* ist: *"Filme"*
Am Ende der *Syntax* wird festgelegt, ob das *Formular:*

- nach Rückfrage geschlossen wird (`acSavePrompt`),
- vor dem Schließen gespeichert wird (`acSaveYes`),
- nicht gespeichert wird (`acSaveNo`).

Funktion: *DatAdd()*

= DatAdd("t";+4;[txtvom])

Die *Syntax* der Funktion *DatAdd()* gliedert sich wie folgt:

Name der Funktion: *DatAdd()*
in Klammern steht zunächst das *Intervall "t"* für *Tage,*
danach die *Anzahl* der *Tage* die *addiert* (+) oder *subtrahiert* (-) werden sollen, und am Ende das *Ausgangsdatum*.

Weitere *Intervalle* finden Sie in unserer Hilfedatei: *Datums- und Zeitfunktionen*

Funktion: *DatTeil()*

= DatTeil("ww";[txtvom])

Die *Syntax* der Funktion *DatTeil()* gliedert sich wie folgt:

Name der Funktion: *DatTeil(),*
in Klammern steht zunächst das *Intervall "ww"* für *Wochen,*
und am Ende das *Ausgangsdatum*.

Weitere *Intervalle* finden Sie in unserer Hilfedatei: *Datums- und Zeitfunktionen*

On Error Resume Next

Mit der *Fehlerroutine:*

```
On Error Resume Next
```

wird erreicht, dass eine *Prozedur* beim Auftreten eines Fehlers nicht mit einer Meldung unterbrochen, sondern mit dem nächsten gültigen Eintrag fortgesetzt wird.

Cancel = True

Damit nach einer Fehlermeldung der *Cursor* solange in einem *Feld* positioniert bleibt bis ein korrekter Wert eingegeben wird, muss in die *Prozedur* die Codezeile:

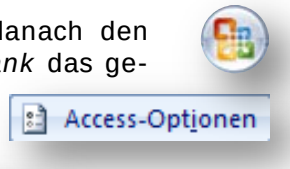
```
Cancel = True
```

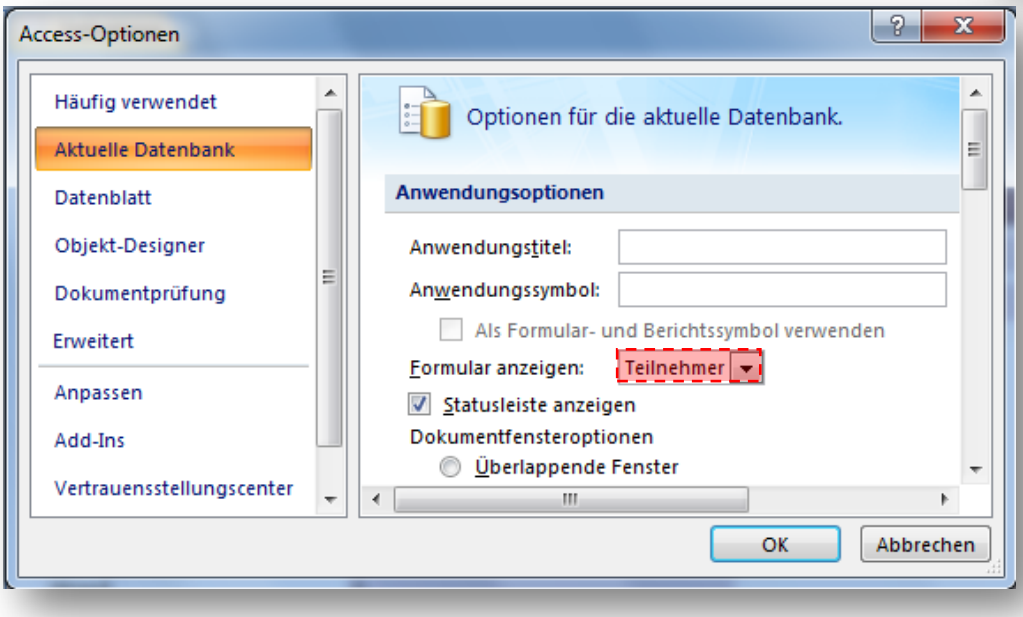
eingebunden werden.

Startformular einrichten

Wenn Sie mit einer *ACCESS*-Version ***bis*** 2003 arbeiten können Sie das *Formular,* das beim Öffnen der *Datenbank* angezeigt werden soll, im *Dialogfenster* des Menüs *Extras – Start* aktivieren.

In *ACCESS 2007* klicken Sie zunächst die *Office*-Schaltfläche, und danach den Button *ACCESS*-Optionen an und aktivieren im Menü *Aktuelle Datenbank* das gewünschte *Formular*.





Vollbildmodus

Fügen Sie im *Visual Basic-Editor*, Formular-Ereignis `Form_Load()`, die Codezeile:

```
DoCmd.Maximize
```

ein wenn Sie erreichen wollen, dass das *Formular* im *Vollbildmodus* angezeigt wird.

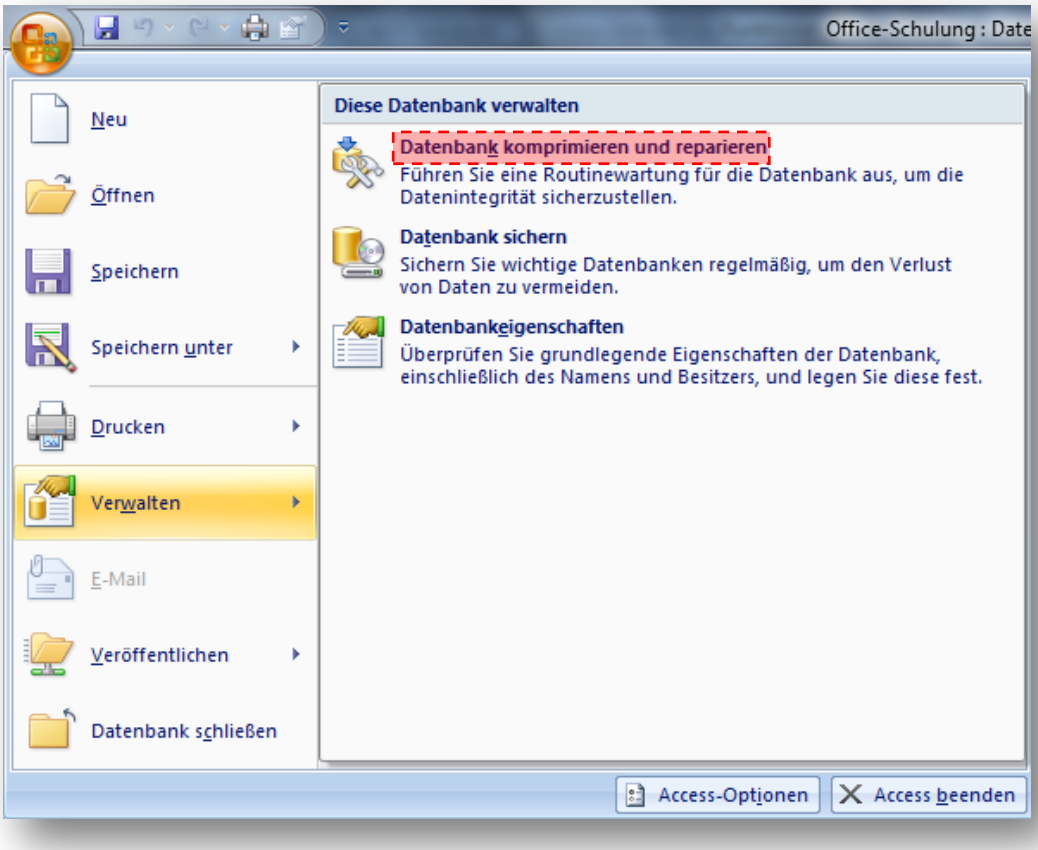
Felddatentyp: AutoWert

Mit Hilfe des Felddatentyps *AutoWert* wird von *ACCESS* eine fortlaufende Nummerierung generiert.

Beachten Sie jedoch, dass in der Auflistung eine Lücke entsteht wenn ein *Datensatz* gelöscht wird. Fehlende Nummern werden nicht automatisch ergänzt.

In den *ACCESS-Versionen* ***bis*** 2003 kann eine neue Reihenfolge durch Anklicken des Menüs *Extras – Datenbank-Dienstprogramme – Datenbank komprimieren und reparieren* generiert werden.

In *ACCESS 2007* finden Sie dieses Dienstprogramm nach Anklicken der *Office-Schaltfläche* im Menü *Verwalten – Datenbank komprimieren und reparieren*.



Um fortlaufende Rechnungs- oder Auftragsnummern zu erzeugen, sollten Sie deshalb andere Lösungen vorziehen, und die fortlaufende Nummerierung mit Hilfe einer entsprechenden *Function* oder mittels Hilfsfeldern erstellen.

Domänenfunktion: *DomWert()*

Mit dieser *Domänenfunktion* können Werte aus einer *Tabelle* oder *Abfrage* in den *Feldern* eines *Formulars* oder *Berichts* ausgegeben werden.

```
=DomWert("[Artikelbez]";"Artikel";"[Art-Nr]=[txtArt-Nr1]")
```

Die *Syntax* aller *Domänenfunktionen* ist gleich und gliedert sich wie folgt:

Name der *Domänenfunktion*: *DomWert()*
in Klammern steht zunächst der ***Ausdruck***, also der Name des *Feldes* in der *Tabelle* oder *Abfrage*, dessen Wert angezeigt werden soll: *"[Artikelbez]"*,
danach die ***Domäne***, also der Name der *Tabelle* oder *Abfrage*, die das vorstehende *Feld* enthält: *"Artikel"*
und danach die ***Kriterien***: *"[Art-Nr]=[txtArt-Nr1]"*
wobei im Beispiel nur dann eine Anzeige erfolgt, wenn der Feldinhalt des **Tabellen**feldes *[Art-Nr]* identisch ist mit dem Feldinhalt des **Formular**feldes *[txtArt-Nr1]*.

Methode: *Requery*

Requery ist eine *Methode* - keine *Eigenschaft* – mit welcher komplette *Formulare* oder berechnende *Felder* aktualisiert werden können.

Der Inhalt eines *Textfeldes* wird beispielsweise mit folgender *Syntax* aktualisiert:

```
Me.[txtArtBez1].Requery
```

Mit dieser Codezeile wird im aktuellen Formular (`Me`) das *Textfeld* `[txtArtBez1]` aktualisiert.

Funktion: *NZ()*

Wenn in einem *Textfeld* die Inhalte verschiedener *Felder* addiert werden sollen, und einzelne *Felder* dabei leer ("") sein können oder den Wert 0 (Null) enthalten, kann eine Addition mit Hilfe der Funktion *NZ()* erfolgen.

In den *Eigenschaften* des Summenfeldes, Zeile *Steuerelementinhalt,* muss dazu beispielsweise die folgende *Syntax* eingetragen werden:

```
=NZ([txtPreisArt1])+NZ([txtPreisArt2])+NZ([txtPreisArt3])+NZ([txtPreisArt4])
```

Ergebnis aus *Textfeld* in *Tabelle* übertragen

Mit der nachfolgenden Codezeile kann das Ergebnis eines berechnenden *Feldes* vom *Formular* in dessen *Tabelle* übertragen werden:

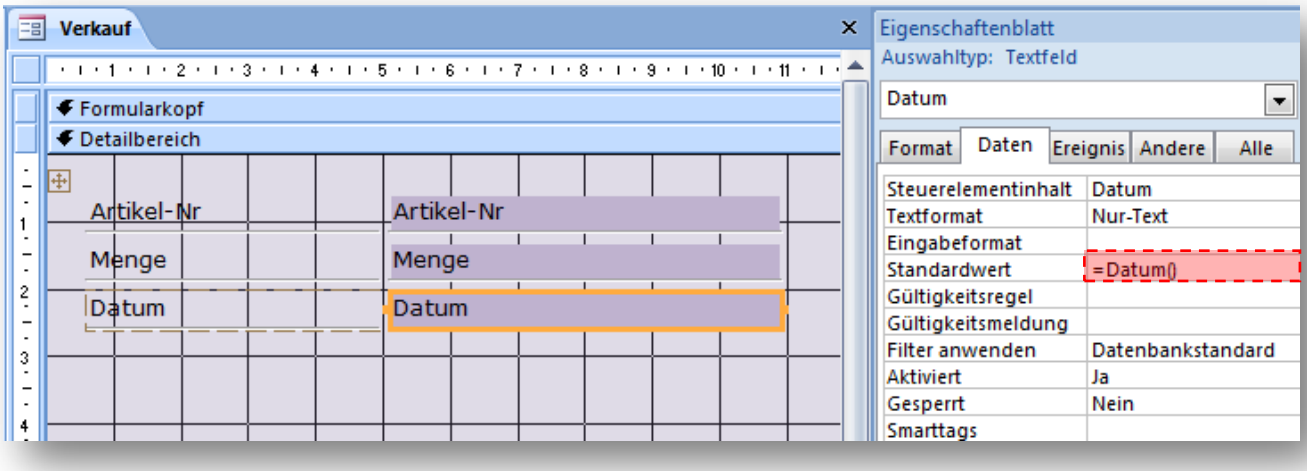
```
Me!Gesamt = Me![txtGesamt]
```

Ziel ist in der *Tabelle* des aktuellen *Formulars* (Me) das Feld *Gesamt*
Quelle ist im aktuellen *Formular* (Me) das Textfeld *[txtGesamt]*

Aktuelles Datum als *Standardwert* in *Textfeld* ausgeben

Wenn Sie das aktuelle Datum als *Standardwert* in einem *Textfeld* ausgeben wollen schreiben Sie in dessen *Eigenschaften*, Zeile *Standardwert*:

```
=Datum()
```



Zeilenschaltung in *Textfeld* einfügen

Wenn Sie in einem *Textfeld* eine komplette Adresse ausgeben wollen, die auch korrekt untereinander angeordnet ist, müssen Sie die einzelnen Feldinhalte nicht nur verketten, sondern am Zeilenende auch einen 'Zeilenumbruch' einfügen. Die entsprechende *Formel* kann in den *Eigenschaften* des *Textfeldes*, Zeile *Steuerelementinhalt* eingegeben werden:

```
=[txtVorname] &" "&[txtName] & Zchn(13) & Zchn(10) & [txtStraße] & Zchn(13) & Zchn(10) & [txtPLZ] & " " & [txtWohnort]
```

Die einzelnen *Textfelder* (*[txtVorname], [txtName]* usw.) verketten Sie, indem Sie dazwischen zwei *Verkettungsoperatoren* (*&*) und einen *Leerschritt* (*" "*) einfügen. Damit die *Zeilen* auch untereinander angeordnet werden, muss jeweils am Zeilenende außerdem eine 'Zeilenschaltung' und ein 'Wagenrücklauf' (ein Relikt aus Schreibmaschinenzeiten) *Zchn(13) & Zchn(10)* eingefügt werden. Alternativ können Sie auch *chr(13) & chr(10)* eintragen.

Beachten Sie unbedingt die Reihenfolge der Einträge, damit die Darstellung auch korrekt erfolgt.

Zeilenschaltung in *MsgBox* einfügen

Sie können 'Zeilenschaltungen' auch innerhalb einer *MsgBox* einfügen. Dazu muss an den entsprechenden Positionen im Programmcode (*Visual Basic-Editor*) die *VB-Konstante* `vbCrLf` eingefügt, und mit dem restlichen Text der Meldung verknüpft werden:


```
MsgBox "kein vorzeitiger Austritt - " & vbCrLf & "deshalb wird das Austrittsdatum wieder gelöscht!", vbInformation + vbOKOnly..."
```

Mit den VB-Konstanten `vbInformation + vbOKOnly` wird in die *MsgBox* außerdem ein *i-Symbol* und eine *OK-Schaltfläche* eingefügt.

Funktion: *DatDiff()* und *DateDiff()*

In *ACCESS* können Sie die Anzahl der Tage zwischen zwei Datumswerten mit Hilfe der Funktion *DatDiff()* ermitteln.

```
=DatDiff("t";[txtvom];[txtbis])
```

Die *Syntax* der Funktion *DatDiff()* gliedert sich wie folgt:

Name der Funktion: *DatDiff()*
in Klammern steht zunächst das *Intervall "t"* für *Tage*
und danach das *Anfangsdatum* und das *Enddatum*.

Wenn Sie die *Funktion* innerhalb einer *Prozedur* im *Visual Basic-Editor* einbinden wollen lautet der Funktionsname *DateDiff()*.

Weitere *Intervalle* finden Sie in unserer Hilfedatei: *Datums- und Zeitfunktionen*

Steuerelemente einblenden

Wenn Sie in einem *Formular Steuerelemente* ein- oder ausblenden wollen, muss in der entsprechenden *Prozedur* im *Visual Basic-Editor* eine der folgenden Codezeilen eingebunden werden:

```
Me.[Steuerelementname].Visible = True
```

Mit dieser Codezeile wird im aktuellen Formular (`Me`), *Steuerelement* [`Steuerelementname`] die Eigenschaft `Visible` (sichtbar) auf `True` (Wahr) gesetzt und das *Steuerelement* eingeblendet.

```
Me.[Steuerelementname].Visible = False
```

Mit dieser Codezeile wird im aktuellen Formular (`Me`), *Steuerelement* [`Steuerelementname`] die Eigenschaft `Visible` (sichtbar) auf `False` (Falsch) gesetzt und das *Steuerelement* ausgeblendet.

Schaltflächen-Beschriftung festlegen

Wenn Sie in einem *Formular* die Beschriftung einer *Befehlsschaltfläche* mittels Programmcode verändern wollen, muss in der entsprechenden *Prozedur* im *Visual Basic-Editor* die folgende Codezeile eingebunden werden:

```
Me.[Buttonname].Caption = "Beschriftung"
```

Mit dieser Codezeile wird im aktuellen Formular (`Me`), der *Befehlsschaltfläche* `[Buttonname]`, Eigenschaft `Caption` (Beschriftung) ein neuer Wert zugewiesen: `="Beschriftung".`

Datensatznavigation

Wenn Sie in einem *Formular* durch Anklicken einer *Schaltfläche* zum letzten *Datensatz* 'springen' wollen, müssen Sie in der entsprechenden *Prozedur* im *Visual Basic-Editor* die folgende Codezeile einbinden:

```
DoCmd.GoToRecord , , acLast
```

`DoCmd.GoToRecord` ist die Anweisung zu einem *Datensatz* zu 'springen', ergänzt durch eine *ACCESS*-Konstante `acLast` - also zum letzten *Datensatz*.

Weitere *ACCESS*-Konstanten sind:

- `acFirst` – zum ersten Datensatz
- `acPrevious` – zum vorherigen Datensatz
- `acNext` – zum nächsten Datensatz

Exit Sub

Wollen Sie erreichen, dass nach Anzeigen einer *MsgBox* die weitere Ausführung einer *Prozedur* abgebrochen, und kein weiterer Programmcode mehr abgearbeitet wird, muss die folgende Codezeile in die *Prozedur* im *Visual Basic-Editor* eingebunden werden:

```
Exit Sub
```

Form Current-Ereignis

Wenn Sie in einem *Formular* beim Anlegen von *Datensätzen* einzelne *Objekte* ein– oder ausgeblendet haben, müssen diese *Eigenschaften* auch beim *Durchblättern* der *Datensätze* festgelegt sein.

Das heißt, die entsprechenden Anweisungen sind z.B. nicht nur dem *Klick-Ereignis* einer *Schaltfläche* oder dem `Form_Load`-Ereignis zuzuweisen, sondern auch dem Ereignis: `Form_Current`.

Formatierung eines Betrages mittels Programmcode

Wenn Sie in einer *MsgBox* einen Wert z.B. im *Währungsformat* ausgeben wollen, muss die Funktion *Format()* in die *Prozedur* im *Visual Basic-Editor* eingebunden, und die Codezeile wie folgt gestaltet werden:

```
MsgBox"..."&Format(Me.[txtProvEinzelbetrag],"#,##0.00 EUR")
```

Mit dieser Codezeile wird die Meldung einer *MsgBox* mit dem Betrag im aktuellen Formular (`Me`), *Textfeld* `[txtProvEinzelbetrag]` verkettet (*&*) und zusätzlich in die Funktion *Format()* eingebunden, damit der Feldinhalt auch im Währungsformat `"#,##0.00 EUR"` ausgegeben wird.

Beachten Sie, dass im Programmcode im *Visual Basic-Editor* **Dezimalstellen** immer mit einem **Punkt** getrennt werden müssen, und das Tausender-Trennzeichen ein **Komma** ist!

Funktion: *DatSeriell()* und *DateSerial()*

Mit diesen Funktionen kann ein Datum in seine *Parameter* (*Jahr, Monat, Tag*) zerlegt, und diese in weitere Berechnungen eingebunden werden. Auf einfache Weise können so z.B. zu einem Geburtsjahr 18 Jahre oder zum Rechnungsdatum 30 Tage Zahlungsziel addiert werden.

```
=DatSeriell(Jahr([txtGeb-Datum]);Monat([txtGeb-Datum]);Tag([txtGeb-Datum]))
```

Die *Syntax* der Funktion *DatSeriell()* gliedert sich wie folgt:

Name der Funktion: *DatSeriell()*
in Klammern stehen die einzelnen *Parameter* eines Datums: *Jahr, Monat* und *Tag*. Mit Hilfe der gleichnamigen Funktionen (*Jahr(), Monat(), Tag()*) werden diese *Parameter* aus einem Datum (*[txtGeb-Datum]*) herausgefiltert und können in weitere Berechnungen eingebunden werden – wie das nachfolgende Beispiel zeigt:

```
=DatSeriell(Jahr([Geburtsdatum]);Monat([Geburtsdatum])+5;Tag([Geburtsdatum]))
```

Mit obiger *Formel* werden zum Geburtsmonat 5 Monate addiert.

Soll diese Funktionsweise in eine *Prozedur* im *Visual Basic-Editor* eingebunden werden, lautet die *Funktion DateSerial()* – die restliche *Syntax* ist gleich.

```
...DateSerial(Year(Me.[txtGeb-Datum]) + 18,
Month(Me.[txtGeb-Datum]), Day(Me.[txtGeb-Datum]))...
```

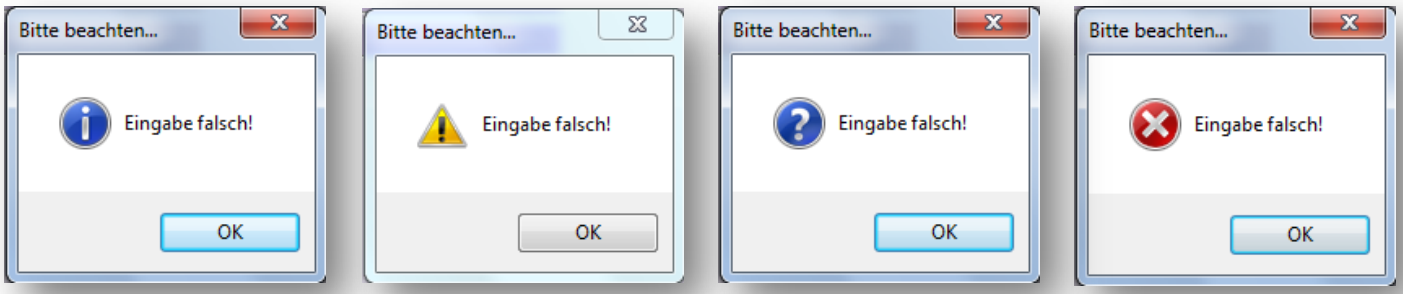
MsgBox-Konstanten

Eine *MsgBox* können Sie nicht nur durch Einbinden von *Schaltflächen* gestalten, sondern auch mit Hilfe von auffälligen *Symbolen*. Diese werden zusätzlich zur angezeigten Meldung, mittels nachfolgender *VB-Konstanten* eingebunden und können das in den *Screenshots* dargestellte *Layout* haben:

i-Symbol (`vbInformation`) – Zahlencode: 64
Ausrufezeichen (`vbExclamation`) – Zahlencode: 48
Fragezeichen (`vbQuestion`) – Zahlencode: 32
Critical-Symbol (`vbCritical`) – Zahlencode: 16

Fügen Sie einfach die entsprechende *VB-Konstante* in den Programmcode der *MsgBox* ein und beachten Sie, dass nicht nur der Text (*vb…*) sondern auch der Zahlencode eingetragen werden kann.

```
MsgBox "Eingabe falsch!", vbExclamation, "Bitte beachten..."
```



Eine Übersicht aller *VB-Konstanten* und ihrer Zahlencodes finden Sie in der *VB-Offline-Hilfe* (im *Visual Basic-Editor*) nach Eingabe des Suchbegriffs: *MsgBox*.

Funktion: *Wahl()*

Wenn Sie in ein *Formular* beispielsweise eine *Optionsgruppe* eingebunden haben und den einzelnen *Optionsfeldern* einen Wert zuweisen wollen, können Sie das mittels verschachtelter *Wenn…dann…sonst-Bedingungen* erreichen:

```
=Wenn([fraMwSt]=1;0,19;Wenn([fraMwSt]=2;0,07;0))*100
```

oder kürzer mit Hilfe der Funktion *Wahl()*:

```
=Wahl([fraMwSt];0,19;0,07;0)*100
```

Die *Syntax* der Funktion *Wahl()* gliedert sich wie folgt:

Name der Funktion: *Wahl()*
in Klammern steht zunächst der Index *[fraMwSt]* und danach der anzuzeigende Wert - in Reihenfolge. Wie das Beispiel zeigt, hat die erste Position den Wert 0,19 (19,00), die zweite Position den Wert 0,07 (7,00) usw.

Weitere *Domänenfunktionen*

Neben der bereits beschriebenen Domänenfunktion *DomWert()* stehen noch weitere *Domänenfunktionen* zur Verfügung. Diese *Funktionen* beziehen sich auf *Datensatzgruppen* (*Domänen*) und dienen dazu statistische Werte zu ermitteln. Sie entsprechen den *SQL-Aggregatfunktionen* und haben alle die gleiche *Syntax*:

Name der *Domänenfunktion*(*Ausdruck, Domäne, Kriterien*)

Folgende Werte können mit den nachfolgend aufgeführten *Domänenfunktionen* ermittelt werden:

Mittelwert:	*DomMittelwert*	(Funktionsname unter *VBA*: *DAvg*)
Anzahl:	*DomAnzahl*	(Funktionsname unter *VBA*: *DCount*)
kleinster Wert:	*DomMin*	(Funktionsname unter *VBA*: *DMin*)
größter Wert:	*DomMax*	(Funktionsname unter *VBA*: *DMax*)
erster Wert:	*DomErsterWert*	(Funktionsname unter *VBA*: *DFirst*)
letzter Wert:	*DomLetzterWert*	(Funktionsname unter *VBA*: *DLast*)
Summe*:*	*DomSumme*	(Funktionsname unter *VBA*: *DSum*)