Kurzmeldungen und Kurzeingaben (Dialogfenster)

Dialogfenster stellen eine weit verbreitete Möglichkeit dar, mit einem Programm zu kommunizieren. Dialogfenster haben unter Windows z. B. die Aufgabe, Informationen anzuzeigen oder Einstellungen vom Anwender eines Programms vornehmen zu lassen bzw. anzuzeigen. Über verschiedene Schaltflächen eines Dialogfensters wie Ok, Abbrechen, Hilfe, Ignorieren oder Verwerfen kann der Anwender eine jeweils unterschiedliche Reaktion des Programms bewirken.

Typen von Dialogfenstern

Modale Dialogfenster	Nichtmodale Dialogfenster

Ausgabe kurzer Texte

```
ShowMessage(AnzeigeText);
```

Die Funktion ShowMessage zeigt ein Dialogfenster (Meldungsfenster) mit einer OK-Schaltfläche an. Der Parameter AnzeigeText enthält den Text der Meldung, die innerhalb des Meldungsfensters erscheint. Als Überschrift des Meldungsfensters wird der Name der aktuellen Anwendung angezeigt.

Beispiel:

```
ShowMessage("Alles in  Ordnung!");
```

Einlesen eines kurzen Textes in eine Variable

```
S = InputBox("Überschrift","Aufforderungstext","Anfangswert")
```

Ausgabe einer Meldung mit der Möglichkeit die Benutzerreaktion einzuholen

```
<Rückgabewert> MessageBox(0,"AnzeigeText","Überschrift",Eigenschaften);
```

Beispiel:

```
Z = MessageBox(0,"Programm beenden?","Frage",MB_YESNO);
```

Die Methode MessageBox der Klasse TApplication zeigt ein Dialogfenster an, das eine Meldung, ein Bild und eine oder mehrere Schaltflächen anzeigt. Der erste Parameter ist die interne Bezugsnummer (Handle) des aktuellen Formulars (z.B. 0 für Forml). Der Wert des Parameters AnzeigeText entspricht der angezeigten Meldung. Der Parameter Überschrift wird in der Titelleiste des Dialogfensters angezeigt.

Die möglichen Werte des Parameters Eigenschaften, die Sie beliebig über den Operator OR

verknüpfen können, sind:

```
MB_ICONEXCLAMATION   Ein Ausrufezeichensymbol erscheint im Meldungsfenster.
MB_ICONINFORMATION   Ein I-Symbol (Information) erscheint im
Meldungsfenster.
MB_ICONQUESTION      Ein Fragezeichensymbol erscheint im Meldungsfenster.
MB_ICONSTOP          Ein Stopzeichensymbol erscheint im Meldungsfenster.
MB_OK                Das Meldungsfenster enthält eine OK-Schaltfläche.
MB_OKCANCEL          Das Meldungsfenster enthält eine Ok- und eine
ABBRECHEN-Schaltfläche.
MB_YESNOCANCEL       Das Meldungsfenster enthält die Schaltflächen Ja, Nein
und Abbrechen.
```

Beispiel:

```
Application->MessageBox("Ende?", "???", MB_ICONQUESTION | MB_OKCANCEL);
```

Den Rückgabewert können Sie nach dem Aufruf des Meldungsfensters auswerten.

```
IDOK                 Der Anwender hat die Schaltfläche Ok gewählt.
IDCANCEL             Der Anwender hat die Schaltfläche Abbruch gewählt.
IDABORT              Der Anwender hat die Schaltfläche Abort gewählt.
IDRETRY              Der Anwender hat die Schaltfläche Wiederholen gewählt.
IDIGNORE             Der Anwender hat die Schaltfläche Ignorieren gewählt.
IDYES                Der Anwender hat die Schaltfläche Ja gewählt.
IDNO                 Der Anwender hat die Schaltfläche Nein gewählt.
```

Beispiel:

```
if (Application->MessageBox(..) == IDOK) ...
```

- Der vierte Parameter p4 setzt sich aus drei Zahlen zusammen, welche addiert werden:
  - p4 = Symbol + Schaltfläche + Standardeingabe
  - Symbol bestimmt das angezeigte Symbol in der Box: 0 = ohne Symbol, 16 = Stoppsymbol „X“, 32 = Fragesymbol „?“, 48 = Warnsymbol „!“, 64 = Informationssymbol „i“.
  - Schaltfläche bestimmt die Anzahl und Art der gezeigten Schaltflächen in der Box: 0 = nur „OK“, 1 = „OK - Abbrechen“, 2 = „Abbrechen - Wiederholen - Ignorieren“, 3 = „Ja - Nein - Abbrechen“, 4 = „Ja - Nein“, 5 = „Wiederholen - Abbrechen“.
  - Standardeingabe gibt an, welche Schaltfläche bei Betätigung der -Taste aktiv wird: 0 = erste Schaltfläche, 256 = zweite Schaltfläche, 512 = dritte Schaltfläche.
  - Die numerischen Rückgabewerte geben an, welche Schaltfläche vom Anwender gedrückt wurde: 1 = „OK“, 2 oder 3 = „Abbrechen“, 4 = „Wiederholen“, 5 = „Ignorieren“, 6 = „Ja“ und 7 = „Nein“.

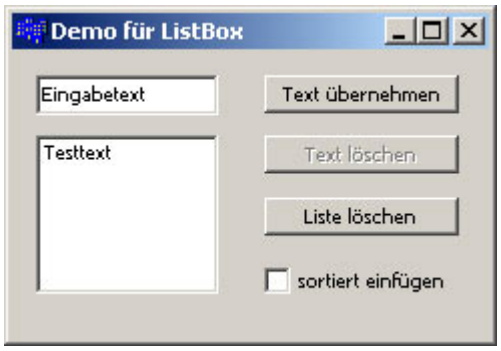
Komponente ListBox

Nützliche Funktionen/Methoden für Listboxen

Beachte: ListBox-Indizes beginnen bei 0, ListBox-Einträge (=AnsiStrings) beginnen beim Index 1.

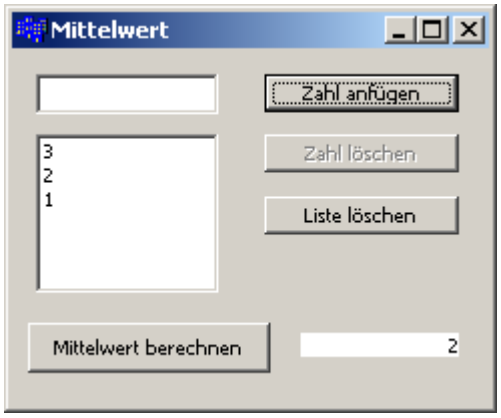
```
ListBox1->Items->Strings[Index] bzw.
ListBox1->Items[0][Index]      Zugriff auf den durch Index angegebenen
ListBox-Eintrag

ListBox1->ItemIndex            gibt den Index desjenigen ListBox-Eintrags
an,
                               der den Focus hat

ListBox1->Selected[Index]      überprüft, ob der angegebene ListBox-Eintrag
                               den Focus besitzt (d.h. ausgewählt ist)
ListBox1->Items->Strings[ListBox1->ItemIndex];
ListBox1->Items[0][ListBox1->ItemIndex];
                               Zugriff auf angeklickten Eintrag in der
ListBox.

ListBox1->Items->Add(Text);    übernimmt Text in ListBox

ListBox1->Sorted=true;         ListBox wird sortiert angezeigt

ListBox1->Count;               zählt die Anzahl der ListBox-Einträge

ListBox1->Clear();             löscht alle ListBox-Einträge

ListBox1->Items->Delete(Index);löscht den ListBox-Eintrag,
                               der den angegebenen Index besitzt

ListBox1->DeleteSelected()     löscht den ListBox-Eintrag,
                               der gerade den Focus besitzt
                               (ohne dass der Index ermittelt werden muss).

if
(SaveDialog1->Execute()){ListBox1->Items->SaveToFile(SaveDialog1->FileName);
                               Speichern des Inhaltes einer ListBox-
Komponente

if
(OpenDialog1->Execute()){ListBox1->Items->LoadFromFile(OpenDialog1->FileName
);
                               Laden des Inhaltes einer ListBox-Komponente
```

Projekt ListBox



In die Listbox soll Text übernommen werden. Außerdem sollen Texteinträge gelöscht werden. Die Buttons sollen nur dann zur Verfügung stehen, wenn es logisch erlaubt ist (Listbox löschen soll nur dann zur Verfügung stehen, wenn die Listbox nicht leer ist, Text löschen nur dann, wenn eine Zeile der Listbox ausgewählt wurde, etc.). Neue Elemente können entweder am Ende oder sortiert eingefügt werden.

Übung Mittelwert



Eine beliebige Anzahl reeller Zahlen soll eingegeben werden und in einem Listenfeld erscheinen. Durch Drücken auf den entsprechenden Schalter soll der Mittelwert berechnet und ausgegeben werden.

Übung Min Max

Eine beliebige Anzahl ganzer Zahlen soll eingegeben werden und in einer ListBox erscheinen. Durch Drücken auf den entsprechenden Schalter werden die kleinste sowie größte der eingegebenen Zahlen ausgegeben.

Übung Zahlensortierung

Eine beliebige Anzahl reeller Zahlen soll in eine Listbox eingegeben werden. Durch Drücken auf den

Button „Sortieren“ soll die Zahlenfolge richtig sortiert werden.

Hinweis: Da die Eigenschaft **Sorted** lexikographisch sortiert, ist dies für diese Aufgabe unbrauchbar. (113 wäre z.B. kleiner als 12!)

Vielmehr müssen die Zahlen der ListBox mittels eines (schon durchgenommenen) Sortieralgorithmus sortiert werden. Einfache Sortieralgorithmen hier zum Nachlesen...

Memoboxen

Beachte: MemoBox-Indizes beginnen bei 0, MemoBox-Einträge (=AnsiStrings) beginnen beim Index 1.

```
Memo1->Lines[0][zeilenindex]            Zugriff auf die durch Index
angegebene Memo-Zeile
Memo1->Lines[0][zeilenindex].Length()   ermittelt die Länge der angegebenen
Memo-Zeile
Memo1->Lines->Add(Zeilentext)           übernimmt Text einer neuen Zeile ins
Memo (d.h.nach einem Zeilenvorschub)
Memo1->SelText="text ... "              übernimmt Text an aktueller Position
ins Memo (d.h. ohne Zeilenvorschub)
Memo1->Lines->Count                     zählt die Anzahl der Memo-Zeilen
Memo1->Clear()                          löscht alle Memo-Zeilen
Memo1->Delete(zeilenindex)              löscht die entsprechende Zeile
Memo1->Insert(zeilenindex,zeileninhalt) fügt den Zeileninhalt an der
entsprechenden Zeile ein
if
(SaveDialog1->Execute()){Memo1->Lines->SaveToFile(SaveDialog1->FileName);}
                                        Speichern des Inhaltes einer Memo-
Komponente
if
(OpenDialog1->Execute()){Memo1->Lines->LoadFromFile(OpenDialog1->FileName);}
                                        Laden des Inhaltes einer Memo-
Komponente
```