18.12 ListContainer

Die Klasse *ListContainer* (gb.form) stellt einen modifizierten Container (ListBox) zur Verfügung, der andere Komponenten aufnehmen und anzeigen kann. Diese Komponenten können auch Container sein.

Die Container-Komponente *ListContainer* werden Sie vor allem dann einsetzen, wenn Sie viele Daten mit gleicher Struktur anzeigen wollen, die bei jedem Programmstart neu eingelesen werden oder die sich zur Programm-Laufzeit ändern können. Der Anwendungszweck bestimmt dabei die Auswahl der Komponenten, die in den ListContainer eingefügt werden.

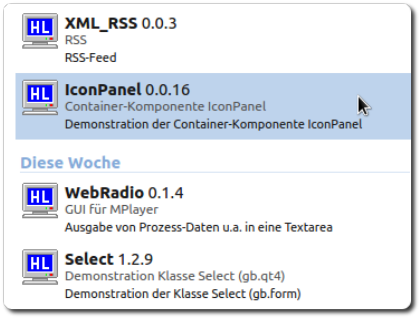

Abbildung 18.12.1: Einsatz der Komponente ListContainer (Gambas-Startbildschirm (Ausschnitt))

18.12.1 Eigenschaften

In der folgenden Tabelle werden die Eigenschaften der Komponente *ListContainer* beschrieben:

ListContainer	Datentyp	Default	Beschreibung
Border	Boolean	False	Legt fest, ob ein Rand angezeigt wird oder gibt diesen Wert zurück.
Count	Integer	1	Gibt die Anzahl der Komponenten im Container zurück.
Current	Control	-	Legt fest, welche Komponente im Container ausgewählt ist oder gibt die Komponente zurück.
Index	Integer	1	Gibt den Index der ausgewählten Komponente im Container zurück oder gibt den Index an.

Tabelle 18.12.1.1: ListContainer-Eigenschaften

18.12.2 Methoden

Die Container-Komponente besitzt fünf spezielle Methoden, deren Beschreibung Sie in der nächsten Tabelle finden:

ListContainer	Beschreibung
Clear	Es werden alle Komponenten im Container gelöscht.
EnsureVisible	Es wird sichergestellt, dass das aktuelle Element im Container sichtbar ist.
Lock	Verhindert ein Container-Refreshing solange, bis die Methode UnLock aufgerufen wurde.
UnLock	Entriegelt den Container und aktualisiert ihn, wenn das nötig ist.
Select (hChild As Control)	Markiert die ausgewählte Komponente im Container. Gegenwärtig (Gambas 3.5.1) arbeitet ein ListContainer nur im Single-Auswahl-Modus.

Tabelle 18.12.2.1: Methoden ListContainer

18.12.3 Ereignisse

Die Container-Komponente besitzt drei spezielle Ereignisse: Activate, Click und Scroll.

- Aktivate: Das Ereignis wird ausgelöst, wenn der Benutzer doppelt auf eine Komponente im Container klickt.

- Click: Das Ereignis wird ausgelöst, wenn sich die ausgewählte Komponente im Container ändert.
- Scroll: Das Ereignis wird ausgelöst, wenn der Benutzer durch den Inhalt im Container scrollt.

18.12.4 Komponenten zu einem ListContainer hinzufügen

Eine Add-Methode – die Sie sicher in der Dokumentation erwartet haben – werden Sie nicht finden, um die Komponenten Ihrer Wahl in einen ListContainer einzufügen. Folgender Ansatz aber führt zum Ziel – probieren Sie ihn unbedingt in einem kleinen Projekt aus!

```
Public Sub bExample_Click()
  Dim hCtrl As Control
  Dim bButton As Button
  Dim txbTextBox As TextBox
  Dim txtTextarea As TextArea
  Dim hForm As Form

  bButton = New Button(ListContainer2) As "B1"
  bButton.H = 24
  bButton.Text = "Button 1 oben"
  txbTextBox = New TextBox(ListContainer2) As "TB1"
  txbTextBox.H = 24
  txbTextBox.Text = "** TextBox **"
  hCtrl = New Panel(ListContainer2) As "P1"
  hCtrl.H = 32
  txtTextarea = New TextArea(ListContainer2) As "TA"
  txtTextarea.H = 80
  txtTextarea.Text = "Die ListContainer-Komponente besitzt ein Click-Ereignis."
  txtTextarea.Text &= gb.NewLine & "Es wird ausgelöst, wenn sich die Auswahl ändert."
  txtTextarea.Wrap = True
  bButton = New Button(ListContainer2) As "B2"
  bButton.H = 24
  bButton.Text = "Button 2 unten"
  hForm = New Form(ListContainer2) As "F1"
  hForm.H = 48
  hForm.Border = True
  hForm.Background = &HDFA08B
      bButton = New Button(hform) As "F1Button1"
      bButton.H = 24
      bButton.X = 8
      bButton.Y = 16
      bButton.W = 88
      bButton.Text = "Mitteilung"
  ListContainer2.Index = 3

End ' bExample_Click()

Public Sub F1Button1_Click()
  Message.Info("Ich fühle mich unter-drückt!")
End ' BF1_Click()
```

Die TextBox und die TextArea sind beschreibbar und auch ein Klick auf die 2 Button sowie auf den Button *F1Button1* im eingefügten Formular funktioniert.

18.12.5 Projekt RSS-Reader

Informationen zum Thema RSS finden Sie unter *http://www.rssboard.org/rss-specification*. Im Vordergrund des vorgestellten Projekts *RSS-Reader* jedoch steht der Einsatz der Komponente *ListContainer*.

- Zuerst wird der Datenstrom im XML-Format für einen bestimmten RSS-Feed vom angegebenen Server geladen und in einem String gespeichert.
- Dann wird der String geparst. Die obligatorischen Kanal-Elemente *Titel*, *Beschreibung* und *Link* werden ausgelesen und (temporär) gespeichert.
- Danach werden (Feed-)Formulare (→ Klasse FeedItem.Class) erzeugt, denen nur die zwei Elemente *Titel* und *Beschreibung* als Inhalt in zwei ausgewählten Komponenten (TextArea) zugewiesen werden. Das eingefügte Icon ist für alle (Feed-)Formulare gleich.
- Anschließend werden die (Feed-)Formulare einem ListContainer hinzugefügt.

Ein Klick auf ein (Feed-)Formular im ListContainer markiert das Formular, gibt einen *Index* zurück und startet den zu diesem Index passenden Link in einer WebView:

Abbildung 18.12.5.1: Projekt RSS-Reader – Programmlaufzeit

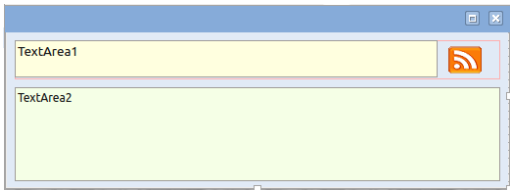

Abbildung 18.12.5.2: (Feed-)Formular zum Projekt RSS-Reader – Entwicklungszeit

Hinweise:

- Das einzufügende (Feed-)Formular können Sie selbstverständlich auch vollständig im Quelltext definieren, wie Sie das im o.a. Quelltext-Ausschnitt im unteren Abschnitt nachlesen können. Schneller und weniger fehleranfällig hinsichtlich der Anordnung der Komponenten auf dem Formular ist jedoch die Verwendung der Klasse *FeedItem.Class* → Abbildung 18.12.5.2.

- Das Projekt *RSS-Reader* sollten Sie noch weiterentwickeln, denn ihm fehlt zum Beispiel noch Einiges in Sachen Fehlervermeidung und Fehlerbehandlung sowie die Möglichkeit, neue Feeds hinzuzufügen.

Den umfangreichen Quelltext können Sie im Projekt *RSS-Reader* nachlesen, auf das Sie Zugriff im Download-Bereich haben.