22.5.4 Klasse DataCombo (gb.db.form)

Mit diesem Steuerelement können Sie den Wert eines Feldes über eine andere Tabelle bearbeiten, deren Primärschlüssel dieses Feld ist. Der Einsatz einer DataCombo ist nur dann sinnvoll, wenn mindestens zwei DB-Tabellen im Projekt verwendet werden.

Der Einsatz einer DataCombo sichert Ihnen, dass nur korrekte Feld-Werte (aus einer Tabelle A) in eine Tabelle B eingegeben werden können. Damit das funktioniert, müssen Sie bereits bei der Tabellen-Definition dafür sorgen, dass Sie passende Bedingungen für das entsprechende Feld deklarieren und beachten: Mit diesem Steuerelement können Sie den Wert eines Feldes über eine andere Tabelle bearbeiten, deren Primärschlüssel dieses Feld ist.

Schema Tabelle A:

```
CREATE TABLE "color" ( "color" INTEGER PRIMARY KEY AUTOINCREMENT, "name" VARCHAR(32), ... )
```

Schema Tabelle B:

```
CREATE TABLE "test" ( "id" BIGINT NOT NULL , "color" INT4 NOT NULL DEFAULT 1, "firstname" VARCHAR(16),
"name" VARCHAR(32), "birth" DATETIME, "active" BOOL, "salary" FLOAT8, "comment" TEXT, "image" BLOB , PRIMA-
RY KEY ("id") )
```

Die Klasse kann erzeugt werden. So erzeugen Sie eine neue DataCombo:

```
  Dim hDataCombo As DataCombo
  hDataCombo = New DataCombo ( Parent As Container )  As "event name"
```

22.5.4.1 Eigenschaften

Die Klasse *DataCombo* verfügt über die folgenden Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Action	String	Gibt die dem Steuerelement zugeordnete Aktionszeichenkette zurück oder setzt sie → Kapitel 12.3.3 Action.
All	String	Definiert ein zusätzliches Element, das als erster Eintrag in der DataCombo-Liste angezeigt wird. Sie können den Wert der Eigenschaft auch auslesen.
Background	Integer	Gibt die vom Steuerelement verwendete Hintergrundfarbe zurück oder setzt sie.
Table	String	Liefert oder setzt den Namen der Tabelle, die für die Anzeige der DataCombo und das Auffüllen der Popup-Liste verwendet wird. Diese Tabelle muss einen Primär-Schlüssel haben, dessen Name mit dem Inhalt der Feldeigenschaft übereinstimmt. Der Tabelleninhalt kann mit der Eigenschaft Filter gefiltert werden.
Value	Variant	Setzt den Primär-Index der Tabelle, die über die Eigenschaft DataCombo.Table festgelegt wurde und hat nichts mit dem zu tun, was in der DataCombo angezeigt wird! Sie können den Wert der Eigenschaft auch auslesen.
Display	String	Liefert oder setzt den Namen des Feldes in der Tabelle, die durch die Eigenschaft `Table` definiert ist, die für die Anzeige der Daten verwendet wird.
Field	String	Liefert oder setzt das von der DataCombo verwendete Feld. Die Daten stammen aus der ersten übergeordneten DataSource.
Filter	String	Liefert oder setzt eine SQL-WHERE-Klausel, die den Inhalt der Combo-Liste filtert.
Modified	Boolean	Gibt True zurück, wenn sich der Inhalt in der DataCombo geändert hat.
ReadOnly	Boolean	Setzt mit True den Modus der DataCombo auf 'ReadOnly'. Sie können den Wert auch auslesen.
Valid	Boolean	Gibt True zurück, wenn der Inhalt des Steuerelements gültig ist.

Tabelle 22.5.4.1.1 : Eigenschaften der Klasse DataCombo

22.5.4.2 Methoden

Die Klasse *DataCombo* verfügt nur über zwei relevante Methoden:

- GetFilter( )
  Gibt die im DataCombo-Filter verwendete SQL-WHERE-Klausel als Zeichenkette "FeldName = 'Filter' " zurück.
- Update( )
  Lädt den (aktuellen) Inhalt der DataCombo-Liste erneut.

22.5.4.3 Ereignisse

Die Klasse *DataCombo* hat nur diese zwei bedeutsamen Ereignisse, die für das Steuerelement DataCombo von Belang sind:

- Click ()
  Das Ereignis wird ausgelöst, wenn ein Eintrag in der DataCombo ausgewählt wurde.
- Validate (Value As Variant)
  Das Ereignis wird ausgelöst, wenn der Inhalt der DataCombo vor dem Speichern validiert werden muss. `Value` ist der zu prüfende Wert aus der DataCombo-Liste. Dieses Ereignis ermöglicht es dem Benutzer, spezifische Validierungsanforderungen festzulegen. Um die Daten zu invalidieren, stoppen Sie das Ereignis einfach mit STOP EVENT.

22.5.4.4 Beispiel

Das Beispiel demonstriert die Auswahl einer Farbe aus einer DataCombo. Die obere Tabelle enthält die wählbaren Farben ("Black", "White", "Red", "Green", "Blue", "Yellow" und "Transparent"). Diese Farbtabelle muss nicht angezeigt werden – das dient hier nur der Demonstration. Alle Werte des Feldes `color` der Farbtabelle bilden den Inhalt der DataCombo-Liste:

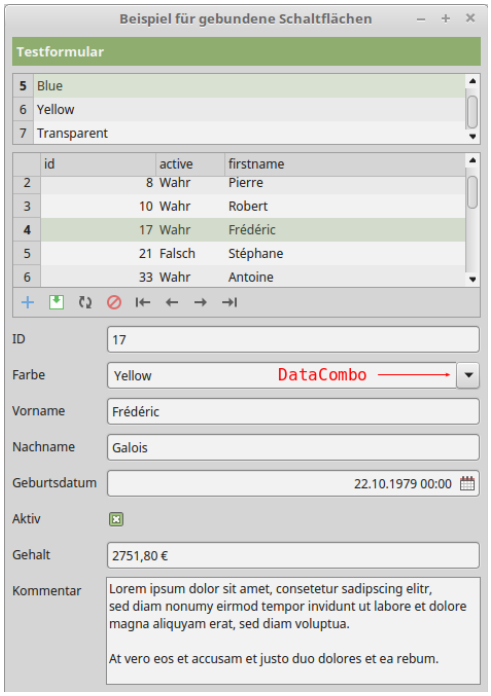

Abbildung 22.5.4.4.1: Steuer-Element DataCombo

Wenn Sie eigene Fehlermeldungen erzeugen und ausgeben wollen, dann sollten Sie das Validate-Ereignis verwenden:

```
Public aColor As String[] = ["Black", "White", "Red", "Green", "Blue", "Yellow", "Transparent"]
...
Public Sub DataCombo1_Validate(Value As Variant)

  If Not aColor.Exist(Value) Then
```

```
    Message.Error("Die Farbe ist nicht zulässig!")
    Stop Event
  Endif

End
```

Die Anweisung 'Stop Event' sorgt dafür, dass zum Beispiel NULL als Wert beim Update für die ungültige Farbe *Orange* eingetragen wird:

```
UPDATE "test" SET "id" = 3, "color" = NULL, "firstname" = 'Frédéric', ... , WHERE "id" = 3
```

Das ergibt für das Beispiel beim Update aber einen Fehler, weil mindestens eine Feld-Bedingung für das Feld "color" nicht erfüllt wird (→ NOT NULL):

```
CREATE TABLE "test" ( "id" BIGINT NOT NULL , "color" INT4 NOT NULL DEFAULT 1, "firstname" VARCHAR(16),
"name" VARCHAR(32), "birth" DATETIME, "active" BOOL, "salary" FLOAT8, "comment" TEXT, "image" BLOB , PRIMA-
RY KEY ("id") )
```

Ohne den Einsatz des Validate-Ereignisses für die DataCombo greift auch eine interne Fehlerbehandlung, weil beim Update des Datensatzes offensichtlich geprüft wird, ob der eingegebene Feld-Wert – im o.a. Beispiel in der (Farb-)Tabelle – enthalten ist oder eine Feld-Bedingung verletzt wird. Es wird eine allgemeine, korrekt formulierte Fehlermeldung in englischer Sprache ausgegeben:

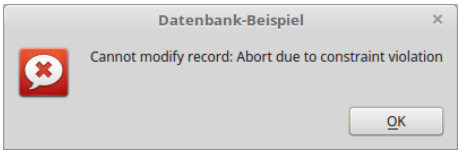

Abbildung 22.5.4.4.2: Der Datensatz kann nicht geändert werden: Abbruch wegen Verletzung von Beschränkungen