

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

Teil 2: Wie finde und ändere ich den Code?

## Voraussetzungen

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

<a href="#">Grundlegender Aufbau von ClassiX®</a>	<a href="#">Widget</a>	<a href="#">Monitor-Fenster</a>
<a href="#">Konventionen</a>	<a href="#">Action List</a>	<a href="#">ObjectListView</a>
<a href="#">Klassen</a>	<a href="#">Event</a>	
<a href="#">Vererbung (Inheritance)</a>	<a href="#">Message</a>	
<a href="#">Modul</a>	<a href="#">Variablen</a>	


## Aufgabenstellung:

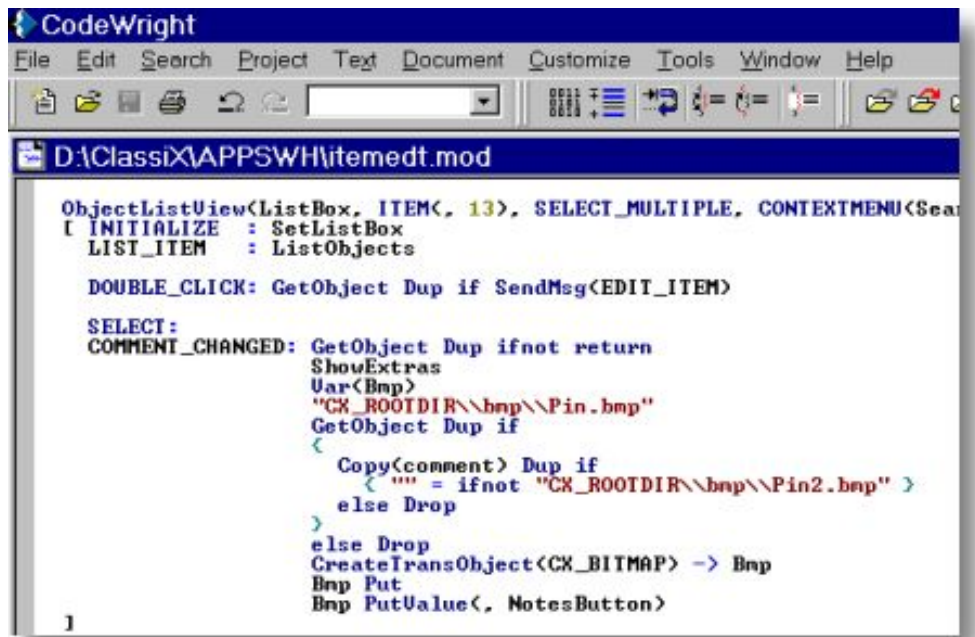
In dem letzten „Kochrezept“ lernten wir wie Vererbung funktioniert und welche Auswirkungen das auf unseren Code hat. Jetzt soll dieses Wissen praktisch angewandt werden.

### Inhalt

- Aufruf von Codewright
- Finden der Listbox Definition
- Finden der Modul Dateien
- Ändern der Modul Dateien

## Aufruf von Codewright

Um den Code verändern zu können müssen wir uns den Vererbungsbaum “längshangeln”. Wir klicken auf den Button  im Monitor-Fenster, der den Codewright Editor mit dem Source-Code des ausgewählten Windowobjekts aufruft. Der Editor öffnet dann das Modul itemedt.mod, da dort der Code das erste mal erscheint:




```
CodeWright
File Edit Search Project Text Document Customize Tools Window Help
D:\Class\X\APPS\WH\itemedt.mod
ObjectListView(ListBox, ITEM<, 13>, SELECT_MULTIPLE, CONTEXTMENU<Search
[ INITIALIZE : SetListBox
  LIST_ITEM : ListObjects

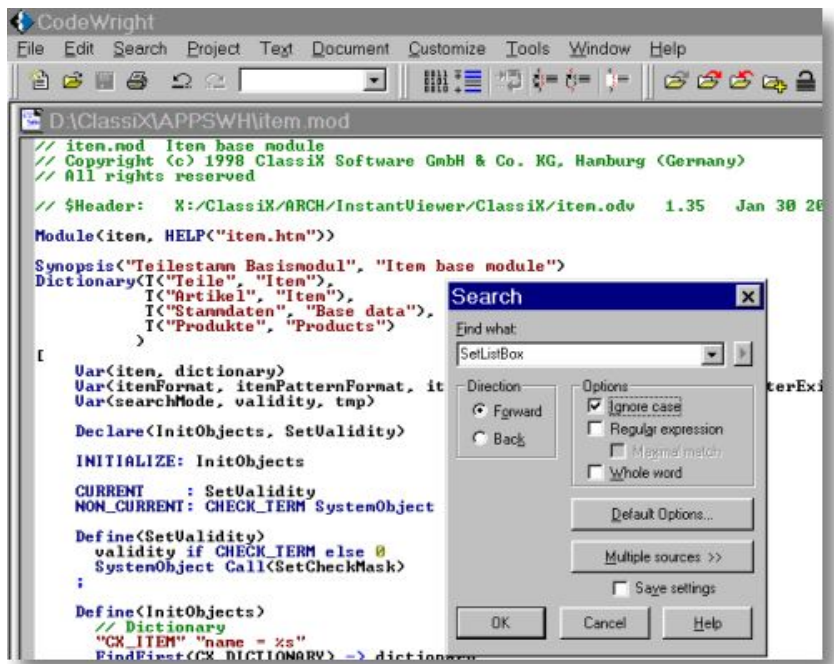
DOUBLE_CLICK: GetObject Dup if SendMsg(EDIT_ITEM)

SELECT:
COMMENT_CHANGED: GetObject Dup ifnot return
  ShowExtras
  Uar<Bmp>
  "CK_ROOTDIR\bmp\Pin.bmp"
  GetObject Dup if
  {
    Copy<comment> Dup if
    { "" = ifnot "CK_ROOTDIR\bmp\Pin2.bmp" }
    else Drop
  }
  else Drop
  CreateTransObject<CX_BITMAP> -> Bmp
  Bmp Put
  Bmp PutValue<, NotesButton>
```

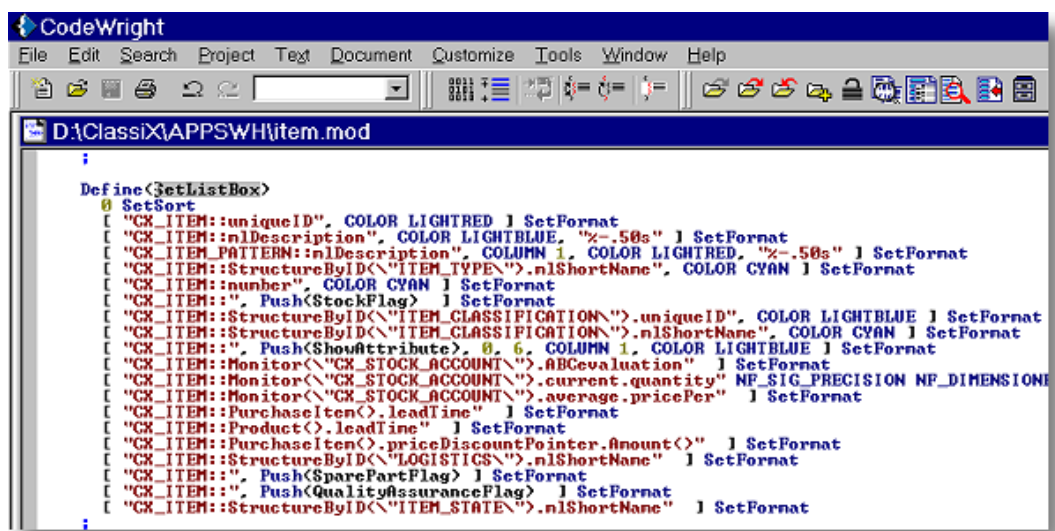
Die Listbox wird hier aufgerufen, aber die Definition wird nur als Message in die Aktionsliste gesetzt. Unsere Listbox ist also abgeleitet.

## Finden der Listbox Definition

Um die Definition der Listbox zu erhalten, rufen wir über den Button  Codewright mit dem Source-Code des 1. Auftretens des Windowobjekts auf:

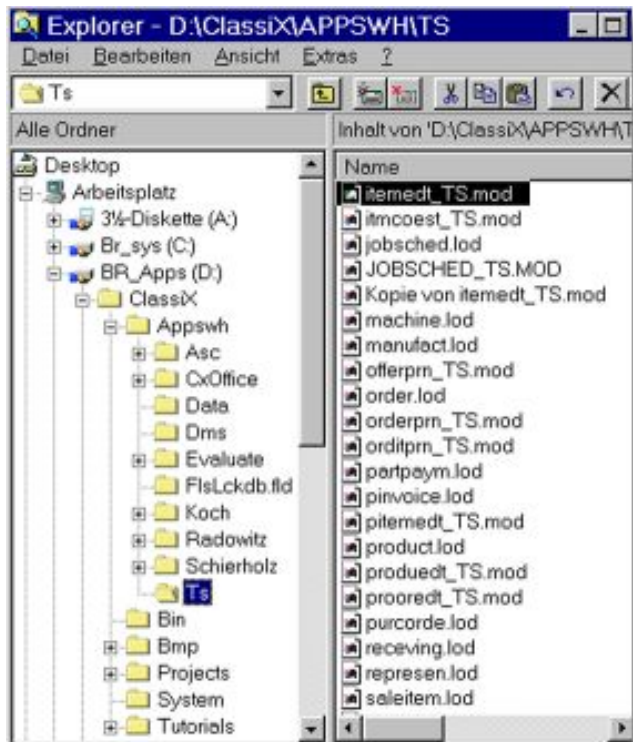


Mit Search suchen wir die Definition unseres Elements.



## Finden der Modul Dateien

Der von uns zu ändernde Code steht, wie in der Titelleiste des Fensters zu sehen, in der Datei (Laufwerk):\ClassiX\APPSWH\item.mod. Damit handelt es sich um ein Modul des AppsWarehouse® und **darf vom Superuser nicht geändert werden**. Was ist zu tun? Unsere Listbox muss neu definiert werden in den individuellen Modulen des Kunden, die sich immer in einem Unterverzeichnis von APPSWH befinden:

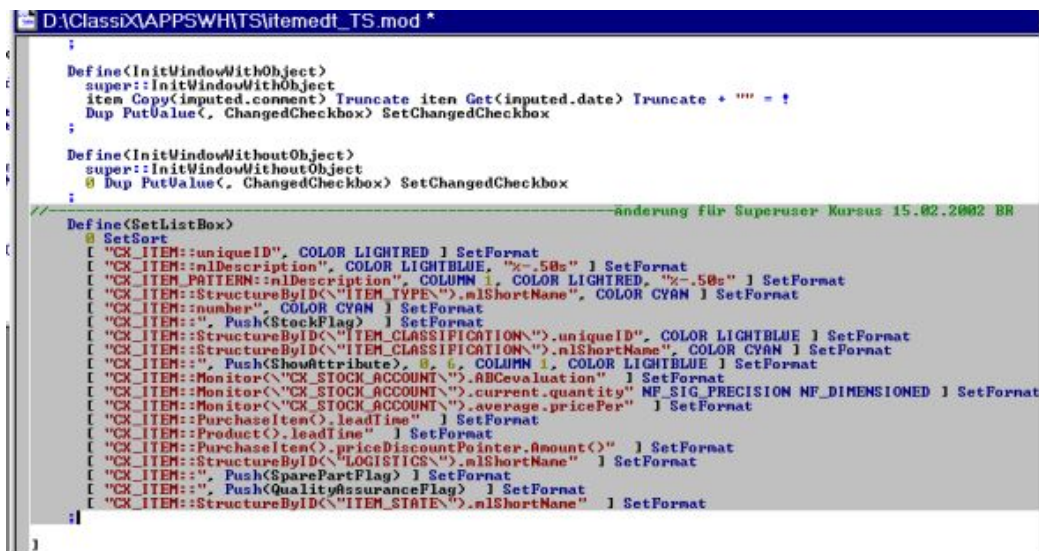


In diesem Fall ändern wir laut Konvention die Datei itemedit\_TS.mod.

## Ändern der Modul Dateien

Mit Hilfe von DragDrop ziehen wir die Datei in den Codewright Editor.

Im Fenster (laufwerk):\ClassiX\APPSWH\item.mod markieren wir das gesamte Define der Listbox und kopieren es an das Ende der Action List des Moduls itemedt\_TS:



```
D:\ClassiX\APPSWH\TS\itemedt_TS.mod *
;
Define<InitWindowWithObject>
super::InitWindowWithObject
item Copy(inputed.comment) Truncate item Get(inputed.date) Truncate + "" = !
Dup PutValue(. ChangedCheckbox) SetChangedCheckbox
;

Define<InitWindowWithoutObject>
super::InitWindowWithoutObject
Dup PutValue(. ChangedCheckbox) SetChangedCheckbox
;

Define(SetListBox)
;-----Änderung für Superuser Kursus 15.02.2002 BR
@ SetSort
[ "CK_ITEM::uniqueID", COLOR LIGHTRED ] SetFormat
[ "CK_ITEM::nlDescription", COLOR LIGHTBLUE, "%-50s" ] SetFormat
[ "CK_ITEM::PATTERN::nlDescription", COLUMN 1, COLOR LIGHTRED, "%-50s" ] SetFormat
[ "CK_ITEM::StructureByID<"ITEM_TYPE">::nlShortName", COLOR CYAN ] SetFormat
[ "CK_ITEM::number", COLOR CYAN ] SetFormat
[ "CK_ITEM::", Push(StockFlag) ] SetFormat
[ "CK_ITEM::StructureByID<"ITEM_CLASSIFICATION">::uniqueID", COLOR LIGHTBLUE ] SetFormat
[ "CK_ITEM::StructureByID<"ITEM_CLASSIFICATION">::nlShortName", COLOR CYAN ] SetFormat
[ "CK_ITEM::", Push(ShowAttribute), @, @, COLUMN 1, COLOR LIGHTBLUE ] SetFormat
[ "CK_ITEM::Monitor<"CK_STOCK_ACCOUNT">::ABCEvaluation" ] SetFormat
[ "CK_ITEM::Monitor<"CK_STOCK_ACCOUNT">::current.quantity" MF_SIG_PRECISION MF_DIMENSIONED ] SetFormat
[ "CK_ITEM::Monitor<"CK_STOCK_ACCOUNT">::average.pricePer" ] SetFormat
[ "CK_ITEM::PurchaseItem().leadTime" ] SetFormat
[ "CK_ITEM::Product().leadLine" ] SetFormat
[ "CK_ITEM::PurchaseItem().priceDiscountPointer.Amount()" ] SetFormat
[ "CK_ITEM::StructureByID<"LOGISTICS">::nlShortName" ] SetFormat
[ "CK_ITEM::", Push(SparePartFlag) ] SetFormat
[ "CK_ITEM::", Push(QualityAssuranceFlag) ] SetFormat
[ "CK_ITEM::StructureByID<"ITEM_STATE">::nlShortName" ] SetFormat
;
]
```

Die Änderung wird mit einem Kommentar dokumentiert. Der Kommentar hat mindestens den Grund der Änderung, das Datum und den Ausführenden zu enthalten. Jetzt kann das gewünschte Datenfeld an der richtigen Stelle eingefügt werden.

Wie kann man herausfinden, wie das Datenfeld heißt, das als Spalte eingefügt werden soll? Dabei helfen die **Inspektoren**.

## Zusammenfassung

Wir lernten, wie die Module zu finden sind, die den Code des zu ändernden Widgets enthalten. Es wurde wiederholt, welche Module der Superuser ändern darf und welche nicht.

## Weiterführendes

**Teil 3 – Wie finde ich ein Datenfeld?**

**Teil 4 – Komplexe Datentypen!**

**Teil 5 – Schönheitspflege für die Listbox**

**Teil 6 – Wie finde/verändere ich die Spaltenüberschrift?**