22.5.3 Klasse DataSource (gb.db.form)

Eine DataSource ist ein Container-Steuerelement, das allen Steuerelementen in diesem Container Daten aus einer Datenbank (rekursiv) zur Verfügung stellt. Die Datenbank wird über die Connection-Eigenschaft festgelegt.

Welche DB-Daten tatsächlich – zum Beispiel zur Anzeige – zur Verfügung stehen sollen, können Sie über die Eigenschaften *Table* und *Filter* festlegen:

```
  DataSource1.Connection = DBCS.DBConnection
'---   Filtering   ------------------------------------------------------------
  DataSource1.Table = "contacts"
  DataSource1.Filter = "id > 4 AND wohnort <> 'Berlin' AND wohnort NOT LIKE 'G%'"
'-----------------------------------------------------------------------------
  DataSource1.Sort = "wohnort"
```

22.5.3.1 Eigenschaften

Die Klasse *DataSource* verfügt über die folgenden Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Connection	Connection	Setzt die Connection (DB-Verbindung), die von der DataSource verwendet werden soll. Ist keine DB-Verbindung angegeben, dann wird die Standard-Connection verwendet.
Count	Integer	Es wird die Anzahl der Datensätze ausgelesen; bezogen auf die ausgewählte DB-Tabelle in der festgelegten Datenbank.
Current	Variant	Alle primären Schlüssel des aktuellen Datensatzes werden in einem Variant-Array gespeichert.
Filter	String	Setzt ein Filter für den Container – ähnlich einer WHERE-Klausel in einer SQL-Anweisung – als optionale Eigenschaft, um allen Steuerelementen im Container Daten aus der Datenbank bereits gefiltert zur Verfügung zu stellen. Der Vorgabewert für die Filter-Eigenschaft ist ein leerer String. Dann werden die Daten *nicht* gefiltert. Sie können den Wert der Eigenschaft auch auslesen.
IgnoreParent	Boolean	Wenn diese Eigenschaft auf TRUE gesetzt ist, filtert die DataSource ihren Inhalt nicht mit den Primär-Schlüsseln ihrer übergeordneten DataSources. Es wird sich so verhalten, als wäre es eine Root-DataSource.
Index	Integer	Gibt den Index des aktuellen Datensatzes zurück.
Modified	Boolean	Gibt True zurück, wenn der aktuelle Datensatz von einem Steuerelement im Container 'DataSource' geändert wurde.
ReadOnly	Boolean	Hat die Eigenschaft den Wert True, so können die DB-Daten nicht geändert werden. Sie können den Wert der Eigenschaft auch auslesen.
Sort	String	Setzt das DB-Tabellen-Feld, nach dem sortiert werden soll. Ist die Eigenschaft nicht gesetzt, dann werden die DB-Daten nicht sortiert (Standard). Sie können den Wert der Eigenschaft auch auslesen.
Table	String	Setzt den Namen der DB-Tabelle in der gewählten Datenbank, die benutzt werden soll. Sie können den Wert der Eigenschaft auch auslesen.

Tabelle 22.5.3.1.1 : Eigenschaften der Klasse DataSource

22.5.3.2 Methoden

Die Klasse *DataSource* verfügt über diese speziellen Methoden:

Methode	Rückgabetyp	Beschreibung
Cancel	-	Bricht die aktuelle Änderung der untergeordneten Steuerelemente ab und setzt alle Änderungen in den untergeordneten Steuerelementen zurück.
Create ([bUpdate As Boolean])	-	Speichert Daten, die in den DataControls stehen und löscht dann deren Inhalt. Hat der optionale Parameter 'bUpdate' den Wert False, dann werden die Daten nicht sofort übernommen – bis die Refresh-Methode benutzt wird.

Methode	Rückgabetyp	Beschreibung
MoveFirst ( )	Boolean	Verschiebt den aktuellen Datensatz-Zeiger auf den ersten Datensatz und gibt False zurück, wenn die Verschiebung möglich war. Wenn es keinen Datensatz in der ausgewählten DB-Tabelle gibt oder wenn der aktuelle Datensatz geändert wurde, wird die Verschiebung abgebrochen und TRUE zurückgegeben.
MoveLast ( )	Boolean	Verschiebt den aktuellen Datensatz-Zeiger auf den letzten Datensatz und gibt False zurück, wenn die Verschiebung möglich war. Wenn es keinen Datensatz in der ausgewählten DB-Tabelle gibt oder wenn der aktuelle Datensatz geändert wurde, wird die Verschiebung abgebrochen und TRUE zurückgegeben.
MoveNext ( )	Boolean	Verschiebt den aktuellen Datensatz-Zeiger auf den nächsten Datensatz und gibt False zurück, wenn die Verschiebung möglich war. Wenn es keinen Datensatz in der ausgewählten DB-Tabelle gibt oder wenn der aktuelle Datensatz geändert wurde, wird die Verschiebung abgebrochen und TRUE zurückgegeben.
MovePrevious ( )	Boolean	Verschiebt den aktuellen Datensatz-Zeiger auf den vorhergehenden Datensatz und gibt False zurück, wenn die Verschiebung möglich war. Wenn es keinen Datensatz in der ausgewählten DB-Tabelle gibt oder wenn der aktuelle Datensatz geändert wurde, wird die Verschiebung abgebrochen und TRUE zurückgegeben.
MoveTo (Index As Integer)	Boolean	Verschiebt den aktuellen Datensatz-Zeiger auf den Datensatz mit dem angegebenen Index und gibt False zurück, wenn die Verschiebung möglich war. Wenn es keinen Datensatz in der ausgewählten DB-Tabelle gibt oder wenn der aktuelle Datensatz geändert wurde, wird die Verschiebung abgebrochen und TRUE zurückgegeben
Remove ()	Boolean	Löscht den aktuellen Datensatz aus der DB-Tabelle und gibt False zurück, wenn das Löschen *erfolgreich* war, sonst True.
Reset ()	-	Setzt den internen Connection-Metadaten-Cache zurück, der von DataSource benutzt wird.
Refresh	-	Zeichnet alle Steuerelemente im Container neu. Der Neuzeichnen wird verzögert und erst beim nächsten Aufruf der Ereignisschleife verarbeitet. Wenn Sie eine sofortige Aktualisierung benötigen, rufen Sie WAIT auf, nachdem Sie diese Methode verwendet haben.
ResetAll ()	-	Setzt den internen Connection-Metadaten-Cache *vollständig* zurück.
Save ([bMessage As Boolean])	Boolean	Schreibt die Werte von den verwendeten DataControls in die DB-Tabelle, sofern eine Verbindung zu ihr besteht. Wenn das Speichern *erfolgreich* war gibt die Funktion True zurück, sonst False.
Update ()	-	Lädt die Datasource-Daten erneut und aktualisiert alle Steuerelemente im Container entsprechend.

Tabelle 22.5.3.2.1 : Methoden der Klasse DataSource

22.5.3.3 Ereignisse

Die Klasse *DataSource* hat nur drei spezielle Ereignisse.

- BeforeSave (Data As Result)
  Dieses Ereignis wird ausgelöst, bevor die Werte von den verwendeten DataControls in die DB-Tabelle gespeichert werden, sofern eine Verbindung zu ihr besteht. Die Daten sind im Ergebnis-Objekt (Daten-Typ Result) gespeichert. Verwenden Sie das Ereignis, um Daten zu ändern, bevor diese in die Datenbank-Tabelle geschrieben werden.
- BeforeDelete (Keys As Variant[])
  Dieses Ereignis wird ausgelöst, bevor ein Datensatz gelöscht wird. Keys ist ein Array mit den Werten des Primärschlüssels, der den zu löschenden Datensatz identifiziert. Sie können das Löschen abbrechen, indem Sie das Ereignis stoppen.
- Change ( )
  Das Ereignis wird ausgelöst, wenn der aktuelle Datensatz geändert wurde und alle Steuerelemente im Container aktualisiert worden sind.

Beispiel 1 – Ereignis: BeforeSave (Data As Result)

Der ausgewählte Datensatz kann vor dem Speichern noch geändert werden. Die hier angegebenen zwei potentiellen Änderungen sollen nur die Vorgehensweise demonstrieren:

```
Public Sub DataSource1_BeforeSave(Data As Result)

'-- If the date is missing, the current date is saved.
    If Data["date"] = Null Then Data["date"] = Now()

'-- If the image is missing, you will be prompted to save an image.
    If Data["picture"].Length = 0 Then
       Message.Info("Action:<br>Insert new image.")
    Else
       If Message.Question("Do you want to change the current image?", "Yes", "No") = 1 Then
          Print "Action: Change current image."
       Else
          Print "Action: No change current image."
          Stop Event
       Endif
    Endif

  Catch
    Message.Error(Error.Text & gb.NewLine & Error.Where)

End
```

Beispiel 2 – Ereignis: BeforeDelete (Keys As Variant[ ])

Im Vordergrund steht hier die Sicherheitsabfrage, ob der ausgewählte Datensatz tatsächlich gelöscht werden soll:

```
Public Sub DataSource1_BeforeDelete(Keys As Variant[])

  Dim vElement As Variant
  Dim sQuestion As String
' For Each vElement In Keys
'   Print vElement
' Next
  sQuestion = Subst("&1 `&2` &3", ("Should the record with the ID"), Keys[0], ("be deleted?"))
  If Message.Question(sQuestion, "Yes", "No") = 2 Then
     Stop Event
  Endif
End
```

22.5.3.4 Beispiele und Ergänzungen

Beispiel 3 – Löschen des aktuellen Datensatzes aus der DB-Tabelle:

```
Dim bFlag As Boolean
If DataSource1.Remove() = False Then Print "DAS LÖSCHEN DES DATENSATZES WAR ERFOLGREICH!"
```

Beispiel 4 – Setzen des DB-Tabellen-Feldes, nach dem sortiert werden soll:

```
DataSource1.Sort = "PLZ"
```

Beispiel 5 – Setzen eines Filters:

```
  MDataBase.ConnectDB("mysql", "localhost", "3606", "root", "sql", "Kontakte", "kontakte")
  DataSource1.Connection = MDataBase.hConnection
  DataSource1.Table = "kontakte"
' DataSource1.Filter = "Wohnort <> 'Leipzig' AND Wohnort NOT LIKE 'G%'" ' "Wohnort <> \"Leipzig\""
  DataSource1.Filter = "" ' Standard-Filter
```

Hinweis: In MySQL sind die Muster standardmäßig nicht von der Groß/Kleinschreibung abhängig.

Ergänzungen

In commit https://gitlab.com/gambas/gambas/commit/6902893dd50e6480a63aa2268207241aea021f3 wurde die neue Eigenschaft "IgnoreParent" im DataSource-Steuerelement hinzugefügt. Wenn diese Eigenschaft auf TRUE gesetzt ist, so filtert die DataSource ihren Inhalt nicht mit den Primärschlüsseln ihrer übergeordneten DataSources. Es wird sich so verhalten, als wäre es eine Root-DataSource.