FF-Dialogeditor

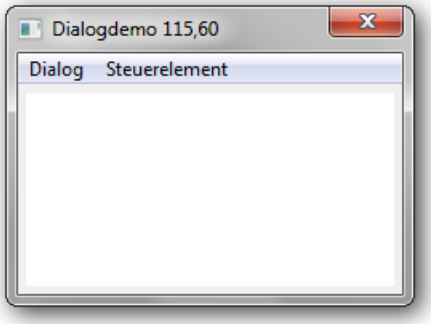
Harald Maeckler

Skripte haben den Nachteil, dass für sie keine Bedienoberflächen programmiert werden können, wie man sie beispielsweise von Visual-Basic-Projekten her kennt. Man muss sich mit Eingabe- und Meldeboxen behelfen. Der FFSkript-Editor schließt diese Lücke, indem er mit dem Dialogeditor eine Möglichkeit anbietet, für den Anwender einfach benutzbare Oberflächen zu konstruieren bzw. zu programmieren.
In den folgenden Beispielen wird als Skriptsprache **V**isual**B**asic**S**kript verwendet. Sie sind aber sicher auch von den Freunden der **J**ava**S**kript-Sprache nach zu vollziehen.

Dialogfeld

Nachdem mit F11 der FFSkript-Editor gestartet wurde, öffnet sich der Dialogeditor durch anklicken mit der linken Maustaste (rechts unten). Unter "*Dialog* / *Neuer Dialog*" gibt man einen Namen für den zu erstellenden Dialog ein, z.B. *Dialogdemo*.

Nach Bestätigung mit OK wird das Dialogfeld mit dem vergebenen Namen und mit Breite und Höhe, hier 115, 60 angezeigt.



Durch Ziehen mit der Maus lässt sich die Größe des Feldes beliebig verändern.

Wenn man jetzt "*Dialog* / *Speichern*" anklickt, wird automatisch ein Code im Editorfenster erzeugt, der eingeschlossen durch

```
'** Start Dialog Dialogdemo **
'** End Dialog Dialogdemo **
```

das Dialogfeld beschreibt, in unserem Fall erst einmal nur

```
FF_AddDialog "Dialogdemo",115,60
```

womit der Dialog "*Dialogdemo*" mit einer Breite von 115 Dialogeinheiten und einer Höhe von 60 Dialogeinheiten angelegt ist. Die Größe des Dialogfeldes kann statt im Dialogeditor auch durch direktes Editieren geändert werden.
Es gibt übrigens keine zwingende Beziehung zwischen Dialogeinheit und Pixel. In der Regel entspricht zwar eine Dialogeinheit zwei Pixel, aber das ist letztendlich abhängig vom Display, von der Grafikkarte und vom Grafiktreiber, kann also individuell unterschiedlich sein.

Oberhalb dieses Codes gibt man nun erst einmal für das neue Skript einen Kategorienamen und den darin gewünschten Eintrag an, hier als Beispiel Demo und Dialog:

```
'FFSubmenu=Demo
'FFName=Dialog
```

und am Codeende ruft man mit *FF_ShowDialog ("Dialogdemo")* den Dialog auf.
Der komplette Code sieht jetzt wie folgt aus:

```
'FFSubmenu=Demo
'FFName=Dialog

'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",115,60
'** End Dialog Dialogdemo **

FF_ShowDialog ("Dialogdemo")
```

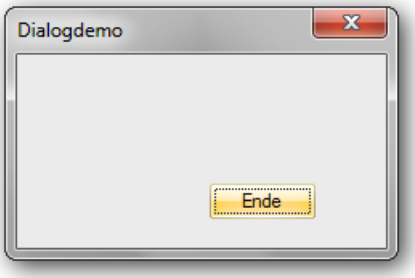
Spätestens jetzt sollte das Skript abgespeichert werden. Dabei ist darauf zu achten, dass unten die richtige Skriptart angekreuzt ist, in unserem Fall *VBScript*. Nach Betätigung von Speichern muss man unbedingt kontrollieren, ob der Skriptordner von FixFoto angewählt ist. In der Regel ist das der Ordner "*C:\Users\Benutzer\Documents\FixFoto\Script*". Nun gibt man einen sinnvollen Dateinamen, z.B. *DialogDemo* ein und bestätigt mit OK, womit das Skript als *DialogDemo.vbs* abgespeichert wird. Wenn man jetzt links Autom. Konfiguration betätigt, wird im linken Fenster die Kategorie *Demo* erzeugt (wenn nicht schon vorhanden) und dort der Eintrag *Dialog* angezeigt. Zum Test kann man jetzt auf Ausführen klicken und es wird das nackte Dialogfeld *Dialogdemo* angezeigt. Der Ausführen-Button bleibt solange rot, bis das **x** im Dialog rechts oben betätigt wird. Damit kann man zwar noch nicht viel anfangen, aber wir haben jetzt ein Grundgerüst, auf dem aufgebaut werden kann.
Übrigens, nach Schließen des Editors, sucht man das neu angelegte Skript *Dialog* in FixFoto vergebens. Dazu bitte den Aufgabenbereich mit F9 öffnen und im Kontextmenü zu *Skripte* die Automatische Skript-Konfiguration anklicken oder FixFoto beenden und neu starten, und siehe da – im Aufgabenbereich unter Skripte in der Kategorie *Demo* kann jetzt das Skript *Dialog* gestartet werden.

Wenn man das Dialogfeld weiter bearbeiten möchte, öffnet man das Skript im FFSkript-Editor wieder durch einen Doppelklick auf den Namen *Dialog* im linken Auswahlfenster. Wird jetzt der Dialogeditor aufgerufen, kann man das Dialogfeld beliebig verändern und ergänzen.

Steuerelemente

Im Dialogfeld können nun verschiedene Steuerelemente platziert werden. Um die prinzipielle Vorgehensweise zu zeigen, wird der oben erzeugte Dialog im Folgenden ergänzt um die Befehlsschaltfläche Ende. Im Dialogeditor klickt man über "*Steuerelement / Neu*" auf BUTTON und gibt ihm den Namen *Ende*. Nach OK wird er im Dialogfeld links oben platziert angezeigt und kann jetzt mit der Maus beliebig verschoben und auch in seiner Größe geändert werden. Dabei wird oben in einer Statuszeile "*Ende:*" angezeigt, gefolgt vom Abstand des Elements bis zum linken und zum oberen Rand und dann Breite x Höhe. Genau positionieren lässt sich das selektierte Element mit den Cursortasten.

Um sich einen Eindruck vom späteren Aussehen zu verschaffen, kann man "*Dialog / Dialog testen*" aufrufen oder einfach F5 drücken. Die Vorschau wird mit Anklicken von **x** rechts oben geschlossen.



Den Dialogeditor beendet man, indem über "*Dialog / Speichern*" die Ergänzung gesichert wird. Der Skriptcode wird automatisch ergänzt um

```
FF_AddControl "Dialogdemo","Ende","BUTTON",60,40,33,11
```

Auch hier kann sowohl der Abstand nach links und nach oben, als auch Breite mal Höhe des Elements frei editiert werden.

Jetzt muss noch per Code die Betätigung des Buttons *Ende* abgefragt und damit der Dialog beendet werden. Die Betätigung einer Befehlstaste stellt ein Ereignis dar. Alle Ereignisse von Steuerelementen sind Bestandteil des Befehls *FF_ShowDialog ("Dialogdemo")*. Dieser wird in einer Schleife auf eine Variable, z.B. *Taste* übertragen und damit gleichzeitig der Dialog dem Benutzer angezeigt. Wenn diese Variable den Inhalt "*Ende*" oder "*CANCEL*" (das ist das Ereignis, das durch Anklicken des **x** rechts oben im Dialogfeld mit der linken Maustaste bzw. durch Betätigung der Escape-*Taste* ausgelöst wird) hat, soll die Schleife verlassen, der Dialog mit *FF_CloseDialog ("Dialogdemo")*. geschlossen und die Meldung "*Dialog wurde beendet*" ausgegeben werden. Der Code sieht dann wie folgt aus:

```
'FFSubmenu=Demo
'FFName=Dialog

'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",115,60
FF_AddControl "Dialogdemo","Ende","BUTTON",60,40,33,11
'** End Dialog Dialogdemo **
```

```
do
        Taste = FF_ShowDialog("Dialogdemo")                 'Befehle abfragen und Dialog öffnen
        if Taste = "Ende" or Taste = "CANCEL" then exit do  'bei Ende oder Cancel Schleife verlassen
loop

FF_CloseDialog("Dialogdemo")                'Dialog schließen
msgbox "Dialog wurde beendet"               'Meldung
```

Statt ein Ereignis einer Variablen zuzuweisen, kann bei VB-Skripten eleganter die Case-Funktion verwendet werden. Dadurch wird der Code erheblich übersichtlicher, vor allem wenn viele Ereignisse abgefragt werden müssen. Der Code sieht dann wie folgt aus:

```
'FFSubmenu=Demo
'FFName=Dialog

'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",115,60
FF_AddControl "Dialogdemo","Ende","BUTTON",60,40,33,11
'** End Dialog Dialogdemo **

do
        Select Case FF_ShowDialog ("Dialogdemo")     'Dialog öffnen
        Case "Ende","CANCEL"                         'Ereignisse abfragen
                exit do                              'Reaktion auf die Ereignisse
        End Select
loop

FF_CloseDialog("Dialogdemo")                'Dialog schließen
msgbox "Dialog wurde beendet"               'Meldung
```

Die Schleife ist notwendig, damit der Dialog bei Verwendung weiterer Befehlsschaltflächen geöffnet bleibt. Dies wird später deutlich.

Zur Konstruktion der Bedienoberfläche stehen folgende Steuerelemente zur Verfügung:

GROUP	- optische Rahmen zur Zusammenfassung mehrer Elemente
BOOL	- Kontrollkästchen zur Auswahl einer Funktion
STATIC	- Bezeichnungsfeld, das nicht vom Benutzer geändert werden kann
BUTTON	- Befehlsschaltfläche
EDIT	- Textfeld mit Zeilenumbruch an der Feldgrenze
SLEDIT	- Textfeld ohne Zeilenumbruch
LISTBOX	- Auflistung von Elementen zur Auswahl
COMBO	- Klappliste von Elementen zur Auswahl
COLOR	- Farbauswahl
FONT	- Schriftauswahl
IMAGE	- Feld zur Anzeige eines Bildes
HSLIDER	- horizontaler Schieber zur Zahlenvorgabe von 0 bis 255
VSLIDER	- vertikaler Schieber zur Zahlenvorgabe von 0 bis 255 (ab Version 2.85)
VSPIN	- vertikale Pfeile zur Zahlenvorgabe in 1er Schritten
PROGRESS	- Fortschrittsanzeige (ab Version 2.90)

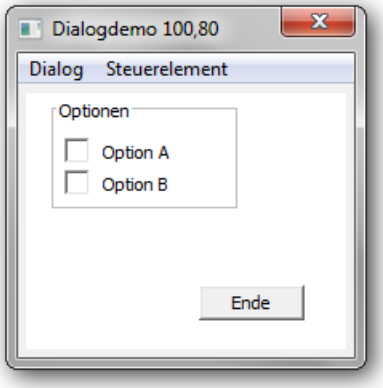
Diese werden in den folgenden Kapiteln im Einzelnen erläutert, allerdings nicht unbedingt in der gezeigten Reihenfolge. Da wo es sinnvoll ist, werden mehrere Steuerelemente in einem Kapitel zusammengefasst.

Und dann ist da noch die Sache mit den Stilen. Bei vielen Steuerelementen lassen sich im Dialogeditor für das Element, das den Fokus hat, über "*Steuerelement / Stile setzen*" diverse Eigenschaften festlegen. Sollen dies direkt mehrere sein, muss bei der Auswahl mit der linken Maustaste die Strg- bzw. die Ctrl-*Taste* festgehalten werden. Was die Eigenschaften bzw. Stile bedeuten, wird im Einzelfall bei den Erläuterungen der jeweiligen Steuerelemente und im Anhang angegeben. Ausführlicheres zu den Stilen ist im Kapitel Stile zu finden.

BOOL, STATIC und GROUP

Das Steuerelement BOOL stellt ein Kontrollkästchen dar. Wir öffnen den Dialogeditor und positionieren zwei dieser Elemente mit den Namen "Option1" und "Option2" in das Dialogfeld.
Danach schließen wir den Editor wieder durch Speichern. Wird jetzt das Skript ausgeführt, sieht man die beiden Kontrollkästchen, die durch Anklicken beliebig ein- und ausgeschaltet werden können.

Gut wären jetzt noch die jeweilige Benennung der Kontrollkästchen und eine optische Zusammenfassung. Die Benennung erfolgt mit zwei STATIC-Elementen, der wir die Namen "Option A" und "Option B" geben. Dann rahmen wir die BOOL- und STATIC-Elemente mit dem Element GROUP ein und geben diesem den Namen *Optionen*. Das sieht jetzt im Dialogeditor ungefähr so aus:



Wieder den Dialog speichern und nicht vergessen, auch immer wieder zwischendurch den Code speichern!

Natürlich muss auch die Aktivierung von Option A und B mal irgendwie ausgewertet werden. Das geschieht mit dem Befehl

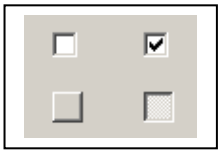
```
FF_GetControl (dialog, name)
```

Der Rückgabewert von BOOL ist im aktivierten Zustand 1 und im deaktivierten 0.
Im Beispiel soll nach Beendigung des Skripts mit dem BUTTON [Ende] angezeigt werden, was angekreuzt wurde:

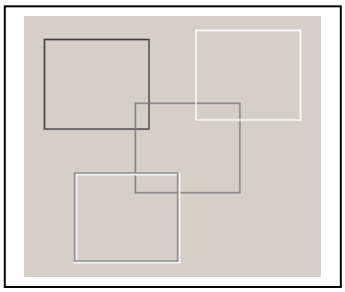
```
'FFSubmenu=Demo
'FFName=Dialog

Option Explicit        'erzwingt die Dimensionierung aller verwendeten Variablen und Konstanten
Dim msg                'Variablendimensionierung

'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",100,80
FF_AddControl "Dialogdemo","Optionen","GROUP",8,2,58,34
FF_AddControl "Dialogdemo","Ende","BUTTON",54,60,33,11
FF_AddControl "Dialogdemo","Option1","BOOL",12,14,8,8
FF_AddControl "Dialogdemo","Option2","BOOL",12,24,8,8
FF_AddControl "Dialogdemo","Option A","STATIC",24,13,33,11
FF_AddControl "Dialogdemo","Option B","STATIC",24,23,33,11
'** End Dialog Dialogdemo **

do
        Select Case FF_ShowDialog ("Dialogdemo")     'Dialog öffnen
        Case "Ende","CANCEL"                         'Ereignisse abfragen
                exit do                              'Reaktion auf die Ereignisse
        End Select
loop


if FF_GetControl ("Dialogdemo","Option1") = 1 then         'BOOL Option1 abfragen
        msg = "Option A wurde aktiviert" & vbNewline
else
        msg = "Option A wurde deaktiviert" & vbNewline
end if

if FF_GetControl ("Dialogdemo","Option2") = 1 then         'BOOL Option2 abfragen
        msg = msg & "Option B wurde aktiviert" & vbNewline
else
        msg = msg & "Option B wurde deaktiviert" & vbNewline
end if

FF_CloseDialog("Dialogdemo")                               'Dialog schließen
msgbox msg & vbNewline & "Dialog ist beendet"              'Ergebnis anzeigen
```

Wichtig ist, dass die Zustandsabfragen mit `FF_GetControl` vor Beendigung des Skripts erfolgen.

Das Aussehen der Steuerelemente BOOL und STATIC kann man beeinflussen kann durch Setzen von:



Stile bei BOOL
- BS_PUSHLIKE - aktiviert wird aus angekreuzt versenkt
- BS_NOTIFY - Anklicken als Ereignis erfassbar



Stile bei STATIC für einen Rahmen
- S_BLACKFRAME - schwarzer Rahmen ohne Text
- S_GRAYFRAME - grauer Rahmen ohne Text
- S_WHITEFRAME - weißer Rahmen ohne Text
- S_ETCHEDFRAME - versenkter Rahmen ohne Text

Stile bei STATIC zur Textausrichtung
- SS_CENTER - Text zentrisch angeordnet
- SS_RIGHT - Text rechts angeordnet

Stile bei STATIC zum Textabbruch mit drei Punkten
- SS_ENDELLIPSIS - zu langer Text wird abgebrochen mit ...

Die Wirkung der Stile wird erst in der Vorschau oder nach Skriptaufruf sichtbar.

BUTTON

Wie das Element BUTTON eingesetzt wird, wurde oben im Kapitel Steuerelemente bereits gezeigt. Hier soll noch mal näher erläutert werden, warum es sinnvoll ist, die Abfrage von Befehlsschaltflächen in einer Schleife vorzunehmen.
Die Aufgabe soll sein, mit einem BUTTON *Anzeigen* die Zustände der BOOL-Elemente aus dem Beispiel vor Beendigung anzuzeigen und dabei das Dialogfeld sichtbar zu lassen. Das Dialogfeld soll nur mit Ende bzw. *CANCEL* geschlossen und das Skript beendet werden. Dazu fügen wir über den Editor einen zusätzlichen BUTTON mit dem Namen Anzeigen hinzu. Nach dem Sichern ändern wir den Code wie folgt:

- Die Abfrage der BOOL-Elemente wird in eine SUB-Routine *Auswertung* überführt.
- In der Schleife wird bei Betätigung der Taste Anzeigen die Auswertung aufgerufen und diese angezeigt ohne die Schleife zu verlassen.
- Bei Betätigung von Ende oder *CANCEL* (**x** rechts oben anklicken oder Escape betätigen) wird die Schleife verlassen und das Dialogfeld wird geschlossen.

```
'FFSubmenu=Demo
'FFName=Dialog

Option Explicit
Dim msg

'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",100,80
FF_AddControl "Dialogdemo","Optionen","GROUP",7,3,58,34
FF_AddControl "Dialogdemo","Ende","BUTTON",54,58,33,11
FF_AddControl "Dialogdemo","Option1","BOOL",12,14,8,8
FF_AddControl "Dialogdemo","Option2","BOOL",12,24,8,8
FF_AddControl "Dialogdemo","Option A","STATIC",24,13,33,11
FF_AddControl "Dialogdemo","Option B","STATIC",24,23,33,11
FF_AddControl "Dialogdemo","Anzeigen","BUTTON",7,58,33,11
'** End Dialog Dialogdemo **
do
        Select Case FF_ShowDialog ("Dialogdemo")     'Dialog öffnen
                Case "Ende","CANCEL"                 'Skript beenden
                        exit do
                Case "Anzeigen"
                        call Auswertung
                        msgbox msg             'Ergebnis anzeigen
        End Select
loop
```

```
FF_CloseDialog ("Dialogdemo")                'Dialog schließen
msgbox "Dialog ist beendet"                  'Meldung

'-------------------------------------------------------------------------
Sub Auswertung
if FF_GetControl ("Dialogdemo","Option1") = 1 then         'BOOL Option1 abfragen
       msg = "Option A ist aktiviert" & vbNewline
else
       msg = "Option A ist deaktiviert" & vbNewline
end if

if FF_GetControl ("Dialogdemo","Option2") = 1 then         'BOOL Option2 abfragen
       msg = msg & "Option B ist aktiviert"
else
       msg = msg & "Option B ist deaktiviert"
end if
End Sub
'-------------------------------------------------------------------------
```

Wer einen BUTTON nicht nur mit einem Mausklick betätigen will, sondern auch mit der Tastatur, der kann sich dafür einen Buchstaben im Namen aussuchen, und diesem ein & voranstellen. In der Anzeige wird das &-Zeichen unterdrückt und der darauffolgende Buchstabe unterstrichen.

Beispiel:	Name	An&zeigen
	Anzeige	Anzeigen

Jetzt kann man den Button auch mit [Alt] + [Z] betätigen oder je nach Betriebssystem sogar nur mit [Z], wenn man vorher `Case "Anzeigen"` in `Case "An&zeigen"` geändert hat – am besten mal ausprobieren.
Aber Achtung: In einem Dialogfeld keinen Buchstaben mit vorangestelltem & doppelt verwenden.

Bei konkreten Skripten würde natürlich [Ende] umbenannt in beispielsweise [Ausführen] und damit das eigentliche Skript weiter ausgeführt. Dabei ist wichtig, dass im Skript Einstellungen des Benutzers im Dialogfeld erst in Variable abgespeichert werden, bevor das Dialogfeld geschlossen wird.

... und noch mal BOOL

Und jetzt werden noch mal BOOL-Elemente eingesetzt, aber diesmal mit Befehlseigenschaft.

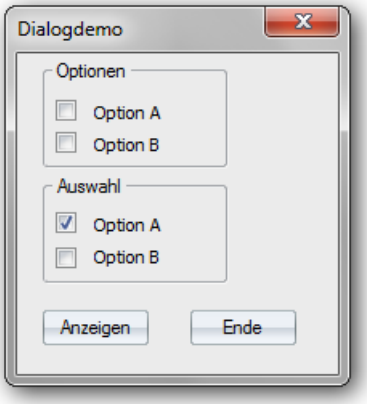
Es besteht folgende Aufgabe:
Das folgende Skript soll erweitert werden um eine zweite Gruppe *Auswahl* mit zwei Kontrollkästchen *Auswahl_ A* und *Auswahl_ B*, die wieder mit *Option A* und *Option B* über STATIC-Elemente benannt werden. Es soll immer nur eins der beiden Elemente auswählbar sein. Bei Start des Skripts soll *Auswahl_A* (bzw. *Option A*) bereits aktiviert sein.
Dazu wird erst einmal im Dialogeditor das Dialogfeld in der Höhe vergrößert auf beispielsweise 100 und die beiden Befehlsschaltflächen [Anzeigen] und [Ende] nach unten gerückt. Dann wird unterhalb der bestehenden Gruppe ein zweites GROUP-Element gleicher Größe mit dem Namen *Auswahl* aufgezogen.
In diesen Rahmen hinein wird ein BOOL-Element mit dem Namen *Auswahl_A* konstruiert. Damit bei diesem Element das Anklicken per Code erkannt werden kann, muss ihm der Stil BS_NOTIFY zugeordnet werden. Dies geschieht über "*Steuerelement / Stil setzen*" und anklicken von BS_NOTIFY,... solange das Element *Auswahl_A* den Fokus hat.
Jetzt soll das nächste Element *Auswahl_B* mit gleicher Eigenschaft erzeugt werden. Das kann auf dem gleichen Wege geschehen oder einfacher mit Klonen. Man gibt dem Element *Auswahl_A* den Fokus und klickt "*Steuerelement / Klonen*" an. Als Name wird *Auswahl_B* vergeben. Jetzt muss das Element nur noch positioniert werden.
Als nächstes sollen die Kontrollkästchen mit STATIC-Elementen mit *Option A* und *Option B* benannt werden. Bei dem Versuch das zu tun, erhält man die Meldung

"Dieser Name existiert bereits"

Das Problem kann gelöst werden, indem man an den Namen ein Pipe-Symbol "|" anhängt oder voran stellt, denn es gilt die Regel:

Enthaltene Pipe-Zeichen '|' in den Namen von BUTTON und STATIC werden bei der Anzeige unterdrückt. Dies ermöglicht die Mehrfachverwendung identischer Beschriftungen.

Das tatsächlich nur *Option A* angezeigt wird, lässt sich mit der Vorschau über F5 leicht kontrollieren.



Die zweite Beschriftung *Option B* wird genauso erzeugt und man erhält folgendes Dialogfeld:

Wenn man den Dialog speichert, sieht man im Code, dass den neuen Kontrollkästchen mit dem Befehl `FF_SetControlStyle` jeweils eine codierte Eigenschaft, nämlich der Dezimalwert des Stils BS_NOTIFY zugeordnet wurde.

Damit die Übersicht nicht verloren geht, wird empfohlen (kein muss), die Dialogbefehle so umzusortieren, dass sie die Reihenfolge ihres Auftretens wiederspiegeln, beispielsweise so:

```
'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",100,100
FF_AddControl "Dialogdemo","Optionen","GROUP",8,2,58,34
FF_AddControl "Dialogdemo","Option1","BOOL",12,14,8,8
FF_AddControl "Dialogdemo","Option A","STATIC",24,13,33,11
FF_AddControl "Dialogdemo","Option2","BOOL",12,24,8,8
FF_AddControl "Dialogdemo","Option B","STATIC",24,23,33,11
FF_AddControl "Dialogdemo","Auswahl","GROUP",8,38,58,34
FF_AddControl "Dialogdemo","Auswahl_A","BOOL",12,49,8,8
FF_SetControlStyle "Dialogdemo","Auswahl_A",16384
FF_AddControl "Dialogdemo","Option A|","STATIC",24,48,33,11
FF_AddControl "Dialogdemo","Auswahl_B","BOOL",12,60,8,8
FF_SetControlStyle "Dialogdemo","Auswahl_B",16384
FF_AddControl "Dialogdemo","Option B|","STATIC",24,58,33,11
FF_AddControl "Dialogdemo","Anzeigen","BUTTON",8,80,33,11
FF_AddControl "Dialogdemo","Ende","BUTTON",54,80,33,11
'** End Dialog Dialogdemo **
```

Um die Aufgabe zu erfüllen

- wird mit *`FF_SetControl "Dialogdemo","Auswahl_A",1`* die *Option A* vorbesetzt
- in der Schleife wird Auswahl_A und Auswahl_B wie ein Befehl abgefragt und die BOOL-Elemente entsprechend gesetzt
- in der SUB *Auswertung* wird der Meldetext ergänzt

Der gesamte Code sieht jetzt wie folgt aus:

```
'FFSubmenu=Demo
'FFName=Dialog

Option Explicit
Dim msg

'** Start Dialog Dialogdemo **
FF_AddDialog "Dialogdemo",100,100
FF_AddControl "Dialogdemo","Optionen","GROUP",8,2,58,34
FF_AddControl "Dialogdemo","Option1","BOOL",12,14,8,8
FF_AddControl "Dialogdemo","Option A","STATIC",24,13,33,11
FF_AddControl "Dialogdemo","Option2","BOOL",12,24,8,8
FF_AddControl "Dialogdemo","Option B","STATIC",24,23,33,11
FF_AddControl "Dialogdemo","Auswahl","GROUP",8,38,58,34
FF_AddControl "Dialogdemo","Auswahl_A","BOOL",12,49,8,8
FF_SetControlStyle "Dialogdemo","Auswahl_A",16384
FF_AddControl "Dialogdemo","Option A|","STATIC",24,48,33,11
FF_AddControl "Dialogdemo","Auswahl_B","BOOL",12,60,8,8
```

```
FF_SetControlStyle "Dialogdemo","Auswahl_B",16384
FF_AddControl "Dialogdemo","Option B|","STATIC",24,58,33,11
FF_AddControl "Dialogdemo","Anzeigen","BUTTON",8,80,33,11
FF_AddControl "Dialogdemo","Ende","BUTTON",54,80,33,11
'** End Dialog Dialogdemo **

FF_SetControl "Dialogdemo","Auswahl_A",1                    'Auswahl_A mit 1 vorbesetzen

do
        Select Case FF_ShowDialog ("Dialogdemo")            'Dialog öffnen
        Case "Auswahl_A"
                FF_SetControl "Dialogdemo","Auswahl_A",1
                FF_SetControl "Dialogdemo","Auswahl_B",0
        Case "Auswahl_B"
                FF_SetControl "Dialogdemo","Auswahl_A",0
                FF_SetControl "Dialogdemo","Auswahl_B",1
        Case "Ende","CANCEL"                                'Skript beenden
                exit do
        Case "Anzeigen"
                call Auswertung
                msgbox msg                          'Ergebnis anzeigen
        Case Else
        End Select
loop

FF_CloseDialog ("Dialogdemo")                       'Dialog schließen
msgbox "Dialog ist beendet"                         'Meldung
'-----------------------------------------------------------------
Sub Auswertung

        msg = "Optionen" & ":" & vbNewLine

        if FF_GetControl ("Dialogdemo","Option1") = 1 then
                msg = msg & "Option A ist aktiviert" & vbNewline
        else
                msg = msg & "Option A ist deaktiviert" & vbNewline
        end if

        if FF_GetControl ("Dialogdemo","Option2") = 1 then
                msg = msg & "Option B ist aktiviert"
        else
                msg = msg & "Option B ist deaktiviert"
        end if

        msg = msg & vbNewLine & vbNewLine & "Auswahl" & ":" & vbNewLine

        if FF_GetControl ("Dialogdemo","Auswahl_A") = 1 then
                msg = msg & "Option A wurde ausgewählt"
        else
                msg = msg & "Option B wurde ausgewählt"
        end if

End Sub
'-----------------------------------------------------------------
```

Zwischenbilanz

Mit dem bisher Gelernten lässt sich bereits eine einfache Bedienoberfläche zur Voreinstellung eines Skripts ansprechend gestalten. Hier werden noch mal alle wichtigen Punkte aufgelistet:

- Unbedingt immer beim Verlassen des Dialogeditors speichern und auch im FFSkript-Editor immer wieder mal zwischenspeichern.
- Im Dialogeditor immer die Position und Größe des Steuerelements in der Statuszeile oben im Auge behalten. Ein Steuerelement, das den Fokus hat, kann genau mit den Cursortasten positioniert werden.
- Bei der grafischen Konstruktion eines Dialogfeldes kann mit F5 eine Vorschau aufgerufen werden. Ausgeschaltet wird über das **x** oben rechts.
- Befehle im Code sollten innerhalb einer Schleife (`do-loop`) abgefragt werden um die Anzeige des Dialogfeldes erhalten zu können.

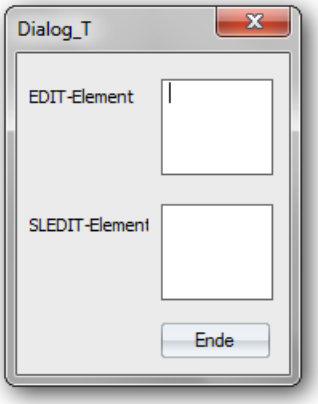
- Ein Dialogfeld beinhaltet immer den Befehl *CANCEL*. Er wird ausgelöst durch Anklicken des **x** oben rechts und durch Betätigung der Escape-Taste und sollte immer für den vorzeitigen Abbruch eines Skripts ausgewertet werden.
- Eigenschaften von Steuerelementen werden über "*Stile setzen*" festgelegt. Dabei muss das Element den Fokus haben. Die Zuordnung mehrerer Stile zu einem Element ist durch Festhalten der Strg bzw. der Ctrl -Taste während des Setzens möglich.
- Mehrere Steuerelemente gleicher Art, gleicher Größe und mit gleichen Eigenschaften (Stilen) können durch Klonen erzeugt werden.
- Um BOOL-Elemente wie Befehlsschaltflächen behandeln zu können, muss der Stil BS_NOTIFY bei diesem Element gesetzt sein.
- Zur Verwendung gleicher Beschriftungen bei BUTTON- und STATIC-Elementen können in den Namen Pipe-Symbole "|" eingesetzt werden. Bei der Anzeige werden diese unterdrückt.
- BUTTON können auch über die Tatstatur mit Alt + Buchstabe oder nur Buchstabe betätigt werden, wenn im Namen des BUTTON dem Buchstaben ein &-Zeichen vorangestellt und er dadurch in der Anzeige unterstrichen wurde. In der Abfrage muss aber unbedingt auch das &-Zeichen stehen.
- Vor Schließen des Dialogfeldes mit FF_CloseDialog die vorgenommenen Einstellungen in Variablen sichern. Nach Beenden des Dialogs stehen diese nicht mehr zur Verfügung.

EDIT und SLEDIT

Beide Steuerelemente dienen zur Eingabe und Anzeige von Text. Um den Unterschied zu verdeutlichen, legen wir ein neues Dialogfeld Dialog_T von der Größe 85 x 100 an und konstruieren dort hinein

- ein Steuerelement EDIT mit dem Namen TxtEdit und bezeichnen es über ein STATIC-Element mit "EDIT-Element"
- ein Steuerelement SLEDIT mit dem Namen TxtSlEdit und bezeichnen es über ein STATIC-Element mit "SLEDIT-Element"
- Beide Elemente sollten mindestens eine Höhe von 30 haben, damit die unterschiedlichen Eigenschaften demonstriert werden können
- Darunter wird noch ein BUTTON Ende angelegt.

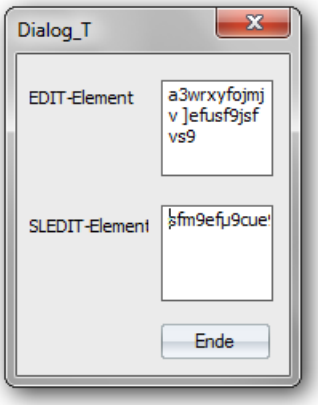
Die Vorschau sollte jetzt ungefähr so aussehen:



Der Code dazu lautet:

```
'FFSubmenu=Demo
'FFName=Dialog_T

'** Start Dialog Dialog_T **
FF_AddDialog "Dialog_T",85,100
FF_AddControl "Dialog_T","EDIT-Element","STATIC",4,8,33,11
FF_AddControl "Dialog_T","TxtEdit","EDIT",45,8,35,30
FF_AddControl "Dialog_T","SLEDIT-Element","STATIC",4,48,37,11
FF_AddControl "Dialog_T","TxtSlEdit","SLEDIT",45,47,35,30
FF_AddControl "Dialog_T","Ende","BUTTON",45,84,34,11
'** End Dialog Dialog_T **

do
        Select Case FF_ShowDialog ("Dialog_T")       'Befehle abfragen und Dialog öffnen
        Case "Ende","CANCEL"                         'bei Ende Schleife verlassen
                exit do
        Case Else
        End Select
loop

FF_CloseDialog ("Dialog_T")                  'Dialog schließen
msgbox "Dialog ist beendet"                  'Meldung
```

Wenn man das Skript ausführt und gibt wahllos Text in die beiden Elemente EDIT und SLEDIT ein, sieht man, dass bei EDIT der Text an der Elementengrenze automatisch umgebrochen und bei SLEDIT der Text einfach fortgeführt wird, auch in den nicht sichtbaren Bereich hinein. Das kann man sich gut merken, wenn man weiß, dass SLEDIT **S**ingle-**L**ine-**Edit** heißt.

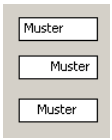


Bei den Textfeldern steht dem Benutzer übrigens das bekannte Kontextmenü zum *Kopieren*, *Einfügen* ect. zur Verfügung. Das genaue Aussehen hängt vom Betriebssystem ab

Ausgelesen wird eingegebener Text mit `FF_GetControl`. Zur Anzeige von Text wird `FF_SetControl` benutzt.

Interessant sind die Stile, die bei beiden Textelementen gleich sind.
Die Textausrichtung wird vorgegeben mit

- ES_LEFT - Textausrichtung links
- ES_RIGHT - Textausrichtung rechts
- ES_CENTER - Textausrichtung Mitte



Beschränkung auf Groß- und Kleinbuchstaben wird festgelegt mit

- ES_LOWER - nur Kleinbuchstaben
- ES_UPPER - nur Großbuchstaben

Weitere zugelassene Stile sind

- ES_NOHIDESEL - bedeutet, dass die Hervorhebung der Selektion nicht weggenommen wird, wenn das Steuerelement den Fokus verliert.
- ES_NUMBER - erlaubt nur die Eingabe von Zahlen. Bei Eingabe anderer Zeichen wird darauf hingewiesen (siehe Screenshot oberes Feld).
- ES_PASSWORD - eingegebene Zeichen werden als Punkte dargestellt (s. Screenshot unteres Feld). Natürlich kann per Code mit `FF_GetControl` die Originaleingabe ermittelt werden (nur bei SLEDIT)
- ES_READONLY - Text per Code zur Anzeige vorzugeben ist mit `FF_SetControl` möglich. Eingaben durch den Benutzer sind gesperrt.

Und es gibt Stile, die sich nicht über den Dialogeditor setzen lassen, sondern nur direkt über Code (Näheres dazu im Kapitel **Stile**):

- WS_VSCROLL - vertikale Scrollbar
- WS_HSCROLL - horizontale Scrollbar
- WS_THICKFRAME - erzeugt ein Fenster mit in seiner Größe veränderbarem Doppelrahmen

LISTBOX und COMBO

Beide dienen der Auswahl eines Elementes aus einer vorgegebenen Auflistung. Um dies zu demonstrieren legen wir ein Feld Dialog_LC mit folgendem Inhalt an:

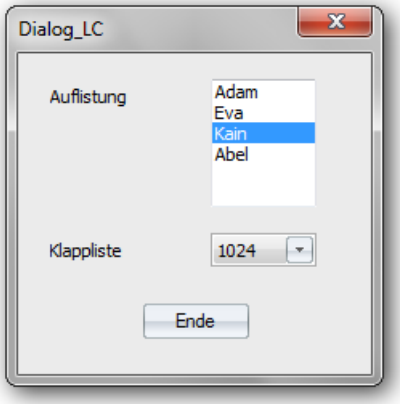
- Oben rechts eine LISTBOX mit dem Namen *Liste_L*, benannt mit *Auflistung* per STATIC-Element

- Darunter eine COMBO-Liste mit dem Namen *Liste_C*, benannt mit *Klappliste* per STATIC-Element
- Darunter einen BUTTON [Ende]

Im dazugehörigen Code müssen die beiden Listen mit den auszuwählenden Begriffen gefüllt werden. Dies geschieht mit dem Befehl `FF_SetControl`.
Hier wurden für die LISTBOX als Beispiel die Begriffe *Adam, Eva, Kain, Abel* und für das COMBO-Element die Zahlen *256, 512, 1024, 2048* und *4096* vorgegeben.
Bei beiden Elementen kann ein Begriff favorisiert werden, indem man ihn mit `FF_SetControl` ein zweites Mal vorgibt. Dies soll hier bei der LISTBOX mit *Kain* und beim COMBO-Element mit *1024* gemacht werden.

Bei der LISTBOX wird mit einem Doppelklick auf eins der gelisteten Elemente ein Ereignis ausgelöst mit dem das angeklickte Element direkt abgefragt werden kann. Bei der COMBO-Liste geschieht dies mit der linken Maustaste durch Auswahl eines Elements, das gerade nicht den Fokus hat. Mit [Ende] oder dem CANCEL-Ereignis wird das Skript beendet und es werden die vom Benutzer gewählten Begriffe angezeigt.

Der Code sieht wie folgt aus:




```
'FFSubmenu=Demo
'FFName=Dialog_LC

Option Explicit
Dim Listenauswahl, Klapplistenwahl

'** Start Dialog Dialog_LC **
FF_AddDialog "Dialog_LC",110,100
FF_AddControl "Dialog_LC","Auflistung","STATIC",10,8,33,11
FF_AddControl "Dialog_LC","Liste_L","LISTBOX",60,8,33,44
FF_AddControl "Dialog_LC","Klappliste","STATIC",10,56,33,11
FF_AddControl "Dialog_LC","Liste_C","COMBO",60,56,33,10
FF_AddControl "Dialog_LC","Ende","BUTTON",39,78,33,11
'** End Dialog Dialog_LC **

'Auflistung füllen
FF_SetControl "Dialog_LC","Liste_L","Adam"
FF_SetControl "Dialog_LC","Liste_L","Eva"
FF_SetControl "Dialog_LC","Liste_L","Kain"
FF_SetControl "Dialog_LC","Liste_L","Abel"
FF_SetControl "Dialog_LC","Liste_L","Kain"   'Vorgabe machen

'Klappliste füllen
FF_SetControl "Dialog_LC","Liste_C","256"
FF_SetControl "Dialog_LC","Liste_C","512"
FF_SetControl "Dialog_LC","Liste_C","1024"
FF_SetControl "Dialog_LC","Liste_C","2048"
FF_SetControl "Dialog_LC","Liste_C","4096"
FF_SetControl "Dialog_LC","Liste_C","1024"   'Vorgabe machen


do
        Select Case FF_ShowDialog ("Dialog_LC")      'Befehle abfragen und Dialog öffnen
        Case "Liste_L"
                msgbox "aus der Liste wurde durch" & vbNewLine & _
                "Doppelklick '" & FF_GetControl ("Dialog_LC","Liste_L") & "' ausgewählt"
        Case "Liste_C"
                msgbox "aus der Klappliste wurde durch" & vbNewLine & _
                "einfachen Klick '" & FF_GetControl ("Dialog_LC","Liste_C") & "' ausgewählt"
        Case "Ende","CANCEL"
                exit do
        Case Else
        End Select
loop

'Auswahlwerte sichern
Listenauswahl = FF_GetControl ("Dialog_LC","Liste_L")
Klapplistenwahl = FF_GetControl ("Dialog_LC","Liste_C")

FF_CloseDialog ("Dialog_LC")                 'Dialog schließen
```

```
msgbox "Gewählt wurde" & vbNewLine & _
          "aus der Auflistung - " & Listenauswahl & vbNewLine & _
          "aus der Klappliste  - " & Klapplistenwahl & vbNewLine & vbNewLine & _
          "Dialog ist beendet",vbInformation,"Meldung"
```

Der Inhalt der LISTBOX und des COMBO-Elements kann jeweils mit einem Nullstring ("") als Wert im Befehl `FF_SetControl (dialog, name, wert)` gelöscht werden:

```
FF_SetControl "Dialog_LC","Liste_L",""              'Löschen der Auflistung
FF_SetControl "Dialog_LC","Liste_C",""              'Löschen der Klappliste
```

FONT und COLOR

Mit FONT kann eine Schrift, mit COLOR eine Farbe ausgewählt werden.
Im neuen Dialog_FC legt man ein FONT- und ein COLOR-Steuerelement an, benannt mit *Schriftwahl* und *Farbwahl*. Mit STATIC-Elementen bezeichnet man sie entsprechend mit "Schriften:" und "Farben:".

Der Code dazu lautet:

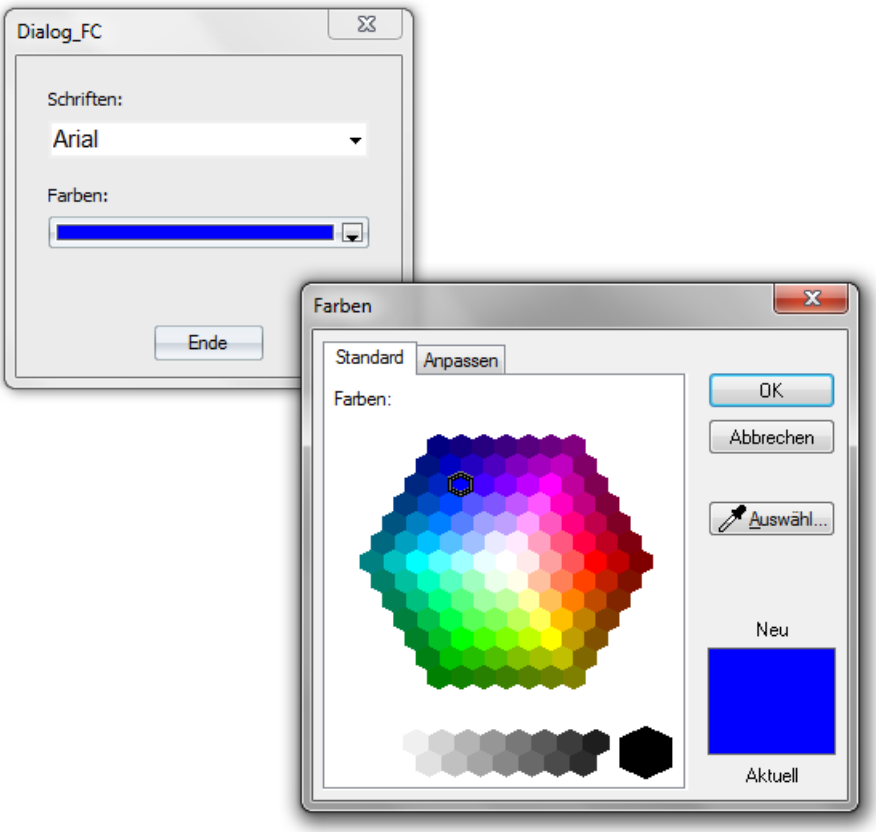
```
'FFSubmenu=Demo
'FFName=Dialog_FC

Option Explicit
Dim Schrift, Farbe

'** Start Dialog Dialog_FC **
FF_AddDialog "Dialog_FC",120,100
FF_AddControl "Dialog_FC","Schriften:","STATIC",10,8,33,11
FF_AddControl "Dialog_FC","Schriftwahl","FONT",10,20,100,10
FF_AddControl "Dialog_FC","Farben:","STATIC",10,38,33,11
FF_AddControl "Dialog_FC","Farbwahl","COLOR",10,50,100,10
FF_AddControl "Dialog_FC","Ende","BUTTON",43,84,34,11
'** End Dialog Dialog_FC **

do
       Select Case FF_ShowDialog ("Dialog_FC")      'Befehle abfragen und Dialog öffnen
       Case "Ende","CANCEL"                         'bei Ende Schleife verlassen
              exit do
       Case Else
       End Select
loop

'Schrift- und Farbwert sichern
Schrift = FF_GetControl ("Dialog_FC","Schriftwahl")
Farbe = FF_GetControl ("Dialog_FC","Farbwahl")

FF_CloseDialog ("Dialog_FC")                 'Dialog schließen

msgbox "Gewählt wurde" & vbNewLine & _
          "Schrift = " & Schrift & vbNewLine & _
          "Farbe = " & Farbe & vbNewLine & vbNewLine & _
          "Dialog ist beendet", vbInformation, "Meldung"
```

Bei Ausführen des Skripts werden in der Schriftenauswahl alle auf dem jeweiligen Rechner installierten Fonts angezeigt. Der Name der angeklickten Schrift wird bei `FF_GetControl` im Klartext zurück gegeben.
Bei den Farben ist eine sehr differenzierte Auswahl bis zur Ermittlung einer bestehenden Farbe mit der Pipette möglich. Die Farbe wird als RGB-Wert ausgegeben (z.B. "195 250 244").

Sowohl bei der Schriftenauswahl als auch bei der Farbauswahl wird bei Auswahl eines Elements mit der linken Maustaste, das gerade nicht den Fokus hat, ein abfragbares Ereignis ausgelöst; beim COLOR-Steuerelement allerdings erst ab FixFoto-Version 3.30.
Die `do...loop` - Schleife könnte beispielsweise wie folgt ergänzt werden:

```
       Case "Schriftwahl"
              msgbox "die Schrift wurde geändert"
       Case "Farbwahl"
              msgbox "die Farbauswahl wurde geändert"
```

Das Ergebnis sieht so aus:



Wird der RGB-Wert in diversen Skriptbefehlen, z.B.

```
FF_Line VT_I4 xstart, VT_I4 ystart, VT_I4 xend, VT_I4 yend, VT_I4 colorref, VT_I4 width
```

als Ganzzahl *colorref* benötigt, kann dieser Wert daraus berechnet werden:

```
FarbWert = split(Farbe," ")                 'Berechnung von colorref
Rot = FarbWert (0) * 1
Gruen = FarbWert (1) * 1
Blau = FarbWert (2) * 1
colorref = Rot + Gruen * (2^8) + Blau * (2^16)
```

Der Wert colorref kann bei Bedarf auch auf RGB-Werte zurück gerechnet werden:

```
Rot = colorref And &HFF&                    'Rückwandlung in RGB-Wert
Gruen = (colorref And &HFF00&)\256
Blau = (colorref And &HFF0000&)\65536
Farbe = CStr(Rot) & " " & CStr(Gruen) & " " & CStr(Blau)
```

Vordefiniert mit einer bestimmten Farbe wird das Steuerelement COLOR mit dem RGB-Wert, z.B.

```
FF_SetControl "Dialog_FC","Farbwahl","195 250 244"
```

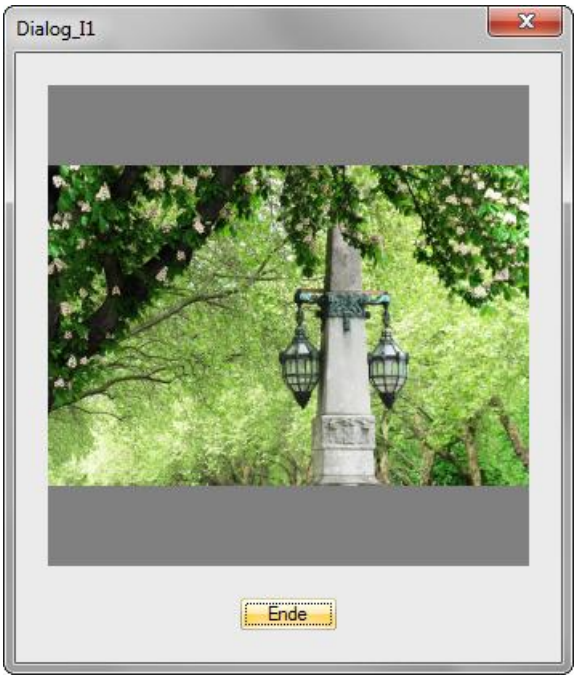
IMAGE

Mit IMAGE können Bilder angezeigt werden.

Im neuen *Dialog_I1* erstellt man ein Steuerelement IMAGE von der Größe 150 x 150 mit Namen *Bild*. Es ist quadratisch, um hoch- und querformatige Bilder gleich gut anzeigen zu können.
Darunter wird noch ein BUTTON [Ende] gesetzt.

Mit `FF_SetControl "Dialog_I1","Bild",wert` kann jetzt ein Bild in das Steuerelement IMAGE übertragen werden, wobei *wert* ein Pfad auf ein Bild oder der Index des Stacks sein kann. Dabei entspricht 0 dem aktuellen Bild, was im folgenden Programmcode genutzt wird.

```
'FFSubmenu=Demo
'FFName=Dialog_I1

'** Start Dialog Dialog_I1 **
FF_AddDialog "Dialog_I1",170,190
FF_AddControl "Dialog_I1","Bild","IMAGE",10,10,150,150
FF_AddControl "Dialog_I1","Ende","BUTTON",70,170,30,10
'** End Dialog Dialog_I1 **

call main
'-------------------------------------------------------
Sub main

if FF_GetImageWidth = 0 then
        msgbox "es wurde kein Bild geladen"
        exit sub
end if

FF_SetControl "Dialog_I1","Bild",0
        'mit Wert 0 wird aktuelles Bild geladen

do
        Select Case FF_ShowDialog ("Dialog_I1")
        Case "Ende","CANCEL"
                exit do
        Case Else
        End Select
loop

FF_CloseDialog ("Dialog_I1")                'Dialog schließen
msgbox "Dialog ist beendet"                 'Meldung

End Sub
'-------------------------------------------------------------------------------
```



Vor der Übertragung wird geprüft, ob überhaupt ein Bild geladen ist. Dies geschieht durch Abfrage der Bildbreite (alternativ der Bildhöhe). Ist diese 0 steht logischerweise kein aktuelles Bild zur Verfügung und das Skript wird mit einer Meldung abgebrochen. Der Abbruch kann direkt mit `FF_ErrorBreak "Abbruch"` vorgenommen werden. Eleganter ist aber den gesamten Ablauf in eine Unterroutine zu packen und im Abbruchfall diese mit `exit sub` zu verlassen.

Oft will man mehrere Bilder in der Computeransicht markieren und diese sich mit einer Blätterfunktion nacheinander ansehen. Dazu wird ein Dialog_I2 wie der obige Dialog_1 angelegt, nur unten ergänzt um die zwei BUTTON [<<<] und [>>>] zum Rückwärts- und Vorwärtsblättern.

```
'FFSubmenu=Demo
'FFName=Dialog_I2

Option Explicit
Dim BldAnzahl, BldNummer                        'Bildanzahl und -nummer

'** Start Dialog Dialog_I2 **
FF_AddDialog "Dialog_I2",170,190
FF_AddControl "Dialog_I2","Bild","IMAGE",10,10,150,150
FF_AddControl "Dialog_I2","Ende","BUTTON",130,170,30,10
FF_AddControl "Dialog_I2","<<<","BUTTON",25,170,30,10
FF_AddControl "Dialog_I2",">>>","BUTTON",65,170,30,10
'** End Dialog Dialog_I2 **

call main
```

```
'-------------------------------------------------------------------------------
Sub main

BldAnzahl = FF_GetImageCount

if BldAnzahl = 0 then
        msgbox "Es wurde kein Bild in der Computeransicht selektiert"
        exit sub
end if

'erstes (0) selektiertes Bild laden
BldNummer = 0
FF_LoadImage FF_GetImageName(BldNummer)
FF_SetControl "Dialog_I2","Bild",0

'Freigaben bzw. Sperren der Tasten <<< und >>>
FF_EnableControl "Dialog_I2","<<<",false
if BldAnzahl = 1 then FF_EnableControl "Dialog_I2",">>>",false

'Befehlskontrolle
do
        Select Case FF_ShowDialog ("Dialog_I2")             'Befehle abfragen und Dialog öffnen
        Case ">>>"
                BldNummer = BldNummer+1
                FF_LoadImage FF_GetImageName(BldNummer)     'Bild laden
                FF_SetControl "Dialog_I2","Bild",0
                'Freigaben bzw. Sperren der Tasten <<< und >>>
                FF_EnableControl "Dialog_I2","<<<",true
                if BldNummer = BldAnzahl-1 then FF_EnableControl "Dialog_I2",">>>",false
        Case "<<<"
                BldNummer = BldNummer-1
                FF_LoadImage FF_GetImageName(BldNummer)     'Bild laden
                FF_SetControl "Dialog_I2","Bild",0
                'Freigaben bzw. Sperren der Tasten <<< und >>>
                FF_EnableControl "Dialog_I2",">>>",true
                if BldNummer = 0 then FF_EnableControl "Dialog_I2","<<<",false
        Case "CANCEL"                                       'bei Cancel Abbruch des Skript
                FF_CloseDialog ("Dialog_I2")
                exit sub
        Case "Ende"                                         'bei Ende Schleife verlassen
                exit do
        Case Else
        End Select
loop

FF_CloseDialog ("Dialog_I2")  'Dialog schließen
msgbox "Dialog ist beendet"   'Meldung

End Sub
'-------------------------------------------------
```

Hier wird erst einmal mit `FF_GetImageCount` geprüft, ob in der Computeransicht mindestens ein Bild markiert ist. Dann wird das erste Bild mit der Nummer 0 geladen und zum Steuerelement IMAGE übertragen.

Um zu verhindern, dass mit den Tasten **<<<** und **>>>** unzulässige Werte erreicht werden können, werden diese mit `FF_EnableControl` frei gegeben oder gesperrt. Für den Benutzer sind diese Zustände sichtbar und machen damit deutlich, ob das Vor- bzw. Zurückblättern sinnvoll ist.

Anmerkung:
Beim Einfachklick auf das Bild mit der linken Maustaste wird ein Befehlsereignis ausgelöst, das im obigen Beispiel in der `do-loop` -Schleife mit `Case "Bild"` abgefragt werden kann.

HSLIDER, VSLIDER und VSPIN

Die drei Elemente dienen der Zahlenvorgabe durch den Benutzer. Dabei steht VSLIDER erst ab FixFoto-Version 2.85 zur Verfügung.
HSLIDER und VSLIDER sind Schieberegler mit denen man Zahlen zwischen 0 und 255 einstellen kann und VSPIN besteht aus zwei Pfeilen. Mit dem oberen kann eine Zahl schrittweise erhöht und mit dem unteren schrittweise verringert werden.

Die Elemente machen nur Sinn, wenn ihnen jeweils ein Textelement zur Werteanzeige zugeordnet wird, am besten SLEDIT mit den gesetzten Stilen ES_RIGHT und ES_READONLY. Dies wird im folgenden Beispiel deutlich:

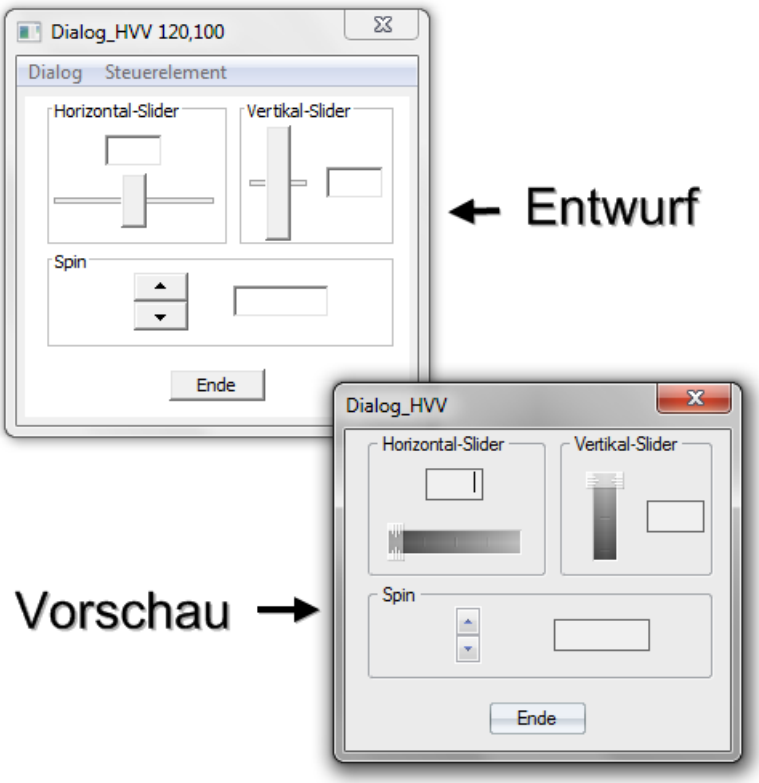
In ein neues Dialogfeld Dialog_HVV werden gesetzt:
- ein GROUP-Element *Horizontal-Slider*
- darin ein SLEDIT-Element *AnzeigeHS* mit oben aufgeführten Eigenschaften und
- ein HSLIDER, bezeichnet mit *HorizontalSlider*

- ein GROUP-Element *Vertikal-Slider*
- ein VSLIDER, bezeichnet mit *VertikalSlider*
- darin ein SLEDIT-Element *AnzeigeVS* mit oben aufgeführten Eigenschaften

- ein GROUP-Element *Spin*
- ein VSPIN, bezeichnet mit *Vertikal*
- darin ein SLEDIT-Element *AnzeigeV* mit oben aufgeführten Eigenschaften

- ein BUTTON-Element Ende

Im Dialogeditor sieht das wie folgt aus:



Im Programmcode muss für die Steuerelemente erst mit `FF_SetControlBuddy` ein Bezug zu den zugehörigen Textelementen hergestellt werden. Mit `FF_SetControl` ist jeweils eine Voreinstellung möglich.

Beim Element HSLIDER
die Voreinstellung: `FF_SetControl "Dialog_HVV","HorizontalSlider",40 'z.B. 40`
die Zuordnung: `FF_SetControlBuddy "Dialog_HVV","HorizontalSlider","AnzeigeHS"`

Beim Element VSLIDER
die Voreinstellung: `FF_SetControl "Dialog_HVV"," VertikalSlider",60 'z.B. 60`
die Zuordnung: `FF_SetControlBuddy "Dialog_HVV"," VertikalSlider","AnzeigeVS"`

Beim Element VSPIN:
die Voreinstellung:
```
WertV = 2      'über eine Variable mit Wert 2 als Beispiel
FF_SetControl "Dialog_HVV","AnzeigeV",WertV
```
die Zuordnung in der Befehlsschleife:

Mit Anklicken eines der Pfeile wird ein Tastenbefehl ausgelöst, der bei Abfrage mit `FF_GetControl` einen Stringwert zurückgibt und zwar beim oberen Pfeil "1" und beim unteren Pfeil "-1". Dieser String gewandelt in eine Zahl, wird in der Variablen *WertV* aufaddiert. Der *WertV* wird dann mit `FF_SetControl` auf das Textelement *AnzeigeV* übertragen.

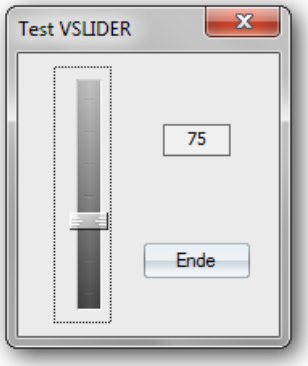
```
'FFSubmenu=Demo
'FFName=Dialog_HVV

Option Explicit
Dim WertV, wertHSL, wertVSL, wertVSP

call main
'-------------------------------------------------------------------------------
Sub main

'** Start Dialog Dialog_HVV **
FF_AddDialog "Dialog_HVV",120,100
FF_AddControl "Dialog_HVV","Horizontal-Slider","GROUP",7,1,56,45
FF_AddControl "Dialog_HVV","AnzeigeHS","SLEDIT",25,12,18,10
FF_SetControlStyle "Dialog_HVV","AnzeigeHS",2050
FF_AddControl "Dialog_HVV","HorizontalSlider","HSLIDER",9,24,50,17
FF_AddControl "Dialog_HVV","Vertikal-Slider","GROUP",67,1,48,45
FF_AddControl "Dialog_HVV","VertikalSlider","VSLIDER",70,9,18,36
FF_AddControl "Dialog_HVV","AnzeigeVS","SLEDIT",94,22,18,10
FF_SetControlStyle "Dialog_HVV","AnzeigeVS",2050
FF_AddControl "Dialog_HVV","Spin","GROUP",7,48,108,30
FF_AddControl "Dialog_HVV","Vertikal","VSPIN",34,55,17,18
FF_AddControl "Dialog_HVV","AnzeigeV","SLEDIT",65,59,30,10
FF_SetControlStyle "Dialog_HVV","AnzeigeV",2050
FF_AddControl "Dialog_HVV","Ende","BUTTON",45,85,30,10
'** End Dialog Dialog_HVV **

FF_SetControl "Dialog_HVV","HorizontalSlider",40                  'Vorbelegung des HSLIDER
FF_SetControlBuddy "Dialog_HVV","HorizontalSlider","AnzeigeHS"    'Zuordnung zu AnzeigeHS

FF_SetControl "Dialog_HVV","VertikalSlider",60                    'Vorbelegung des VSLIDER
FF_SetControlBuddy "Dialog_HVV","VertikalSlider","AnzeigeVS"      'Zuordnung zu AnzeigeVS

WertV = 2                                                 'Vorbelegung des WertV
FF_SetControl "Dialog_HVV","AnzeigeV",WertV               'Anzeige des WertV

do
        Select Case FF_ShowDialog ("Dialog_HVV")          'Befehle abfragen und Dialog öffnen
        Case "Vertikal"
                WertV = WertV + CInt(FF_GetControl ("Dialog_HVV","Vertikal"))
                FF_SetControl "Dialog_HVV","AnzeigeV",WertV
        Case "CANCEL"                                     'bei Cancel Abbruch des Skript
                FF_CloseDialog ("Dialog_HVV")
                exit sub
        Case "Ende"                                       'bei Ende Schleife verlassen
                exit do
        Case Else
        End Select
loop
```

```
wertHSL = FF_GetControl ("Dialog_HVV","AnzeigeHS")          'Dialogwerte speichern
wertVSL = FF_GetControl ("Dialog_HVV","AnzeigeVS")
wertVSP = FF_GetControl ("Dialog_HVV","AnzeigeV")

FF_CloseDialog ("Dialog_HVV")                               'Dialog schließen

msgbox "Eingestellt wurde" & vbNewLine & _
          "SliderH = " & wertHSL & vbNewLine & _
          "SliderV = " & wertVSL & vbNewLine & _
          "Spin    = " & wertVSP & vbNewLine & vbNewLine & _
          "Dialog ist beendet", vbOk, "Meldung"

End Sub
'-------------------------------------------------------------------------------
```

Die Einstellwerte können natürlich auf mathematischem Wege manipuliert werden:

Beim VSPIN-Element bietet sich an, direkt beim Aufaddieren im *WertV* den Rückgabewert mit einem Faktor zu versehen, z.B. 0,25 (im Code Punkt statt Komma, also 0.25 einsetzen).

Wenn man das Element HSLIDER oder VSLIDER beispielsweise zur Einstellung eines JPEG-Faktors von 60 bis 100 verwenden will, muss das Ergebnis mit (40/255 + 60) multipliziert und in einen Integerwert umgewandelt werden. Leider gibt es bis FF-Version 2.84 keine Möglichkeit, dies während des Verstellens anzuzeigen und ist daher nicht sinnvoll anzuwenden. Es gibt nur die Möglichkeit, selbst eine Skala aus Zahlen zu konstruieren (sehr mühsam!).
Ab FF-Version 2.85 gibt es den Befehl `FF_SetDialogTimer`. Mit ihm kann über eine in ms einzugebende Zeit der Sliderwert abgefragt, manipuliert und angezeigt werden. Baut man dies in die übliche Befehlsschleife ein, wird die Schleife nicht durch `FF_ShowDialog` angehalten. Hier ein Testbeispiel mit dem VSLIDER:




```
'FFSubmenu=Test
'FFName=Test VSlider

Option Explicit

const Min = 60
const Max = 100
const Start = 75

Dim Wert

'** Start Dialog Test VSLIDER **
FF_AddDialog "Test VSLIDER",81,88
FF_AddControl "Test VSLIDER","VSL","VSLIDER",11,4,18,80
FF_AddControl "Test VSLIDER","AnzVSL","SLEDIT",45,22,21,10
FF_SetControlStyle "Test VSLIDER","AnzVSL",2049
FF_AddControl "Test VSLIDER","Ende","BUTTON",39,59,33,11
'** End Dialog Test VSLIDER **

FF_SetControl "Test VSLIDER","VSL", CInt((Start - Min) * 255 / (Max - Min))
FF_SetDialogTimer "Test VSLIDER",100        'Dialogtimer setzen (100ms)

do
        Wert = FF_GetControl ("Test VSLIDER","VSL")  'Wert des Sliders abfragen,
        Wert = CInt(Wert * (Max - Min) / 255 + Min)  '       anpassen an die Konstanten
        FF_SetControl "Test VSLIDER","AnzVSL", Wert  '       und anzeigen

        Select Case FF_ShowDialog ("Test VSLIDER")   'Befehle abfragen und Dialog öffnen
        Case "CANCEL"                                'bei Cancel Abbruch des Skript
                exit do
        Case "Ende"                                  'bei Ende Schleife verlassen
                msgbox  "Es wurde eingestellt:" & vbNewLine & _
                        "Wert " & FF_GetControl("Test VSLIDER","AnzVSL")
                exit do
        End Select
loop

FF_CloseDialog ("Test VSLIDER")                      'Dialog schließen
```

Wie man sieht, liegt der Einstellwert zwischen 60 und 100 und der Voreinstellwert bei 75, was sich leicht über die Konstantenwerte Min, Max und Start ändern lässt. Wird kein Startwert vorgegeben, ist die Zeile

```
Wert = CInt(Wert * (Max - Min) / 255 + Min)
```

unbedingt abzuändern in

```
If len(Wert) > 0 then Wert = CInt(Wert * (Max - Min) / 255 + Min)
```

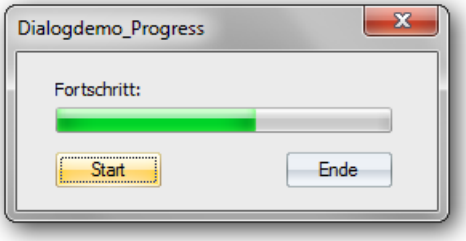
da sonst beim ersten Durchlauf ein Nullstring zu einer Fehlehrmeldung führt.

Der Befehl `FF_SetControlBuddy` entfällt.
Da hier eine Zeitsteuerung vorliegt, ist das Verhalten nicht immer sicher. Die Betätigung des **x** oben rechts funktioniert nicht und auch die Betätigung eines Button wird nicht immer erkannt. Evtl. muss man ein bisschen mit der Zeit beim Befehl `FF_SetDialogTimer` *dialog, zeit in ms* experimentieren.

PROGRESS

Wenn eine Skriptbearbeitung länger dauert, ist es sinnvoll dem Benutzer den Fortschritt der Bearbeitung anzuzeigen. Dazu kann man lt. Bedienanleitung die Befehle `FF_SetProgress progress,range` und `FF_SetProgressText progress,range,text` verwenden oder ab FixFoto-Version 2.9 im Dialog das Element PROGRESS einsetzen.

Hier ein Beispiel:




```
'FFSubmenu=Demo
'FFName=Dialog_Progress

Option Explicit
Dim Taste,m,n

'** Start Dialog Dialogdemo_Progress **
FF_AddDialog "Dialogdemo_Progress",131,49
FF_AddControl "Dialogdemo_Progress","Fortschritt:","STATIC",12,6,33,11
FF_AddControl "Dialogdemo_Progress","Progress","PROGRESS",12,17,105,8
FF_AddControl "Dialogdemo_Progress","Start","BUTTON",12,31,33,11
FF_AddControl "Dialogdemo_Progress","Ende","BUTTON",84,31,33,11
'** End Dialog Dialogdemo_Progress **

call Main
'-----------------------------------------------------------------
Sub Main

do
        Taste = FF_ShowDialog ("Dialogdemo_Progress")       'Befehle abfragen und Dialog öffnen

        Select Case Taste
        Case "Start"
                call Fortschrittsanzeige
        Case "Ende","CANCEL"                                'mit Ende oder Cancel Programmende
                exit do
        Case Else
        End Select
loop

FF_CloseDialog ("Dialogdemo_Progress")                      'Dialog schließen
End Sub
'-----------------------------------------------------------------
Sub Fortschrittsanzeige
Dim max : max = 10000

For n = 1 to max
        FF_SetControl "Dialogdemo_Progress","Progress",n/max*100
next

End Sub
'-----------------------------------------------------------------
```

Mit *Start* wird gezählt von 1 bis 10.000. Der Fortschritt wird für die Anzeige durch die Berechnung **n/max*100** normiert auf 100% und mit `FF_SetControl` an das Steuerelement übertragen.

Zusammenfassung

Die wichtigen Punkte zur Dialogerstellung allgemein und zu GROUP, BOOL, STATIC und BUTTON sind im Kapitel **Zwischenbilanz** bereits aufgeführt.

Für die übrigen Steuerelemente ist folgendes zu merken:

- EDIT erlaubt mehrzeiligen Text, SLEDIT (**S**ingle**L**ine**Edit**) nur einzeiligen Text.
- Bei EDIT und SLEDIT gibt es viele, gut nutzbare Stile (siehe entsprechendes Kapitel)
- Bei Steuerelementen des Typs COMBO und LISTBOX fügt jeder Aufruf von `FF_SetControl (dialog,element,wert)` der Liste einen weiteren Eintrag (wert) hinzu; der wert "" löscht die gesamte Liste.
- Wird ein Wert für COMBO und LISTBOX zweimal definiert, so wird er als Vorgabe selektiert.
- Ein leerer String löscht die Liste bei COMBO und LISTBOX.
- Ein IMAGE-Steuerfeld sollte immer quadratisch angelegt werden, um hoch- und querformatige Bilder gleich gut darstellen zu können.
- Für IMAGE kann mit `FF_SetControl dialog,element,wert` als *wert* ein Pfad auf ein Bild oder der Index des Stacks angegeben werden, wobei 0 dem aktuellen Bild entspricht.
- Bei FONT wird die ausgewählte Schrift im Klartext ausgegeben, bei COLOR erhält man die dezimalen RGB-Werte der gewählten Farbe.
- HSLIDER, VSLIDER und VSPIN sollten immer zusammen mit einem Textfeld eingesetzt werden.
- HSLIDER bzw. VSLIDER sollten zur Werteanzeige mit einem dazugehörigen Textfeld über den Befehl `FF_SetControlBuddy` verbunden oder ab FF-Version 2.85 unter Einsatz des Befehls `FF_SetDialogTimer` der Wert mit `FF_GetControl` abgeholt und mit `FF_SetControl` in das Textfeld eingetragen werden. Im letzteren Fall ist der Wert manipulierbar.
  Bei VSPIN muss ein Aufaddieren von Zahlenwerten (String erst in eine Zahl umwandeln) erfolgen, dessen Ergebnis anschließend in ein Textfeld übertragen wird.
- Bei PROGRESS muss der mit `FF_SetControl` zu übertragende Wert auf 100% normiert werden durch die Formel **Ereignisse/max. Ereigniszahl * 100**.

Noch ein Tipp:

Alle Beispiele können schnell ausprobiert werden, indem jeweils der Programmcode aus der vorliegenden Beschreibung kopiert und in den FFSkript-Editor eingefügt wird. Bei Betätigung der Taste *Ausführen* wird dann das Beispielskript gestartet.
Wenn man will, kann man bei dieser Gelegenheit das Skript auch unter einem beliebigen Namen abspeichern und mit der *Automatischen Konfiguration* ein Eintrag der Kategorie *Demo* mit Dialognamen in der Skriptliste erzeugt werden.

Stile

Stile werden im Dialogeditor angeboten für die Steuerelemente

BOOL, STATIC, BUTTON, EDIT und SLEDIT

Die Stile für BUTTON sind freilich ohne Wirkung und brauchen daher nicht beachtet zu werden.
Es gibt weitere, nicht über Dialogeditor setzbare Stile für EDIT und SLEDIT.

Im Dialogeditor wird beim "*Stile setzen*" ein vorhandener Stil mit Konstantenname und Nennung des hexadezimalen Werts der Konstante angeboten, beispielsweise ES_READONLY,=x0800 bei EDIT und SLEDIT. ES_READONLY ist die Konstante mit dem Hexwert 800. Beim Speichern des Dialogs wird automatisch der Hexwert als Dezimalwert, hier also 2048 (8×16^2) eingetragen mit

```
FF_SetControlStyle "dialog","elementname",2048
```

Werden mehrere Stile gleichzeitig gesetzt, wird die Summe der Dezimalwerte eingetragen.
Die Umrechnung von Hex auf Dezimal kann beispielsweise bei WINDOWS mit dem naturwissenschaftlichen Rechner unter Zubehör oder über Excel (mal die Hilfe fragen!) oder natürlich manuell (siehe bei WIKIPEDIA unter Hexadezimalsystem) erfolgen.

Wenn manuell Stile mit FF_SetControlStyle gesetzt werden sollen, definiert man am besten diese als Konstanten, z.B. :

```
const ES_READONLY = 2048
const ES_RIGHT = 2
```

und setzt beispielsweise ES_READONLY mit

```
FF_SetControlStyle "dialog","elementname", ES_READONLY
```

Will man beide Stile setzen, verknüpft man diese mit "+" oder "Or":

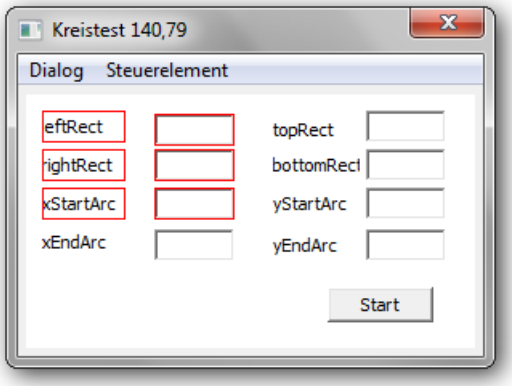
```
FF_SetControlStyle "dialog","elementname", ES_READONLY + ES_RIGHT
```

Im Anhang sind die interessanten Stile als Konstanten mit ihren Dezimalwerten zum Kopieren und Einfügen in ein Skript aufgelistet. Auch eine Tabelle mit allen im FF-Dialogeditor verwendeten Stilen und deren Hex- und Dezimalwerte ist mit Kurzerläuterung dort zu finden.

Dialogdesign ändern

Bei einem bestehenden Dialog lassen sich Steuerelemente leicht hinzufügen, umbenennen, in der Größe ändern und einzeln verschieben. Vor allem das Verschieben kann aber bei größeren Veränderungen, notwendig z.B. bei Erweiterungen, in Arbeit ausarten. Ab FixFoto-Version 3.30 gibt es daher die Möglichkeit mehrere Elemente zu markieren und diese zusammen zu verschieben.

Am einfachsten geht die Markierung mit der rechten Maustaste. Das markierte Element wird rot eingerahmt und macht man das mit mehreren Elementen, lassen die sich gemeinsam handeln. Wird ein markiertes Element erneut mit der rechten Maustaste angeklickt, wird die Markierung dieses Elementes wieder aufgehoben. Bei allen Elementen gleichzeitig die Markierung beseitigen kann man über den Menüpunkt "*Steuerelement / Keine auswählen".* Auf ähnliche Weise sind auch alle markierbar durch "*Steuerelement / Alle auswählen*".
Schneller geht es meistens, wenn man mit der linken Maustaste ein Rechteck aufzieht. Dann werden alle Elemente innerhalb des Rechtecks markiert. Werden dadurch mal zu viele Elemente rot gekennzeichnet, kann man bei diesen die Markierung mit der rechten Maustaste wieder aufheben.



Manuelle Dialogerstellung

Die Dialogerstellung per FF_Dialogeditor ist eine grafische Erstellung, die keinerlei Kenntnisse der Dialogbefehle erfordert. Der Code wird mit dem Speicherbefehl automatisch erzeugt.

Natürlich kann man aber auch Dialoge direkt über Code erzeugen. Man benutzt dazu die gleichen Befehle, die der Dialogeditor generiert.

Die einfassenden Kommentarzeilen

```
'** Start Dialog Dialogdemo **
```

und

```
'** End Dialog Dialogdemo **
```

werden nicht benötigt.

- Man beginnt mit dem Befehl `FF_AddDialog`, mit dem die Benutzerfläche mit ihrem Namen und ihrer Größe festgelegt wird.
- Mit `FF_AddControl` werden dann die Steuerelemente mit Festlegung ihres Namens, Typs, Größe und Lage hinzugefügt.
- `FF_SetControlStyle` dient dazu, bei Bedarf den Steuerelementen Stile (Eigenschaften) zuzuordnen.

Die Syntax dieser Befehle wird im Anhang unter *Dialogbefehle* erläutert.

Das Interessante bei der manuellen Erstellung ist, dass alle Parameter als Variable oder Konstante übergeben werden können. Will man beispielsweise den Dialognamen zentral festlegen, verwendet man für den Namen statt eines Strings eine Stringvariable und ordnet ihr zentral einen Namen zu.

Die Parameter für Größe und Lage kann man ebenfalls als Variable oder Konstante übergeben und man kann sie mathematisch behandeln. Soll z.B. der Abstand mehrerer untereinander angeordneter Steuerelemente gleich sein, hält man im Befehl

```
FF_AddControl "dialog","name","type",left,top,width,height
```

den `left`-Wert gleich (beispielsweise 10) und multipliziert den `top`-Wert mit einem wachsenden Faktor, z.B. 20, 20 x 2, 20 x 3 ect.

Man kann natürlich auch erst einmal das Grundgerüst der Benutzeroberfläche grafisch erstellen und den so erzeugten Code anschließend manuell ändern und ergänzen.
Aber Vorsicht, je nach Änderung funktioniert der Weg zurück zum Dialogeditor nicht mehr.

Empfehlung

Viele Tipps und Beispiele zur Erstellung von FixFoto-Skripten sind auf der Webseite

http://fixfoto-scripting.wikispaces.com/

zu finden.

Historie

2. Ausgabe vom März 2008

- diverse Korrekturen und Ergänzungen
- Element VSLIDER (neu ab Version 2.85) hinzugefügt
- zusätzliches Kapitel: Stile
- zusätzliches Kapitel: Manuelle Dialogerstellung
- Anhang mit den wichtigsten Dialogbefehlen, Stilkonstanten zum Kopieren und Tabelle aller Stilkonstanten mit Kurzerläuterung
- Inhaltsverzeichnis

3. Ausgabe vom November 2008

- Ergänzung bei LISTBOX und COMBO

4. Ausgabe vom Februar 2009

- Element PROGRESS (neu ab Version 2.90) hinzugefügt
- einige kleine Ergänzungen

5. Ausgabe vom Mai 2012

- Generelle Überarbeitung der Beschreibung
- Mehrere Dialogelemente markierbar und gemeinsam verschiebbar (ab Version 3.10)
- COLOR-Steuerelement Farbwahl erzeugt Button-Event (ab Version 3.30)

Anhang

Ereignisse

Steuerelement	Ereign. ausgelöst durch ..	Anmerkung
GROUP	-	kein Ereignis
BOOL	Einfachklick	Stil **BS_NOTIFY** erforderlich
STATIC	-	kein Ereignis
BUTTON	Einfachklick	-
EDIT	-	kein Ereignis
SLEDIT	-	kein Ereignis
LISTBOX	Doppelklick	-
COMBO	Auswahl eines Eintrags	-
COLOR	Auswahl eines Eintrags	ab FF-Version 3.30
FONT	Auswahl eines Eintrags	-
IMAGE	Einfachklick	-
HSLIDER	-	kein Ereignis
VSLIDER	-	kein Ereignis
VSPIN	-	kein Ereignis
PROGRESS	-	kein Ereignis

Dialogbefehle

1. FF_ActivateControl "dialog","name"
   Setzt den Focus auf das angegebene Steuerelement <name> im Dialog <dialog>.

   dialog = Name des Dialogs
   name = Name des Steuerelements

2. FF_AddControl "dialog","name","type",left,top,width,height
   Fügt <dialog> das Steuerelement <name> hinzu.
   Folgende Typen werden unterstützt: STATIC, EDIT, SLEDIT, BUTTON, IMAGE, BOOL, COLOR, GROUP, COMBO, FONT, LISTBOX, HSLIDER, VSLIDER, PROGRESS, LABEL.
   Enthaltene Pipe-Zeichen '|' in den Namen von BUTTON und STATIC werden bei der Anzeige unterdrückt. Dies ermöglicht die Verwendung identischer Beschriftungen.

   dialog = Name des Dialogs
   name = Name des Steuerelements
   type = Typ des Elements
   left = Abstand von links
   top = Abstand von oben
   width = Breite des Elements
   height = Höhe des Elemente

3. FF_AddDialog "dialog",width,height
   Legt einen neuen Dialog <dialog> an, Größe in Dialog-Einheiten

   dialog = Name des Dialogs
   width = Breite des Dialogs
   height = Höhe des Dialogs

4. value = FF_CheckKey
   Liefert den Zahlencode der gedrückten Taste

   value = numerischer Keycode

5. FF_ClearControlStyle "dialog","name",style
   Entfernt Fenstereigenschaften des Steuerelements <name> im Dialog <dialog>.

   dialog = Name des Dialogs
   name = Name des Steuerelements
   style = Fenstereigenschaft des Steuerelements (Dezimalwert der Konstante)

6. FF_CloseDialog "dialog"
   Löscht den Dialog <dialog>

   dialog = Name des Dialogs

7. FF_EnableControl "dialog","name",enable
   Aktivieren / Deaktivieren des Steuerelements <name> im Dialog <dialog>

   dialog = Name des Dialogs
   name = Name des Steuerelements
   enable = true für Aktivierung oder false für Deaktivierung des Steuerelements

8. value = FF_GetControl ("dialog","name")
   Liefert den Wert des Steuerelements <name> im Dialog <dialog>

   value = Wert des Steuerelements
   dialog = Name des Dialogs
   name = Name des Steuerelements

9. value = FF_GetControlHeight ("dialog","name")
   Liefert aus <dialog> die Höhe eines Steuerelementes in Pixeln (nicht identisch mit den Dialogeinheiten).
   Unbedingt erst anwenden nach dem ersten Aufruf von FF_OpenDialog oder FF_ShowDialog

   dialog = Name des Dialogs
   control = Name des Steuerelements
   value = Ergebnis

10. value = GetControlWidth ("dialog","name")
    Liefert aus <dialog> die Breite eines Steuerelementes in Pixeln (nicht identisch mit den Dialogeinheiten).
    Unbedingt erst anwenden nach dem ersten Aufruf von FF_OpenDialog oder FF_ShowDialog

    dialog = Name des Dialogs
    control = Name des Steuerelements
    value = Ergebnis

11. FF_OpenDialog "dialog"
    Zeigt den Dialog <dialog> an und kehrt zurück, ohne auf eine Aktion zu warten

    dialog = Name des Dialogs

12. FF_RefreshDialog "dialog"
    Aktualisiert das angegebene Dialogfenster

    dialog = Name des Dialogs

13. FF_SendControl ("dialog","control","message",wparam,lparam)
    Erlaubt den direkten Durchgriff auf das zugrundeliegende Windows API.
    Nur für Experten

    dialog = Name des Dialogs
    control = Name des Steuerelements
    message = Text
    wparam und lparam = Parameter

14. FF_SetControl "dialog","name","value"
    Setzt das Steuerelement <name> im Dialog <dialog> auf den Wert <value>.

    Bei Steuerelementen des Typs COMBO fügt jeder Aufruf der Liste einen weiteren Eintrag hinzu. Wird ein Wert zweimal definiert, so wird er als Vorgabe selektiert. Ein leerer String löscht die Liste bei COMBO + LISTBOX. Bei IMAGE kann ein Pfad auf ein Bild oder der Index des Stacks angegeben werden, wobei 0 dem aktuellen Bild entspricht.

    dialog = Name des Dialogs
    name = Name des Steuerelements
    value = Wert des Steuerelements (bei IMAGE wird mit "" das aktuelle Bild gelöscht)

15. FF_SetControlBuddy "dialog","control","buddy"
    Ordnet dem Steuerelements <control> im Dialog <dialog> ein anderes Steuerelement <buddy> zu, welches bei Änderungen automatisch aktualisiert wird. z.Zt von HSLIDER, VSLIDER, LISTBOX, COLOR genutzt.

    dialog = Name des Dialogs
    control = Name des Steuerelements
    buddy = Steuerelement, welches bei Änderungen automatisch aktualisiert wird

16. FF_SetControlImage "dialog","control","imagepath"
    Zeigt eine Grafik anstelle eines Textes im BUTTON an.

    Folgende Grafikformate werden dabei unterstützt: *. jpg, *.jp2, *.png, *.tif und *.bmp
    IMAGE wird ohne Rahmen dargestellt.
    Die Grafik vorher verkleinern auf passende Pixelgröße (z.B. in der Größenordnung 16 x 32)

    dialog = Name des Dialogs
    control = Name des Steuerelements
    imagepath = Pfad zum Bild

17. FF_SetControlStyle "dialog","name",style
    Definiert zusätzliche Fenstereigenschaften des Steuerelements <name> im Dialog <dialog>.
    Ist <name> = "" wird der Dialog selbst verändert.

    dialog = Name des Dialogs
    name = Name des Steuerelements
    style = Stil bzw. Fenstereigenschaft des Steuerelements (Dezimalwert der Konstante)

18. FF_SetDialogTimer ("name", msec) ???
    Aktiviert einen Timer, der den Rückgabewert "TIMER" aus FF_ShowDialog() liefert.

    msec = Zeit in ms (0 = Timer aus)

19. "value" = FF_ShowDialog ("dialog")
    Zeigt den Dialog <dialog> an und liefert den Namen des gedrückten Buttons an <value>

    dialog = Name des Dialogs
    value = Wert mit dem Namen des gedrückten Buttons

20. FF_SetProgress (progress,range)
    Anzeige des Fortschritts unterhalb der Arbeitsfläche

    progress = Fortschritt
    range = Bereich

21. FF_SetProgress (progress,range,text)
    Anzeige des Fortschritts unterhalb der Arbeitsfläche mit vorangestelltem Text

    progress = Fortschritt
    range = Bereich
    text = Text

Die interessantesten Stile als Konstanten

```
'BOOL-Styles
const BS_PUSHLIKE = 4096          'aus "mit Kreuz" / "ohne Kreuz"  wird "versenkt" / "erhaben"
const BS_NOTIFY = 16384           'Anklicken als Befehl erfassbar

'STATIC-Styles
const SS_CENTER = 1               'Text zentrisch angeordnet
const SS_RIGHT = 2                'Text rechts angeordnet
const SS_BLACKFRAME = 7           'schwarzer Rahmen ohne Text
const SS_GRAYFRAME = 8            'grauer Rahmen ohne Text
const SS_WHITEFRAME = 9           'weißer Rahmen ohne Text
const SS_ETCHEDFRAME = 18         'versenkter Rahmen (wie GROUP) ohne Text
const SS_ENDELLIPSIS = 16384      'zu langer Text wird abgebrochen mit ...

'EDIT und SLEDIT
const ES_CENTER = 1               'Text zentrisch angeordnet
const ES_RIGHT = 2                'Text rechts angeordnet
const ES_UPPERCASE = 8            'nur Großbuchstaben, klein wird groß
const ES_LOWERCASE = 16           'nur Kleinbuchstaben, groß wird klein
const ES_PASSWORD = 32            'eingegebene Zeichen dargestellt als Punkte (nur bei SLEDIT)
const ES_READONLY = 2048          'nur Lesen zugelassen, Schreiben gesperrt
const ES_NUMBER = 8192            'lässt nur Zahlen für die Eingabe zu

const WS_HSCROLL = 1048576        'horizontale Scrollbar
const WS_VSCROLL = 2097152        'vertikale Scrollbar
const WS_THICKFRAME = 262144      'Doppelrahmen in seiner Größe veränderbar
```

Obige Stile können kopiert und in ein Skript eingefügt werden,
setzbar mit

```
FF_SetControlStyle "dialog","elementname",konstante
```

Mehrere Stile für dasselbe Steuerelement werden mit "+" oder "Or" verknüpft.

```
FF_SetControlStyle "dialog","elementname",konstante1 + konstante2 + .....
```

Alle Stile mit Erläuterung

Element / Stil	Hex-Wert	Dez.-Wert	Erläuterung
BOOL			
BS_PUSHLIKE	**0x 00001000**	4096	aus "mit Kreuz" / "ohne Kreuz" wird "versenkt" / "erhaben"
BS_NOTIFY	**0x 00004000**	16384	Anklicken als Befehl erfassbar
STATIC			
SS_BLACKFRAME	**0x 00000007**	7	schwarzer Rahmen ohne Text
SS_BLACKRECT	**0x 00000004**	4	schwarze Fläche deckend
SS_GRAYFRAME	**0x 00000008**	8	grauer Rahmen ohne Text
SS_GRAYRECT	**0x 00000005**	5	graue Fläche deckend
SS_WHITEFRAME	**0x 00000009**	9	weißer Rahmen ohne Text
SS_WHITERECT	**0x 00000006**	6	weiße Fläche ohne Text
SS_ETCHEDFRAME	**0x 00000012**	18	versenkter Rahmen ohne Text
SS_SUNKEN	**0x 00001000**	4096	versenkte Fläche mit Text
SS_ETCHEDHORZ	**0x 00000010**	16	nur obere, versenkte Kante ohne Text
SS_ETCHEDVERT	**0x 00000011**	17	nur linke, versenkte Kante ohne Text
SS_LEFT	**0x 00000000**	0	Text links angeordnet (Standard)
SS_CENTER	**0x 00000001**	1	Text zentrisch
SS_RIGHT	**0x 00000002**	2	Text rechts angeordnet
SS_SIMPLE	**0x 0000000B**	11	Text oben links angeordnet
SS_ENDELLIPSIS	**0x 00004000**	16384	zu langer Text wird abgebrochen mit ... (wie mit SS_WORDELLIPSIS)
SS_WORDELLIPSIS	**0x 0000C000**	49152	zu langer Text wird abgebrochen mit ... (wie mit SS_ENDELLIPSIS)
SS_LEFTNOWORDWRAP	**0x 0000000C**	12	Text linksbündig, der bei Überlänge abgeschnitten wird
SS_NOPREFIX	**0x 00000080**	128	Es wird "&" angezeigt und keine Unterstreichung vorgenommen
BUTTON (alle BS_Konstanten im FF-Dialogeditor ohne Wirkung)			
BS_BOTTOM	**0x 00000800**	2048	Platziert Text am unteren Rand des Schalter-Rechtecks
BS_CENTER	**0x 00000300**	768	Text wird horizontal zentriert (Standard bei BUTTON)
BS_DEPUSHBUTTON	**0x 00000001**	1	Voreingestellter Schalter, wird mit ENTER-Taste betätigt
BS_FLAT	**0x 00008000**	32768	flache Darstellung ohne 3-D-Schattierung
BS_LEFT	**0x 00000100**	256	Text linksbündig angeordnet
BS_MULTILINE	**0x 00002000**	8192	Text wird bei Bedarf umgebrochen
BS_RIGHT	**0x 00000200**	512	Text rechtsbündig angeordnet
BS_TOP	**0x 00000004**	4	Platziert Text am oberen Rand des Schalter-Rechtecks
BS_VCENTER	**0x 00000C00**	3072	Text wird vertikal zentriert (Standard bei BUTTON)
EDIT und SLEDIT			
ES_LEFT	**0x 00000000**	0	Text linksbündig angeordnet (Standard)
ES_CENTER	**0x 00000001**	1	Text zentrisch angeordnet
ES_RIGHT	**0x 00000002**	2	Text rechts angeordnet
ES_LOWERCASE	**0x 00000010**	16	nur Kleinbuchstaben, Großb. werden in Kleinb. umgewandelt
ES_UPPERCASE	**0x 00000008**	8	nur Großbuchstaben, Kleinb. werden in Großb. umgewandelt
ES_NOHIDESEL	**0x 00000100**	256	die Auswahl wird auch angezeigt wenn das Element den Fokus verliert
ES_NUMBER	**0x 00002000**	8192	lässt nur Zahlen für die Eingabe ins Bearbeitungsfeld zu
ES_PASSWORD	**0x 00000020**	32	eingegebene Zeichen werden als Punkte dargestellt (wirksam nur bei SLEDIT)
ES_READONLY	**0x 00000800**	2048	nur Lesen zugelassen, Schreiben gesperrt
nicht über Dialogeditor setzbar (für EDIT und SLEDIT)			
WS_VSCROLL	**0x 00200000**	2097152	vertikale Scrollbar
WS_HSCROLL	**0x 00100000**	1048576	horizontale Scrollbar
WS_THICKFRAME	**0x 00040000**	262144	erzeugt ein Fenster mit in seiner Größe veränderbarem Doppelrahmen

Inhaltsverzeichnis