10.2.3 Select .. Case .. End Select

Die Syntax für diese Kontrollstruktur mag Sie zuerst mit ihrer Vielfalt überraschen – sie ist aber im praktischen Einsatz schnell umzusetzen:

```
SELECT [ CASE ] Expression

  [ CASE [ Expression ] [ TO Expression #2 ] [ , ... ]
    ... ]

  [ CASE [ Expression ] [ TO Expression #2 ] [ , ... ]
    ... ]

  [ CASE LIKE Expression [ , ... ]
    ... ]

  [ { CASE ELSE | DEFAULT }
    ... ]

END SELECT
```

10.2.3.1 Hinweise

- Die Notation von *Case* nach *Select* ist optional.
- Es wird ein Ausdruck als Selektor angegeben und dann der Code in der entsprechenden passenden Case-Anweisung ausgeführt.
- Wenn kein Case-Ausdruck mit dem Selektor übereinstimmt, wird die Default- bzw. Case-Else-Anweisung – wenn vorhanden – ausgeführt.
- Eine Case-Anweisung ist ein Einzelwert, eine durch Komma getrennte Liste von Einzelwerten oder ein Intervall von Werten, die durch das To-Schlüsselwort getrennt sind.
- Fallunterscheidungen können Sie auch mit dem Like-Operator realisieren.
- In ausgewählten Fällen ist die Kontrollstruktur *Select..Case..End_Select* einer verschachtelten Kontrollstruktur *If..Then..Else_If_End_If* – bei gleicher Wirkung – vorzuziehen, weil sie dem Quelltext eine gut lesbare Struktur verleiht.

10.2.3.2 Beispiele

Die Beispiele zeigen vorwiegend Quelltext-Ausschnitte und werden kurz kommentiert.

Beispiel 1

Die Auswahl wird für verschiedene (Zahlen-)Bereiche mit unterschiedlichen Selektoren vorgenommen:

```
Public Sub GetInformation(iNumber As Integer)
  Select Case iNumber
    Case 0
      Print "Auswahl = 0"
    Case 1 To 2
      Print "Auswahl 1 oder 2"
    Case 3 To 5, 7 To 9
      Print "Auswahl zwischen 2-5 oder 7-9"
    Case 6
       Print "Auswahl = 6"
    Case Else
      Print "Auswahl nicht im Bereich von 0-9"
    End Select
End ' GetInformation(iNumber As Integer)

' Public Sub GetInformation(iNumber As Integer)
'   Select CStr(iNumber)
'     Case "0"
'       Print "Auswahl 0"
'     Case Like "[1-2]"
'       Print "Auswahl 1 oder 2"
'     Case Like "[3-57-9]" ' Case Like "[345789]"
'       Print "Auswahl zwischen 3-5 oder 7-9"
'      Case "6"
'       Print "Auswahl 6"
'     Case Else
'       Print "Auswahl nicht im Bereich von 0-9"
'     End Select'
' End ' GetInformation(iNumber As Integer)
```

Als Selektoren werden ein Zahlenwert (0 und 6), ein Zahlenbereich 1-2 oder eine Liste von zwei Zahlenbereichen 3-5 und 7-9 eingesetzt. Der auskommentierte Quelltext setzt für die Selektoren den Like-Operator ein.

Mit diesem Quelltext werden nach einem Klick auf den entsprechenden Button

```
Public Sub btnSelectedRange_Click()
  GetInformation(0)
  GetInformation(2)
  GetInformation(8)
  GetInformation(6)
  GetInformation(13)
End ' btnSelectedRange_Click()
```

folgende Zeilen in der Konsole der IDE ausgegeben:

```
Auswahl = 0
Auswahl 1 oder 2
Auswahl zwischen 2-5 oder 7-9
Auswahl = 6
Auswahl nicht im Bereich von 0-9
```

Beispiel 2

Zuerst wird eine If..Then..Else_If_End_If-Kontrollstruktur für drei *alternative* Fälle angegeben:

```
For iNumber = 0 To xmlNode.Children.Count – 1

    sBuffer = xmlNode.Children[iNumber].Name
    If sBuffer = "title" Then
       sTitle = xmlNode.Children[iNumber].Value
    Else If sBuffer = "link" Then
       sItemLink = xmlNode.Children[iNumber].Value
    Else If sBuffer = "description" Then
       sDescription = xmlNode.Children[iNumber].Value
    Endif

Next
```

und dann die Umsetzung in eine adäquate Select..Case..End_Select-Kontrollstruktur:

```
For iNumber = 0 To xmlNode.Children.Count - 1

    sBuffer = xmlNode.Children[iNumber].Name
    Select Case sBuffer
      Case "title"
        sTitle = xmlNode.Children[iNumber].Value
      Case "link"
        sItemLink = xmlNode.Children[iNumber].Value
      Case "description"
        sDescription = xmlNode.Children[iNumber].Value
    End Select

Next
```

Beispiel 3

```
PUBLIC SUB Form_KeyPress()

  IF Key.Control
     SELECT CASE Workspace1.Children.Find(Workspace1.ActiveWindow)
       CASE 0 TO Workspace1.Children.Count - 2
         Workspace1.ActiveWindow =
         Workspace1.Children[Workspace1.Children.Find(Workspace1.ActiveWindow) + 1]
       CASE Workspace1.Children.Count - 1
         Workspace1.ActiveWindow = Workspace1.Children[0]
       CASE ELSE
       ' Alternative ...
     END SELECT
  ENDIF

END ' Form_KeyPress()
```

Das erste *Case* prüft eine Liste von Werten. Eine alternative Auswahl existiert bei 'CASE ELSE' nicht – ist aber vorgesehen.

Beispiel 3

```
Public Sub GetStatus(sStatus As String)
  Select sStatus
    Case Like "[+Pp]*"
      Message.Info(IIf(sStatus Begins "+", "Status: positiv", "Status: " & sStatus))
    Case Like "[-Nn]*"
      If sStatus Begins "-" Then sStatus = "negativ"
      Message.Info("Status: " & sStatus)
    Default
      Message.Error("Der Status konnte NICHT ermittelt werden!")
  End Select
End ' GetStatus(sStatus As String)
```

- Dem *Select* in der zweiten Zeile folgt hier kein *Case*, weil dessen Angabe optional ist.
- Die Fallunterscheidungen werden in den beiden regulären Fällen über den Like-Operator (→ Kapitel 8.3.3 LIKE) umgesetzt.
- Die Aufbereitung der Meldungen wird auf unterschiedliche Weise vorgenommen.
- Die Fehlermeldung wird nur dann angezeigt, wenn keiner der beiden regulären Fälle zutrifft.

Der Aufruf von GetStatus("Status ok") in der folgenden Prozedur

```
Public Sub btnSelect_Click()
  GetStatus("Status ok")
End ' btnSelect_Click()
```

erzeugt diese Fehler-Meldung in einem separaten Fenster: