

Wie... wird eine Spalte in eine Listbox eingefügt?

Teil 6 – Header und Sortierung

Voraussetzungen

- Installiertes ClassiX® System
- Codewright oder ein anderer Texteditor
- Kenntnisse über:

Grundlegender Aufbau von ClassiX®	Widget	Monitor-Fenster
Konventionen	Action List	ObjectListView
Klassen	Event	SetFormat
Vererbung (Inheritance)	Message	Objektinspektor
Modul	Variablen	

Aufgabenstellung:

In den letzten „Kochrezepten“ haben wir erfolgreich neue, einfache und komplexe, Datenfelder in eine Listbox eingefügt und formatiert. Leider fehlen für die neuen Spalten noch die Spaltenüberschriften. Diese zu erstellen, wollen wir jetzt lernen.

Inhalt

- Definition des Headers
- Änderung des Headers

Definition des Headers

Der **Header** einer Listbox ist ein eigenes, von ihr unabhängiges Widget und wird auch unabhängig definiert.

Die Syntax lautet:

```
Header(name, flags, x, y, w, h, objectBox)
```

Parameter:

name	*	Identifikator des headers hier „ListBoxHeader“
flags		Flags
x	*	Position X (in Minicells)
y	*	Position Y (in Minicells)
w	*	Breite (in Minicells)
h	*	Höhe (in Minicells)
objectBox	*	Name einer Objektbox hier „ListBoX“

* - Pflichtparameter

Die Childobjekte eines Headers sind oft - aber nicht notwendigerweise - **Prompts**, mit denen die Überschriften für eine **Objektbox** angegeben werden. Durch den letzten Parameter wird eine Verbindung zur Objektbox hergestellt, so dass die Überschriften am horizontalen Scrolling der Box teilnehmen. Die Konventionen für die Namensvergabe von Headern lauten: an den Objectboxnamen wird „Header“ angehängt. In unserem Fall also „**ListBoxHeader**“.

Änderung des Headers

Wir suchen jetzt „ListBoxHeader“ im Code und werden an 2 Stellen im Zusammenhang mit Header fündig: in item.mod und itemedt.mod. Im letzteren Modul handelt es sich um eine ATTACH Anweisung, die wir z.Zt. ignorieren können. In item.mod finden wir den Header dann als Widget des Windows „SelectWin“. Der Code für unseren ListBox Header lautet:

```
Header(ListBoxHeader, 7, 0, 862, 22, ListBox)
```

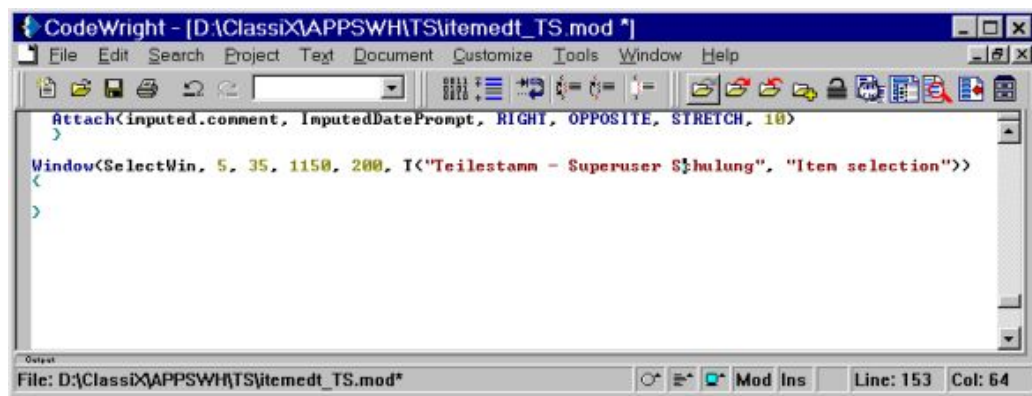
```
{
  Prompt(uniqueID, 0, 2, T("Nummer", "Number"))
//.
//.
//.
// Childobjekte folgen:
  Prompt(NONE::StockFlag, HIDDEN, 770, 2, T("Lager", "Stock"))
  Prompt(NONE::Monitor("CX_STOCK_ACCOUNT").current.quantity, HIDDEN, 980, 2, T("Menge", "Quantity"))
  Prompt(NONE::Monitor("CX_STOCK_ACCOUNT").average.pricePer, HIDDEN, 1000, 2, T("Ø Wert", "Ø value"))
  Prompt(NONE::PurchaseItem().leadTime, HIDDEN, 960, 2, T("Bestellzeit", "Lead time"))
  Prompt(NONE::Product().leadTime, HIDDEN, 1070, 2, T("Eindeckungszeit", "Production time"))
  Prompt(NONE::PurchaseItem().priceDiscountPointer.Amount(), HIDDEN, 1240, 2, T("EK-Preis", "Purchase price"))
  Prompt(NONE::SparePartFlag, HIDDEN, 1340, 2, T("E-Teil", "Spare part"))
  Prompt(NONE::QualityAssuranceFlag, HIDDEN, 1440, 2, T("QS", "QA"))
  Prompt(NONE::StructureByID("ITEM_STATE").mlShortName, HIDDEN, 1540, 2, T("Status", "State"))
}
```

Auch hier gilt: geändert wird nur im abgeleiteten Kundenmodul!

Da der Header ein Childobjekt des Widget „Window(SelectWin,..)“ ist, welches in itemedt_TS nicht verändert wurde, muss auch „Window(SelectWin,..)“ im Modul itemedt_TS neu definiert werden.

Wir suchen also im Vererbungspfad das *letzte* Auftreten der Definition des Widget (damit die Ableitungen nicht wieder überschrieben werden) und kopieren es ans Ende von itemedt_TS:

```
Window(SelectWin, 5, 35, 1150, 200, T("Teilestamm - TestSelektion", "Item
selection"))
{
}
```



Danach kopieren wir den Header aus dem Modul item als Childobjekt in das Widget. Bis auf die Definitionszeile und den Prompt für **Monitor("CX_STOCK_ACCOUNT").current.quantity** löschen wir alle Childobjekte von Header, da wir sie nicht überschreiben wollen:

```
Window(SelectWin, 5, 35, 1150, 200, T("Teilestamm - TestSelektion", "Item
selection"))
{
  Header(ListBoxHeader, 7, 0, 862, 22, ListBox)
  {
    Prompt(NONE::Monitor("CX_STOCK_ACCOUNT").current.quantity, HIDDEN, 0, 0,
    T("Menge", "currant quantity"))
  }
}
```

Dann ändern wir den **Prompt** folgendermaßen ab:

```
Prompt(NONE::Monitor("CX_STOCK_ACCOUNT").received.quantity, HIDDEN, 0, 0,
T("WE-Menge", "Received quantity"))
```

NONE:: Die "leere" Klasse (muss historisch bedingt) angegeben werden.

Monitor("CX_STOCK_ACCOUNT").received.quantity :

Name des Datenelements, ordnet die Überschrift der Spalte zu

Hidden: Es wird kein Suchfeld ausgegeben

0,0: Koordinaten egal bei Hidden

T: Überschrift und 1. Fremdsprache

Übungsaufgabe:

Wir hatten in Teil 4 auch ein einfaches Datenelement eingefügt (qualityAssurance). Ändern Sie den Header so ab, dass auch diese Spalte eine Überschrift hat.

Zusammenfassung

Wir lernten, wie wir eine Listbox nach unseren Vorstellungen mit Hilfe der von SetFormat verwalteten Formatelemente gestalten können, und lernten die wichtigsten Parameter und ihre Ausprägungen kennen.

Weiterführendes

Gratulation!

Die ersten Gänge im Menü für Superuser haben Sie erfolgreich verdaut.

Beim nächsten Workshop erwartet Sie sowohl leicht- als auch schwerverdauliches:

Einfügen von Eingabefeldern in Masken.

Die Definition von dynamischen Datenfeldern (Slots).