



# VFX 8.0 – Was ist neu?

August - Oktober 2003

# Inhaltsverzeichnis

Neue Entwicklerwerkzeuge .....	3
Neue Eigenschaften der Klasse cApplication .....	4
Die Klasse cTreeViewForm.....	5
Einstellungen zur Datenanbindung des TreeView-Steuerelements .....	5
Layout-Einstellungen des TreeView-Steuerelements .....	6
Die Klasse cTreeViewOneToMany .....	7
Einstellungen zur Datenanbindung des TreeView-Steuerelements .....	7
Layout-Einstellungen des TreeView-Steuerelements .....	8
Die Klasse cPickAlternate .....	9
Die Klasse cDownload .....	11
Eigenschaften .....	11
Methoden .....	11
Befehle der Makrosprache .....	11
Beispiel .....	12
Die Klasse cCreatePDF .....	14
Eigenschaften .....	14
Methoden .....	14
Die Klasse cEmail .....	15
Eigenschaften .....	15
Methoden .....	15
Die Klasse cArchive .....	17
Eigenschaften .....	17
Methoden .....	17
VFX 8.0 Task Pane .....	19
Öffnen-Dialog im Windows-XP-Stil.....	20
Aktualisierung einer SQL Server Kundendatenbank .....	21
Applikationsschutz durch Produktaktivierung.....	23
Liste der verwendeten Begriffe .....	24
Eigenschaften der Klasse cVFXActivation .....	25
Das Funktionsprinzip.....	26
Die Definition der Aktivierungsregeln .....	26
Definition der Rechte für einen Anwender .....	28
Der VFX Menü-Designer .....	29

## Neue Entwicklerwerkzeuge

Neue Power Builder für die Klassen:

- cTreeViewForm
- cTreeViewOneToMany
- cPickAlternate

Produktaktivierungsassistenten: (Siehe: Applikationsschutz durch Produktaktivierung)

- Define Activation Rules – Einstellen der Systemeigenschaften, die zur Produktaktivierung verwendet werden sollen.
- Create Activation Key – Erstellen eines Aktivierungsschlüssels anhand des Installationschlüssels des Kunden.

Assistent zum Anlegen von SQL Metadaten:

- Create meta data definition

Menü-Designer:

- VMD (Visual Extend Menu Designer)

## Neue Eigenschaften der Klasse cApplication

Die Klasse des Applikationsobjekts wurde um eine Reihe neuer Eigenschaften zur Steuerung des Verhaltens der Applikation im Fehlerfall, zur Verwendung der Produktaktivierung und zur Installation eines Postscript-Druckertreibers, der zur Erstellung von PDF-Dateien benötigt wird, erweitert. Die Werte dieser Eigenschaften können in Vfxmain.prg unter *DEFINE CLASS capplication AS cfoxapp* eingestellt werden.

*nAppOnErrorBehavior* – Diese Eigenschaft steuert das Verhalten der Applikation im Fehlerfall.

- 0 – Laufzeitfehler werden ignoriert.
- 1 – Es wird eine Fehlermeldung angezeigt (Standardwert).
- 2 – Die Ausführung der Applikation wird beendet.

*ErrorDetailLevel* – Diese Eigenschaft steuert welche Informationen im Fehlerfall protokolliert werden.

- 0 – Nur die Fehlermeldung aber keine Information über den Aufrufstapel.
- 1 – Die Fehlermeldung und Informationen über den Aufrufstapel (Standardwert).
- 2 – Vollständige, detaillierte Fehlerinformationen.

*PSPrinterToInstall* – Diese Eigenschaft enthält den Namen des Standard-Postscript-Druckertreibers. Dieser Druckertreiber wird automatisch installiert, wenn die Applikation einen Postscript-Druckertreiber braucht um eine PDF-Datei zu erstellen. Der Standardwert ist "HP DeskJet 1200C/PS".

*cConnectionCheckURL* – Diese Eigenschaft enthält die Adresse einer Internetseite, die verwendet wird um zu testen, ob eine Internet-Verbindung besteht. Diese Eigenschaft wird benötigt wenn Ghostscript nicht installiert ist. Ghostscript wird bei Bedarf automatisch aus dem Internet heruntergeladen und installiert. Ghostscript wird verwendet um Postscript-Dateien in PDF-Dateien umzuwandeln. Wenn keine Verbindung mit dem Internet besteht und auch keine DFÜ-Netzwerkverbindung eingerichtet ist, wird von VFX ein Eintrag im DFÜ-Netzwerk angelegt. Alle Eigenschaften der DFÜ-Verbindung können vom Entwickler vorgegeben werden. Der Anwender kann bei Bedarf in einem Dialog die Telefonnummer, den Benutzernamen und das Kennwort ändern.

*lUseActivation* – Über diese Eigenschaft wird die Produktaktivierung ein- bzw. ausgeschaltet. Diese Eigenschaft kann im VFX Application Wizard eingestellt werden, wenn ein neues Projekt erstellt wird. Später kann der Eintrag in Vfxmain.prg geändert werden. Der Standardwert ist .F., die Produktaktivierung wird nicht verwendet.

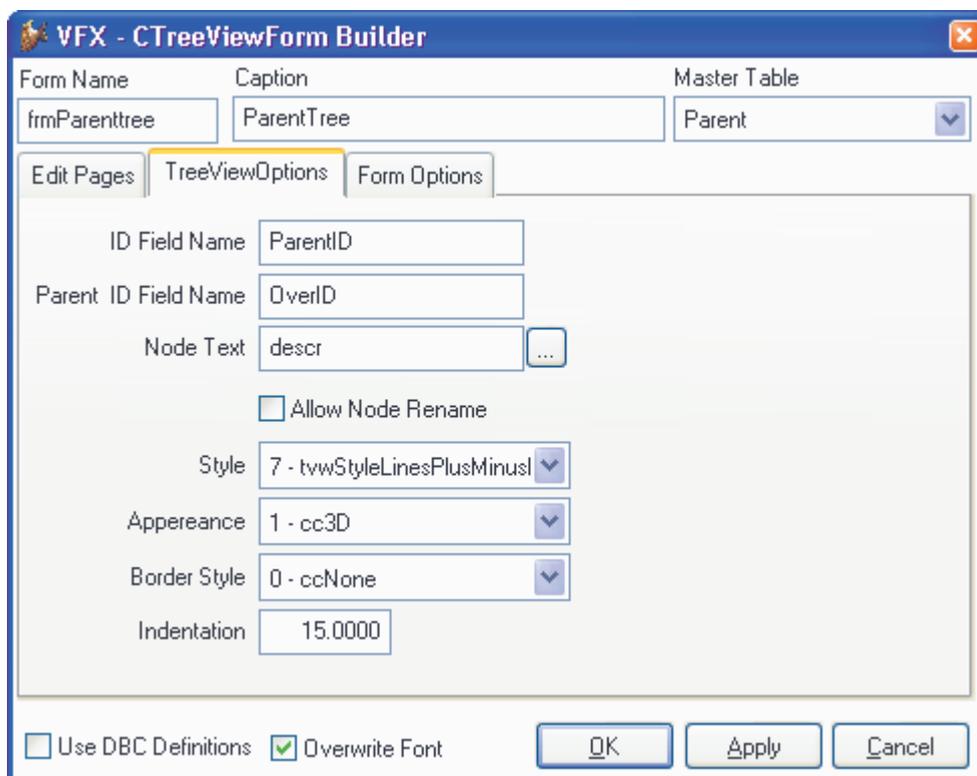
*lActivationType* – Wenn diese Eigenschaft auf .T. gesetzt wird, überprüft die Klasse cActivation ob die Datei „FirstInstall.txt“ existiert, wenn die Applikation gestartet wird. Diese Eigenschaft kann im VFX Application Wizard eingestellt werden, wenn ein neues Projekt erstellt wird. Später kann der Eintrag in Vfxmain.prg geändert werden. Der Standardwert ist .F., es wird nicht auf das Vorhandensein der Datei „FirstInstall.txt“ geprüft.

## Die Klasse cTreeViewForm

Der Haupteinsatzzweck dieser Klasse ist die Darstellung von Daten aus einer Tabelle in einer Baumstruktur. Die Baumstruktur gibt dem Endanwender einen kompletten Überblick über die hierarchischen Beziehungen in einer Tabelle.

Diese Klasse basiert auf der Klasse cDataFormPage (Vfxform.vcx) und enthält ein Treeview-Steuer-element aus der Klasse cTreeView (Vfxappl.vcx). Die Klasse kombiniert die Funktionalität von cDataFormpage mit den Möglichkeiten der hierarchischen Datenpräsentation in einer Baumstruktur. Wenn ein Eintrag im Treeview-Steuer-element ausgewählt wird, wird der Datensatzzeiger in der zugrunde liegenden Tabelle mitgeführt und der Anwender kann die Daten im rechten Teil des Formulars bearbeiten.

Mit dem VFX – CTreeViewForm Builder können sehr schnell Formulare basierend auf der Klasse cTreeViewForm erstellt werden und alle benötigten Eigenschaften können eingestellt werden.



Der Builder arbeitet so ähnlich wie der VFX – CDataFormPage Builder. Die Einstellungen können auf den Seiten *Edit Pages* und *Form Options* genauso gemacht werden, wie im VFX – CDataFormPage Builder. Zusätzlich müssen die Einstellungen für das Treeview-Steuer-element auf der Seite *TreeViewOptions* gemacht werden. Es müssen zwei Arten von Einstellungen für das Treeview-Steuer-element gemacht werden.

### **Einstellungen zur Datenanbindung des TreeView-Steuer-elements**

*IDFieldName* – Hier wird der Name des Feldes mit dem Primärschlüssel der Bearbeitungstabelle eingetragen.

*ParentIDFieldName* – Diese Eigenschaft enthält den Namen des Feldes, in dem der Primärschlüssel des Parent-Datensatzes gespeichert ist.

*NodeText* – Hier kann entweder der Name eines Feldes, das einen Beschreibungstext enthält eingetragen werden oder es wird ein Ausdruck eingetragen, der zur Laufzeit evaluiert wird und dessen Rückgabewert als Bezeichnung in der Baumstruktur angezeigt wird. Wenn ein Feldname verwendet wird, kann dem Anwender erlaubt werden die Bezeichnung direkt im Treeview-Steuerelement zu ändern. Dies hängt vom Wert der Eigenschaft *AllowNodeRename* ab. Wenn *AllowNodeRename* auf *.T.* gesetzt ist, kann der Anwender die Bezeichnungen im Treeview-Steuerelement ändern. Dabei werden die Daten im zugrunde liegenden Tabellenfeld automatisch aktualisiert.

*AllowNodeRename* – Über diese Eigenschaft wird gesteuert, ob der Anwender die Bezeichnung im Treeview-Steuerelement ändern kann. Die Bearbeitung der Bezeichnung im Treeview-Steuerelement ist nur möglich, wenn die Bezeichnung auf einem einzelnen Tabellenfeld basiert. Dieses Tabellenfeld wird bei der Bearbeitung automatisch aktualisiert.

## **Layout-Einstellungen des TreeView-Steuerelements**

Diese Einstellungen entsprechen denen des TreeView-ActiveX-Steuerelements.

*Style:* 0 - *tvwStyleText*  
1 - *tvwStylePictureText*  
2 - *tvwStylePlusMinusText*  
3 - *tvwStylePlusMinusPictureText*  
4 - *tvwStyleLinesText*  
5 - *tvwStyleLinesPictureText*  
6 - *tvwStyleLinesPlusMinusText*  
7 - *tvwStyleLinesPlusMinusPictureText*

*Appearance:* 0 - *ccFlat*  
1 - *cc3D*

*BorderStyle:* 0 - *ccNone*  
1 - *ccFixedSingle*

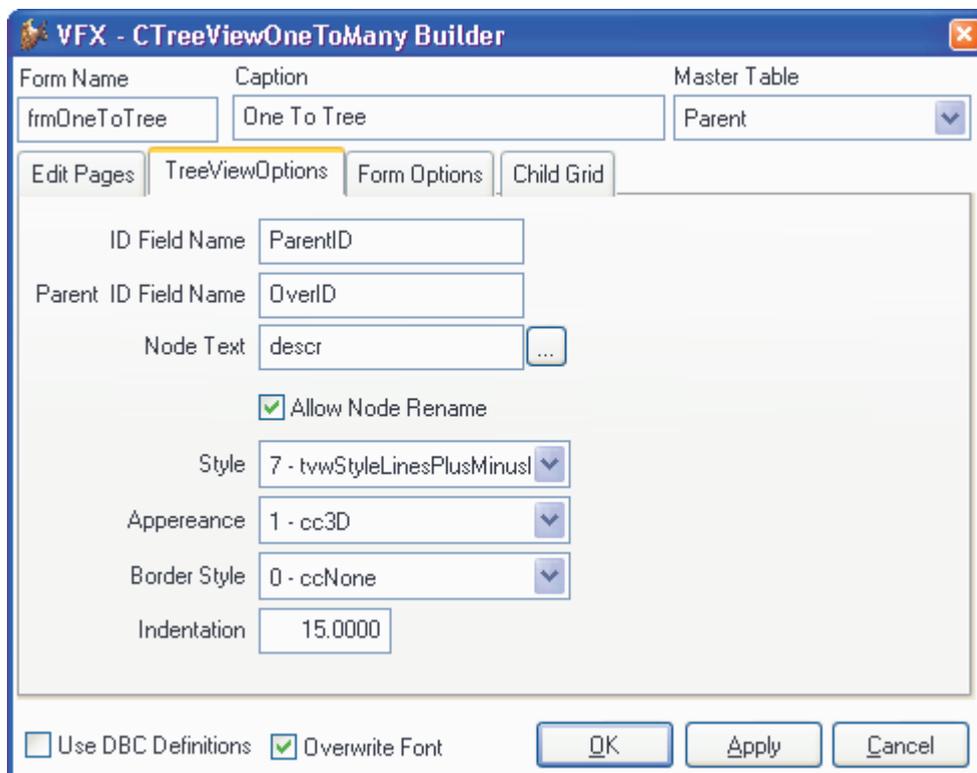
*Indentation:* Diese Eigenschaft bestimmt die Breite des Einzugs der Knoten.

## Die Klasse cTreeViewOneToMany

Der Haupteinsatzzweck dieser Klasse ist die Darstellung der Daten aus einer Tabelle in einer Baumstruktur zusammen mit der leistungsfähigen Funktionalität, die die cOneToMany-Klasse dem Entwickler bietet. Die Baumstruktur gibt dem Anwender den kompletten Überblick über die hierarchischen Datenbeziehungen.

Diese Klasse basiert auf der Klasse cOneToMany (Vfxform.vcx) und enthält ein Treeview-Steuer-element aus der Klasse cTreeView (Vfxappl.vcx). Die Klasse kombiniert die Funktionalität von cOneToMany mit den Möglichkeiten der hierarchischen Datenpräsentation in einer Baumstruktur. Wenn ein Eintrag im Treeview-Steuer-element ausgewählt wird, wird der Datensatzzeiger in der zugrunde liegenden Tabelle mitgeführt und der Anwender kann die Daten im rechten Teil des Formulars bearbeiten. Zusätzlich können die Child-Daten im unteren Teil des Formulars bearbeitet werden.

Mit dem VFX – CTreeViewOneToMany Builder können sehr schnell Formulare basierend auf der Klasse cTreeViewOneToMany erstellt und alle benötigten Eigenschaften eingestellt werden.



Dieser Builder arbeitet so ähnlich wie der VFX – COneToMany Builder. Die Einstellungen auf den Seiten *Edit Pages*, *Form Options* und *Child Grid* werden genauso gemacht, wie bei Formularen basierend auf der Klasse cOneToMany. Zusätzlich müssen die Einstellungen für das Treeview-Steuer-element auf der Seite *TreeViewOptions* gemacht werden. Es müssen zwei Arten von Einstellungen für das Treeview-Steuer-element gemacht werden.

### **Einstellungen zur Datenanbindung des TreeView-Steuer-elements**

*IDFieldName* – Hier wird der Name des Feldes mit dem Primärschlüssel der Bearbeitungstabelle eingetragen.

*ParentIDFieldName* – Diese Eigenschaft enthält den Namen des Feldes, in dem der Primärschlüssel des Parent-Datensatzes gespeichert ist.

*NodeText* – Hier kann entweder der Name eines Feldes, das einen Beschreibungstext enthält eingetragen oder ein Ausdruck eingetragen werden, der zur Laufzeit evaluiert wird und dessen Rückgabewert als Bezeichnung in der Baumstruktur angezeigt wird. Wenn ein Feldname verwendet werden soll, kann dem Anwender erlaubt werden die Bezeichnung direkt im Treeview-Steuerelement zu ändern. Dies hängt vom Wert der Eigenschaft *AllowNodeRename* ab. Wenn *AllowNodeRename* auf *.T.* gesetzt ist, kann der Anwender die Bezeichnungen im Treeview-Steuerelement ändern. Dabei werden die Daten im zugrunde liegenden Tabellenfeld automatisch aktualisiert.

*AllowNodeRename* – Über diese Eigenschaft wird gesteuert, ob der Anwender die Bezeichnung im Treeview-Steuerelement ändern kann. Die Bearbeitung der Bezeichnung im Treeview-Steuerelement ist nur möglich, wenn die Bezeichnung auf einem einzelnen Tabellenfeld basiert. Dieses Tabellenfeld wird bei der Bearbeitung automatisch aktualisiert.

### ***Layout-Einstellungen des TreeView-Steuerelements***

Diese Einstellungen entsprechen denen des TreeView-ActiveX-Steuerelements.

*Style:* 0 - *tvwStyleText*  
1 - *tvwStylePictureText*  
2 - *tvwStylePlusMinusText*  
3 - *tvwStylePlusMinusPictureText*  
4 - *tvwStyleLinesText*  
5 - *tvwStyleLinesPictureText*  
6 - *tvwStyleLinesPlusMinusText*  
7 - *tvwStyleLinesPlusMinusPictureText*

*Appearance:* 0 - *ccFlat*  
1 - *cc3D*

*BorderStyle:* 0 - *ccNone*  
1 - *ccFixedSingle*

*Indentation:* Diese Eigenschaft bestimmt die Breite des Einzugs der Knoten.

## Die Klasse cPickAlternate

Ähnlich zum cPickField-Steurelement kann die Klasse cPickAlternate verwendet werden um eine Benutzereingabe zu verifizieren. Es kann eine Auswahlliste aufgerufen werden, die dem Anwender erlaubt einen Wert aus einer Liste auszuwählen. Diese Klasse ermöglicht dem Anwender einen Wert aus der Auswahltabelle zu verifizieren und zusätzlich einen anderen Wert anzuzeigen.

Das cPickAlternate-Steurelement ist einer Combobox zu bevorzugen, wenn aus einer Tabelle mit vielen Datensätzen ausgewählt werden soll. Der Einsatz ist auch sinnvoll, wenn die Benutzereingabe direkt überprüft werden soll oder wenn der vom Anwender eingegebene Wert nicht dem Schlüssel der Auswahltabelle entspricht. Das Ziel dieser Klasse ist es dem Anwender eine einfach zu bedienende Schnittstelle zu geben, die es erlaubt ihm bekannte Werte einzugeben anstelle von vom Programm generierten Schlüsseln. Der vom Anwender eingegebene Wert wird verwendet um den dazugehörigen Datensatz in der Auswahltabelle zu finden. Wenn der gesuchte Datensatz gefunden ist, wird der Rückgabewert an das cPickAlternate-Steurelement zurückgegeben.

Diese Klasse basiert auf der Klasse cPickField und erbt alle ihre Eigenschaften und Methoden. Zusätzlich hat diese Klasse die neue Eigenschaft *cControlSourceInternalKey* in die der Name des Feldes der Bearbeitungstabelle mit dem Fremdschlüssel eingetragen wird. Dieser Fremdschlüssel entspricht dem Primärschlüssel aus der Auswahltabelle.

Mithilfe des VFX – CPickAlternate Builder können die Eigenschaften dieser Klasse einfach eingestellt werden.

*Pick Table Name* – Hier kann der Name der Auswahltabelle aus einer der Datenquellen der Datenumgebung ausgewählt werden.

*Pick Table Index Tag* – Dies ist der Name des Indexschlüssels, der verwendet wird um in der Auswahltabelle zu suchen. Dieser Indexschlüssel entspricht dem Wert des Eingabefeldes.

*CPickAlternate.txtField.ControlSource* – Die Controlsources des Eingabefeldes. Dieses Feld muss aus der Auswahltabelle stammen.

*CPickAlternate.txtDesc.ControlSource* – Der Name des Beschreibungsfeldes. Der Wert wird nach der erfolgreichen Überprüfung der Benutzereingabe im Beschreibungsfeld angezeigt. Dieses Feld stammt ebenfalls aus der Auswahltabelle.

*Return Field Name (Code)* – Der Name des Feldes mit dem vom Anwender eingegebenen Wert aus der Auswahltabelle. In der Regel entspricht dieser Feldname dem Namen, der in *txtField.ControlSource* angegeben ist. Hier ist nur der Feldname ohne den Tabellennamen anzugeben. Dieses Feld muss vom Typ „Zeichen“ sein. Gegebenenfalls ist der Name mit TRANSFORM() in einen Zeichentyp umzuwandeln.

*Return Field Name (Description)* – Der Name des Feldes mit der Beschreibung, die aus der Auswahltabelle zurückgegeben wird. Es kann auch ein Ausdruck zurückgegeben werden. Der Wert wird im Beschreibungsfeld angezeigt.

*Return Field Name (Internal Key)* – Der Name des Feldes aus der Auswahltabelle, das den Primärschlüssel enthält. Über dieses Feld wird die Beziehung von der Bearbeitungstabelle zur Auswahltabelle in der Datenumgebung hergestellt.

*Control Source Internal Key* – Der Name des Feldes aus der Bearbeitungstabelle, das den Primärschlüssel enthält. Dieses Feld enthält den Fremdschlüssel aus der Auswahltabelle.

## Die Klasse cDownload

Diese Klasse ermöglicht das Herunterladen von Dateien aus dem Internet. Bei Bedarf können die heruntergeladenen Dateien ausgeführt werden und es können weitere Aktionen ausgeführt werden. Insbesondere ist hierdurch die Installation von Programmen aus dem Internet möglich.

Durch die Verwendung von Makros und die Execmacro-Funktion von VFP kann diese Klasse sehr vielseitig eingesetzt werden.

Makros sind Zeichenketten, die eine Folge von Befehlen aus der Makrosprache enthalten. Eigene Makros können erstellt werden. Ein Beispiel ist in der Tabelle Vfxsys.dbf im Feld *Install\_GS* zu finden. Mit diesem Makro wird das Programm Ghostscript aus dem Internet heruntergeladen und installiert.

Diese Klasse verwendet die in der Eigenschaft *goProgram.cConnectionCheckURL* gespeicherte Internetseite um zu überprüfen, ob eine Internetverbindung besteht. Bei Bedarf wird eine Verbindung automatisch hergestellt. Wenn im DFÜ-Netzwerk keine Verbindung eingetragen ist, wird ein neuer Eintrag hinzugefügt. Die Verbindungsinformationen kann der Entwickler in den Eigenschaften vorgeben. Der Anwender kann die Telefonnummer, den Benutzernamen und das Kennwort für die neue Verbindung bei Bedarf in einem Dialog ändern.

### Eigenschaften

*LastErrorNo* – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

*LastErrorTest* – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

### Methoden

*ExecMacro* (*vcMacro*, *lnNoRun*)

*vcMacro* – Skript der Makrosprache, das ausgeführt werden soll.

*lnNoRun* – Wenn diese Eigenschaft auf .T. gesetzt wird, wird die heruntergeladene Datei nicht ausgeführt.

### Befehle der Makrosprache

„D:“ *URL*

Unter dieser Internetadresse ist die herunterzuladende Datei zu finden. Dieser Befehl führt die Datei nach dem erfolgreichen Herunterladen aus, wenn die Eigenschaft *lnNoRun* auf .F. gesetzt ist.

„C:“ *nTimeout*; *lPartial*; *lTopLevelForm*; *lResultOnError*; *SearchedString*

Wartet bis das Fenster mit dem Titel *SearchedString* erscheint.

*nTimeout* – Timeout in Sekunden. Wenn das erwartete Formular nicht innerhalb dieser Zeitspanne erscheint, wird ein Timeout-Fehler erzeugt.

*lPartial* – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, reicht es, wenn der übergebene Titel einem Teil des Fensternamens entspricht. Wenn diese Eigenschaft auf .F. gesetzt ist, muss der übergebene Titel exakt dem Namen des Fensters entsprechen.

*lTopLevelForm* – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, wird der Fensternamen nur in Top-Level-Fenstern gesucht.

*lResultOnError* – Mit dieser Eigenschaft wird das Verhalten des Skripts gesteuert, falls das Fenster nicht innerhalb der vorgegebenen Zeitspanne gefunden wurde. Wenn das Fenster für die weitere Ausführung des Skripts zwingend erforderlich ist, muss nach Ablauf der vorgegebenen Zeitspanne die Ausführung des Skripts abgebrochen werden. In diesem Fall muss der Wert von *lResultOnError* auf .F. gesetzt werden. Wenn die Ausführung des Skripts unabhängig vom Vorhandensein des Fensters nach der vorgegebenen Zeitspanne fortgesetzt werden soll, muss *lResultOnError* auf .T. gesetzt werden.

*SearchedString* – Bezeichnung, die in einem Fensternamen gesucht wird.

„W:“ *nTimeout; lPartial; lTopLevelForm; lResultOnError; SearchedString*

Es wird gewartet bis das Fenster, das die angegebene Zeichenkette im Titel enthält, geschlossen ist.

*nTimeout* – Timeout in Sekunden. Wenn das erwartete Fenster innerhalb dieser Zeitspanne nicht geschlossen ist, wird ein Timeout-Fehler ausgelöst.

*lPartial* – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, reicht es, wenn der übergebene Titel einem Teil des Fensternamens entspricht. Wenn diese Eigenschaft auf .F. gesetzt ist, muss der übergebene Titel exakt dem Namen des Fensters entsprechen.

*lTopLevelForm* – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, wird der Fensternamen nur in Top-Level-Fenstern gesucht.

*lResultOnError* – Mit dieser Eigenschaft wird das Verhalten des Skripts gesteuert, falls das Fenster nicht innerhalb der vorgegebenen Zeitspanne gefunden wurde. Wenn das Fenster für die weitere Ausführung des Skripts zwingend erforderlich ist, muss nach Ablauf der vorgegebenen Zeitspanne die Ausführung des Skripts abgebrochen werden. In diesem Fall muss der Wert von *lResultOnError* auf .F. gesetzt werden. Wenn die Ausführung des Skripts unabhängig vom Vorhandensein des Fensters nach der vorgegebenen Zeitspanne fortgesetzt werden soll, muss *lResultOnError* auf .T. gesetzt werden.

*SearchedString* – Eine Zeichenkette nach der im Titel eines Fensters gesucht wird.

„X:“

Schließt das Top-Level-Fenster. Mit dem „C:“-Befehl muss zuvor sichergestellt werden, dass das gewünschte Fenster sichtbar ist.

„K:“ *nKeyCode1; nKeyCode2; ...*

Die aufgeführten Tastenschlüssel werden in den Windows-Tastaturpuffer übertragen.

„U:“ *URL*

Von dieser Internetadresse wird das Herunterladen ausgeführt. Die heruntergeladene Datei wird unabhängig vom Wert der Eigenschaft *lnNoRun* nicht ausgeführt.

## **Beispiel**

Beschreibung der Installation von Ghostscript:

D: <ftp://mirror.cs.wisc.edu/pub/mirrors/ghost/AFPL/gs811/gs811w32.exe>

Lädt die Datei gs811w32.exe aus dem Internet herunter und führt sie anschließend aus.

*C: 30; .F.; .F.; .F.; WinZip Self-Extractor - gs811w32.exe*

Wartet bis das Fenster mit dem Titel „WinZip Self-Extractor - gs811w32.exe“ erscheint.

*K: 43*

Sendet den Tastenschlüssel „Eingabetaste“ an das aktive Fenster. Dadurch wird das Entpacken der Dateien ausgelöst.

*C: 60; .F.; .F.; .F.; AFPL Ghostscript Setup*

Wartet bis das Fenster mit dem Titel „AFPL Ghostscript Setup“ erscheint.

*K: 43*

Sendet den Tastenschlüssel „Eingabetaste“ an das aktive Fenster. Dadurch wird die Installation von Ghostscript gestartet.

*W: 240; .F.; .F.; .F.; AFPL Ghostscript Setup Log*

Wartet solange das Fenster „AFPL Ghostscript Setup Log“ geöffnet ist. Dieses Fenster zeigt den Fortschritt der Installation an und die Skriptausführung muss warten, bis dieser Vorgang beendet ist.

*C: 30; .T.; .T.; .T.; Ghostscript*

Wartet bis das Fenster mit dem Titel „Ghostscript“ erscheint. Dieses Fenster zeigt die Nachricht an, dass die Installation erfolgreich war.

*X:*

Schließt das letzte Fenster.

Hiermit ist die Installation von Ghostscript beendet.

## Die Klasse cCreatePDF

Diese Klasse erstellt Berichtsausgaben im PDF-Format. Als Parameter werden der Aliasname des zu verwendenden Cursors, der Name der zu erstellenden PDF-Datei, der Name der Berichtsdatei sowie eine optionale For-Klausel übergeben.

Um eine PDF-Datei erstellen zu können, müssen Ghostscript und ein Postscript-Druckertreiber auf dem jeweiligen Computer installiert sein. Diese Klasse prüft, ob Ghostscript bereits installiert ist. Sollte dies nicht der Fall sein, wird Ghostscript automatisch aus dem Internet heruntergeladen und installiert. Für das Herunterladen aus dem Internet wird die Klasse cDownload verwendet. In dem Memofeld *Install\_gs* aus der Tabelle *Vfxsys.dbf* befindet sich das Skript, das zum Herunterladen und zur Installation von Ghostscript verwendet wird. In der Beschreibung der Klasse cDownload befinden sich weitere Hinweise.

Wenn kein Postscript-Druckertreiber installiert ist, installiert diese Klasse automatisch den Druckertreiber, dessen Name in der Eigenschaft *goProgram.PSPrinterToInstall* hinterlegt ist. In der Regel sind hierfür keine Benutzereingaben erforderlich.

Der Bericht wird über den Postscript-Druckertreiber ausgegeben und in einer Datei gespeichert. Das Programm Ghostscript wandelt diese Postscript-Datei in eine PDF-Datei um.

### Eigenschaften

*LastErrorNo* – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

*LastErrorText* – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

### Methoden

*AddAttachment(tcAlias, tcFileName, tcReport, tcFor)*

Fügt dem CreatePDF-Objekt Informationen über eine Datei hinzu. Es wird automatisch eine PDF-Datei zum dem als Parameter übergebenen Bericht erstellt. Ein weiterer Ausdruck kann als Parameter angegeben werden. Dieser Ausdruck dient dazu die Daten des Berichts zu filtern.

*Create\_PDF(tcAlias, tcRezFile, tcFRXName, tcFor)*

*tcAlias* – Aliasname, der für die Berichtsausgabe verwendet wird.

*tcRezFile* – Vollständiger Pfadname der zu erstellenden PDF-Datei.

*tcFRXName* – Name der Berichtsdatei, die zur Erstellung der PDF-Datei verwendet wird.

*tcFor* – For-Klausel zur Filterung der zu exportierenden Daten.

Diese Methode gibt den Wert *.T.* zurück, wenn die PDF-Datei erfolgreich erstellt werden konnte. *.F.* wird zurückgegeben, wenn die PDF-Datei nicht erstellt werden konnte. In diesem Fall sind die Nummer und die Beschreibung des aufgetretenen Fehlers in den Eigenschaften *LastErrorNo* und *LastErrorText* gespeichert.

## Die Klasse cEmail

Diese Klasse gibt dem Entwickler die Möglichkeit E-Mails zu versenden. Es müssen nur wenige Parameter der Methode *Send\_Email\_Report* übergeben werden um eine Berichtsausgabe im PDF-Format als E-Mail-Anhang versenden zu können.

### Eigenschaften

*LastErrorNo* – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

*LastErrorTest* – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

*oEmail\_Attachment* – Diese Eigenschaft wird nur intern verwendet. Sie enthält eine Collection der Anhänge.

### Methoden

*AddAttachment* (*tsAlias*, *tcFileName*, *tcReport*, *tcFor*)

Fügt dem E-Mail-Objekt Informationen über einen E-Mail-Anhang hinzu, der mit der nächsten E-Mail gesendet wird. Die Informationen über alle vorzubereitenden PDF-Anhänge werden in der Eigenschaft *oEmail\_Attachment* gespeichert. Wenn der Aliasname einer geöffneten Tabelle oder Ansicht angegeben und der Name einer Berichtsdatei übergeben wird, wird diese Klasse automatisch eine PDF-Datei zu dem Bericht erstellen. Es kann ein weiterer Ausdruck als Parameter angegeben werden, der dazu verwendet wird die Daten des Berichts zu filtern. Wenn kein Aliasname angegeben wird und keine Tabelle im aktuellen Arbeitsbereich geöffnet ist, nimmt die Klasse an, dass ein Dateianhang vorbereitet wurde. In diesem Fall muss die Datei existieren, wenn die Methode *Send\_Email\_Report* aufgerufen wird.

*tcAlias* – Aliasname, der für die Berichtsausgabe und für den PDF-Export verwendet wird.

*tcRezFile* – Name des Dateianhangs (wenn eine PDF-Datei erstellt wird, wird dies der Name der PDF-Datei).

*tcFRXName* – Name der Berichtsdatei, aus der die PDF-Datei erstellt wird.

*tcFor* – For-Klausel mit der die Berichtsdaten für die PDF-Ausgabe gefiltert werden.

*Send\_Email\_Report* (*tcEmail*, *tcSubject*, *tcText*)

Sendet eine E-Mail. Wenn die E-Mail mit Anhängen versendet werden soll, müssen diese vorher mit der Methode *AddAttachment* angefügt werden.

*tcEmail* – Adresse des E-Mail-Empfängers.

*tcSubject* – Betreff der E-Mail.

*tcText* – Text der E-Mail.

*ClearAttachment*

Löscht alle E-Mail-Anhänge.

Die Methode *AddAttachment* kann entsprechend der Anzahl der benötigten Anhänge beliebig oft aufgerufen werden. Es werden die Aliasnamen der Tabellen oder Ansichten, die Namen der zu erstellenden Dateien, die Namen der Berichtsdateien und eventuell zu verwendende For-Klauseln als Parameter übergeben. Dann wird die Methode *Send\_Email\_Reports* aufgerufen. Alle PDF-Dateien werden erstellt und als E-Mail-Anhänge versendet. Auch die Dateien, die zuvor vorbereitet wurden und als Anhang versendet werden sollen, werden an die E-Mail angehängt.

## Die Klasse cArchive

Diese Klasse dient der Datensicherung und Datenwiederherstellung. Die Daten werden in Zip-Archiven gesichert. Der Name des Archivs wird aus dem Namen des Datenordners und dem aktuellen Datum in ANSI-Form zusammengesetzt. Wenn zum Beispiel der Datenordner „Data“ heißt und die Datensicherung am 4. November 2003 durchgeführt wird, heißt das Archiv Data20031104.zip.

### Eigenschaften

*OverrideFile* – Mit dieser Eigenschaft wird festgelegt was passiert, wenn eine Datei mit dem gleichen Namen schon vorhanden ist.

0 – Vorgang abbrechen, wenn bereits eine Datei mit dem gleichen Namen existiert.

1 – Wenn eine Datensicherung durchgeführt wird, werden neue Dateien dem Archiv hinzugefügt und bestehende Dateien werden aktualisiert. Wenn eine Wiederherstellung durchgeführt wird, werden existierende Dateien nicht überschrieben.

2 – Wenn eine Datensicherung durchgeführt wird, wird ein bestehendes Archiv überschrieben. Wenn eine Wiederherstellung durchgeführt wird, werden existierende Dateien überschrieben.

*OperationSuccessfully* – Enthält das Ergebnis der letzten Aktion.

.T. – wenn die Aktion erfolgreich ausgeführt werden konnte.

.F. – wenn die Aktion nicht ausgeführt werden konnte.

### Methoden

*CreateArchive (lcFileLocation, lcMask, lcArchFilePathName)*

*lcFileLocation* – Vollständiger Pfad zu dem Ordner, dessen Inhalt gesichert werden soll.

*lcMask* – Zu sichernde Dateien, Beispiel: „\*.DBF;\*.FPT;\*.CDX“.

*lcArchFilePathName* – Vollständiger Pfadname der zu erstellenden Archivdatei.

Rückgabewert: .T. – wenn die Aktion erfolgreich ausgeführt werden konnte, .F. – wenn die Aktion nicht ausgeführt werden konnte.

*ZipProgress (tcCurrentOperatedFile, nState, nAllFilesSize, nZIPedFilesSize, nArchiveCurrentSize)*

Callback-Funktion der *CreateZipArchive*-Funktion (in VFX.fll).

*tcCurrentOperatedFile* – Der Name der Datei, die dem Archiv hinzugefügt wird.

*nState* – Aktuelle Aktion:

1 – Datei existiert

2 – Datei wird dem Archiv hinzugefügt

3 – Datei erfolgreich dem Archiv hinzugefügt

4 – Datei konnte dem Archiv nicht hinzugefügt werden

5 – Archivierungsvorgang erfolgreich beendet

6 – Archivierungsvorgang nicht erfolgreich beendet

7 – Keine Dateien zu archivieren

*nAllFilesSize* – Die Größe aller zu archivierenden Dateien.

*nZIPedFilesSize* – Die Größe der dem Archiv bereits hinzugefügten Dateien.

*nArchiveCurrentSize* – Die aktuelle Größe des Archivs.

Rückgabewert: 0 – Abbruch der Aktion. 1 – Fortsetzen Dateien dem Archiv hinzuzufügen und existierende Dateien zu überschreiben. 2 – Bestehende Archivdatei überschreiben

*ExtractFromArchive(lcArchFileForExtract, lcPathForExtract)*

*lcArchFileForExtract* – Vollständiger Pfadname der zu entpackenden Zip-Datei.

*lcPathForExtract* – Zielordner, in den die Dateien entpackt werden sollen.

*UnZipProgress(tcCurrentOperatedFile, nState, nArchiveFilesSize, nUnZIPedFilesSize)*  
Callback-Funktion der ExtractZipArchive-Funktion (in VFX.fll).

*tcCurrentOperatedFile* – Name der aktuell entpackten Datei aus dem Archiv.

*nState* – Aktuelle Aktion

- 1 – Datei existiert bereits
- 2 – Datei wird entpackt
- 3 – Datei entpacken beendet
- 4 – Datei konnte nicht entpackt werden
- 5 – Entpacken des Archiv erfolgreich abgeschlossen
- 6 – Entpacken des Archiv nicht erfolgreich abgeschlossen

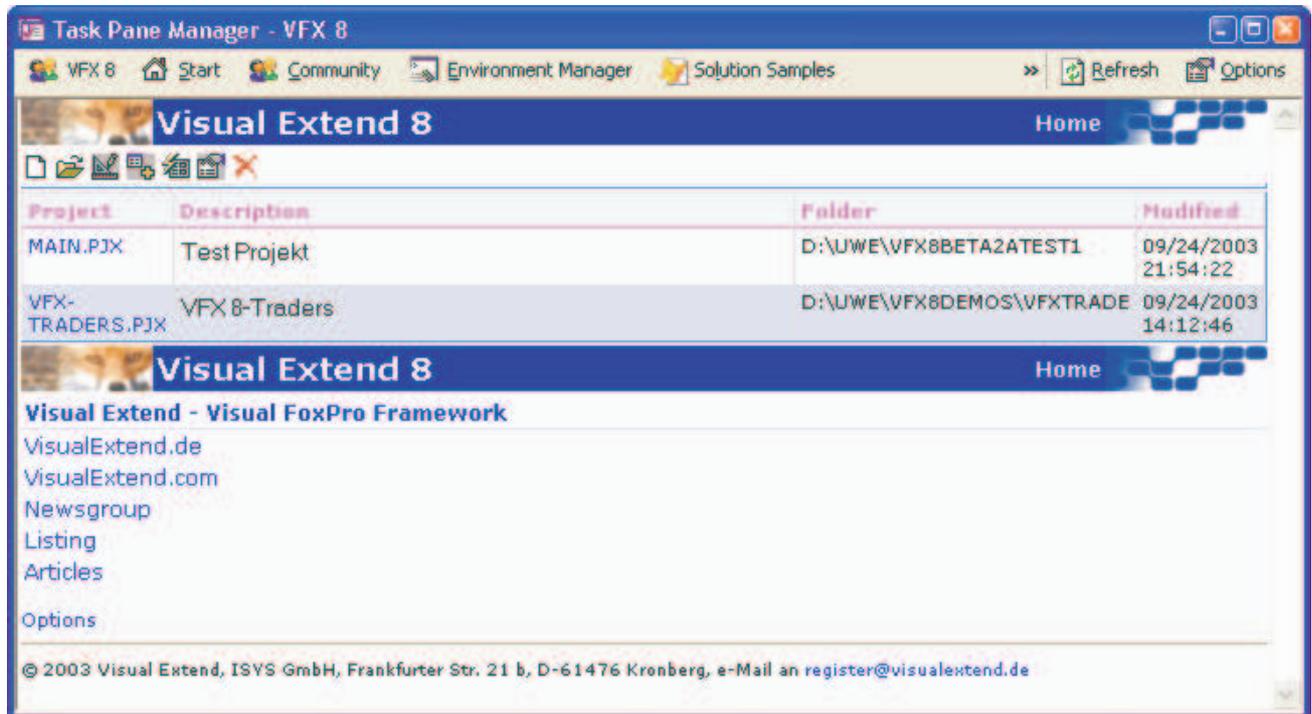
*nArchiveFilesSize* – Größe des Archivs

*nUnZIPedFilesSize* – Größe des Teils des Archivs, das bereits entpackt wurde.

Rückgabewert: 0 – Abbruch der Aktion. 1 – Aktuelle Datei nicht entpacken. 2 – Vorhandene Datei überschreiben.

## VFX 8.0 Task Pane

Der VFX – Application Manager wurde in die VFP Task Pane integriert.



Über die Symbolleiste stehen folgende Funktion zur Verfügung:

- New Project* Startet den VFX – Application Wizard.
- Open Project* Öffnet ein VFP-Projekt und stellt den aktuellen Pfad auf den Projektordner.
- Modify Project* Öffnet das in der VFX 8.0 Task Pane selektierte Projekt und stellt den aktuellen Pfad auf den Projektordner.
- Add Project* Fügt ein vorhandenes VFP-Projekt der VFX 8.0 Task Pane hinzu.
- Rebuild* Neu kompilieren aller Dateien des in der VFX 8.0 Task Pane selektierten Projekts. Das Projekt wird nach dem kompilieren zur Bearbeitung geöffnet.
- Properties* Start der VFX – Project Properties zum in der VFX 8.0 Task Pane selektierten Projekt.
- Delete* Entfernt das selektierte Projekt aus der VFX 8.0 Task Pane.

## Öffnen-Dialog im Windows-XP-Stil

Zusätzlich zu dem in bisherigen VFX-Versionen vorhandenem Öffnen-Dialog (*Vfxfopen.scx*) steht in VFX 8.0 ein neuer Öffnen-Dialog im Windows-XP-Stil (*Vfxxpopen.scx*) zur Verfügung. Dieser neue Öffnen-Dialog ist standardmäßig aktiviert. Mit der Eigenschaft *goprogram.lxpopenstyle* kann auf Wunsch auf den alten Öffnen-Dialog umgeschaltet werden.



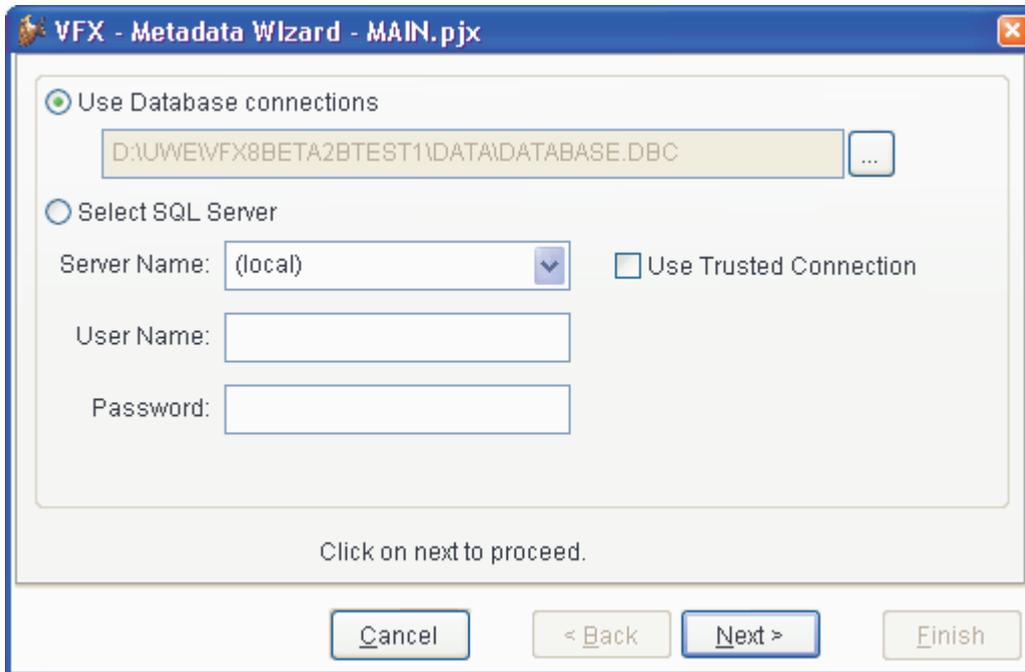
*lxpopenstyle*

- .T. – der neue Öffnen-Dialog im Windows-XP-Stil wird verwendet.
- .F. – der alte Öffnen-Dialog (*Vfxfopen.scx*) wird verwendet.

Die Gruppenüberschriften im neuen Öffnen-Dialog werden aus dem neuen Tabellenfeld *Vfxopen.groupcap* gelesen.

## Aktualisierung einer SQL Server Kundendatenbank

Dieser Assistent hilft Ihnen Metadaten aus Ihrer aktuell benutzten SQL Server-Datenbank zu erstellen. Diese Metadaten können zur Aktualisierung der Datenbank beim Kunden verwendet werden.



Wahlweise können die Verbindungen aus einer VFP-Datenbank ausgelesen werden um die Verbindung zu einem SQL Server herzustellen oder der SQL Server kann manuell ausgewählt werden.



Über den Menüpunkt „Metadata Wizard“ aus dem VFX 8.0-Menü kann die Tabelle „Datadict.dbf“ erstellt werden. Dies ist eine freie Tabelle, in der die Struktur der Datenbank inklusiv Constraints, benutzerdefinierten Datentypen, Regeln, Ansichten und gespeicherten Prozeduren gespeichert wird.

Der Assistent durchsucht das aktive Projekt nach Verbindungen und analysiert die Strukturen der Datenbanken. Wenn die Tabelle „Datadict.dbf“ an die Kunden weitergegeben wird, wird die Struktur der dortigen Datenbank aktualisiert. Dabei werden wieder die bestehenden Verbindungen zum Zugriff auf die Datenbank verwendet.

## Applikationsschutz durch Produktaktivierung

Die Produktaktivierung hat das Ziel die unerlaubte Verwendung der Anwendung auf nicht aktivierten Computern zu verhindern.

Der Applikationsschutz durch Produktaktivierung kann im VFX – Application Wizard auf der Seite 3. *Options* durch aktivieren des Kontrollkästchens „*Enable product activation*“ eingeschaltet werden.

Später kann diese Einstellung in *Vfxmain.prg* geändert werden. Die Eigenschaft *goProgramm.UseActivation* muss auf *.T.* gesetzt werden, um die Produktaktivierung einzuschalten. Wenn die Eigenschaft *goProgramm.UseActivation* auf *.F.* gesetzt ist, ist die Applikation nicht durch die Produktaktivierung geschützt.

Wenn der Applikationsschutz durch Produktaktivierung aktiviert ist, wird beim Start der Applikation das Objekt *goProgram.SecurityRights* instanziiert. Dieses Objekt hat Eigenschaften mit den Namen der Benutzerrechte, die der Entwickler definiert hat. Jede dieser Eigenschaften kann einen von drei Werten annehmen:

-1 – Die Applikation ist nicht aktiviert. In diesem Fall kann der Entwickler entscheiden welche Aktion ausgeführt werden soll. Der Anwender könnte zum Beispiel begrenzten Zugriff auf Funktionen haben, solange die Applikation nicht aktiviert ist.

0 – Die Applikation ist aktiviert, aber der Anwender hat nicht das Recht diese Aktion auszuführen.

1 – Die Applikation ist aktiviert und der Anwender hat das Recht die Aktion auszuführen.

Wenn der Applikationsschutz durch Produktaktivierung aktiviert ist, werden der Aktivierungsschlüssel und das Datum des ersten Starts der Anwendung in einer INI-Datei gespeichert. Der Entwickler kann den Namen dieser INI-Datei selbst wählen, sodass jede Applikation ihre eigene INI-Datei verwendet. Der Standardname ist *VFX.ini*.

Der Aktivierungsschlüssel ist mit einer Entwickler-Definierten Kombination von verschiedenen Hardware-Merkmalen verschlüsselt. Der Schutz kann durch Hinzufügen von Zeichenkonstanten, Schlüsseln aus der Windows-Registrierungsdatenbank und durch das Erstellungsdatum einer beliebigen Datei weiter verbessert werden. Diese Kombination kann für jede Applikation getrennt festgelegt werden, sodass jede Applikation ihre eigenen Aktivierungsregeln hat. Die Erstellung des Sicherheitsmusters wird später in diesem Dokument beschrieben.

Zusätzlich zu diesen Einstellungen kann der Entwickler den Typ des Schutzes festlegen.

Der Standardschutz erstellt die INI-Datei beim ersten Start der Applikation. Das während des Erstellens der INI-Datei aktuelle Systemdatum wird in der Datei gespeichert. Dieses Datum steht während der Ausführung der Anwendung in der Eigenschaft *goProgram.InstallationDate* zur Verfügung und kann dazu verwendet werden die Laufzeit der Applikation zu beschränken.

Der Nachteil dieses Schutzes ist, dass der Anwender die erstellte INI-Datei löschen kann und die INI-Datei beim nächsten Start der Applikation mit einem neuen Datum erneut erstellt wird. Um eine solche Manipulation durch den Anwender auszuschließen, kann der Entwickler einen erweiterten Schutz einstellen. Hierbei wird eine zusätzliche Datei verwendet, die mit der Applikation vertrieben werden muss. Der Standardname dieser Datei heißt „*FirstInstall.txt*“. Der Dateiname kann mit der

Eigenschaft *cFirstInstall* aus der Klasse *cActivation* (appl.vcx) eingestellt werden. Die Datei „FirstInstall.txt“ wird im Windows-Ordner abgelegt.

Wenn der Entwickler den Schutz mit der Datei „FirstInstall.txt“ auswählt, wird sich die Applikation folgendermaßen verhalten.

Beim Start der Applikation wird die INI-Datei überprüft. Wenn diese Datei existiert, wird das Datum des ersten Starts der Eigenschaft *goProgram.InstallationDate* zugewiesen und die Benutzerrechte werden entsprechend dem Aktivierungsschlüssel eingestellt.

Wenn die INI-Datei nicht existiert wird angenommen, dass dies der erste Start der Applikation ist. Wenn dies der Fall ist, wird zusätzlich überprüft, ob die Datei „FirstInstall.txt“ existiert. Wenn diese Datei existiert ist sichergestellt, dass die Applikation wirklich zum ersten Mal gestartet wird. Das Systemdatum wird jetzt in der INI-Datei gespeichert und die Datei „FirstInstall.txt“ wird gelöscht. Wenn ein Anwender nun versucht eine Applikation zu reaktivieren indem er die INI-Datei löscht, wird die Ausführung der Applikation beendet, weil die Datei „FirstInstall.txt“ nicht existiert. Dieser erweiterte Schutz der Applikation bedeutet eine bessere Sicherheit. Der Entwickler darf jedoch nicht vergessen die Datei „FirstInstall.txt“ beim Vertrieb der Applikation mit auszuliefern.

Wenn der Anwender die installierte Applikation aktivieren möchte, muss er seinen Installationsschlüssel an den Entwickler senden. Der Installationsschlüssel kann auf drei verschiedene Arten an den Entwickler gesendet werden. Die gewünschte Art kann in der Eigenschaft *nRegWay* eingestellt werden:

0 – Der Installationsschlüssel wird in einer Messagebox angezeigt. Der Anwender kann den Schlüssel kopieren und in einer anderen Applikation (zum Beispiel in einer E-Mail) einfügen.

1 – Der Installationsschlüssel wird in einer Datei gespeichert. Diese Datei kann später an den Entwickler gesendet werden. Der Dateiname wird in der Eigenschaft *cParamFile* hinterlegt.

2 – Der Installationsschlüssel wird in einer Datei gespeichert und sofort als E-Mail-Anhang an den Entwickler geschickt. Der Dateiname muss in der Eigenschaft *cParamFile* hinterlegt werden. Die E-Mail-Adresse des Entwicklers muss in der Eigenschaft *cRegEMail* eingetragen werden.

Die Länge der Installationsschlüssel hängt von dem gewählten Sicherheitsmuster ab (siehe *Defining Security key pattern*). Der Anwender könnte den Installationsschlüssel per E-Mail an den Entwickler senden oder auf der Registrierungs-Webseite eintragen. Der Entwickler trägt den Installationsschlüssel im Define User Rights-Assistenten ein um einen Aktivierungsschlüssel für den Anwender zu erstellen. Der generierte Aktivierungsschlüssel wird dann an den Anwender geschickt und vom Anwender im Aktivierungsformular eingegeben um die Applikation zu aktivieren.

## **Liste der verwendeten Begriffe**

*Systemspezifischer Wert* – Ein systemspezifischer Wert, zum Beispiel die Seriennummer einer Hardware-Komponente oder das Erstellungsdatum einer bestimmten Datei oder ein Schlüssel aus der Windows-Registrierungsdatenbank. Die zu verwendete Datei und der zu verwendende Schlüssel aus der Windows-Registrierungsdatenbank können vom Entwickler festgelegt werden.

*Aktivierungsschlüssel* – Dies ist eine Zeichenkette, die die Berechtigungen für einen speziellen Anwender enthält. Der Aktivierungsschlüssel wird vom Entwickler anhand des Installationsschlüssels erstellt. Der Aktivierungsschlüssel ist für andere Anwender nutzlos.

*Installationsschlüssel* – Dies ist eine Zeichenkette, die Informationen über die beim Anwender eingesetzte Hardware enthält. Der Installationsschlüssel wird vom Entwickler benötigt um einen Aktivierungsschlüssel erstellen zu können.

*Sicherheitsmuster* – Für jede Applikation kann ein eindeutiger Sicherheitsschlüssel angelegt werden. Dieser Schlüssel setzt sich aus einer Reihe systemspezifischen Werten zusammen und identifiziert einen PC eindeutig. Bei der Erstellung des Sicherheitsschlüssels können Textverarbeitungsfunktionen verwendet werden.

*Installationsdatum* – An diesem Datum wurde eine Applikation installiert oder erstmalig auf einem PC gestartet.

## **Eigenschaften der Klasse *cVFXActivation***

*cFirstInstall* – Diese Eigenschaft enthält den Namen einer Datei. Anhand des Vorhandenseins dieser Datei entscheidet diese Klasse, ob die Applikation erstmalig gestartet wird. Wenn dieser Eigenschaft eine leere Zeichenkette zugewiesen wird, kann nicht überprüft werden, ob die Applikation erstmalig gestartet wird. Das Datum des Starts wird dann ohne weitere Überprüfung in der INI-Datei eingetragen.

*cINIFileName* – Der Name der INI-Datei, in der die Aktivierungsinformationen und das Datum des ersten Applikationsstarts gespeichert sind. Der Standardwert ist „VFX.INI“.

*cParamFile* – Der Name der Datei, in der der Installationsschlüssel gespeichert wird. Abhängig vom Wert der Eigenschaft *nRegWay* kann diese Datei per E-Mail versendet oder auf einem anderen Weg verarbeitet werden.

*cRegMail* – In dieser Eigenschaft wird die E-Mail-Adresse des Entwicklers gespeichert, an die die Datei mit dem Installationsschlüssel gesendet wird, wenn die Eigenschaft *nRegWay* den Wert 2 hat.

*cRegFileName* – Hier kann der Name einer Datei angegeben werden, die bei der Installation erstellt wird. Das Erstellungsdatum dieser Datei wird verwendet um das Installationsdatum zu ermitteln. Wenn dieser Eigenschaft kein Wert zugewiesen wird, wird das Systemdatum des ersten Starts der Anwendung verwendet.

*nRegWay* – In dieser Eigenschaft kann eingestellt werden, wie der Entwickler den Installationsschlüssel bekommen soll.

0 – Der Installationsschlüssel wird in einer Messagebox angezeigt und der Anwender kann den Schlüssel kopieren und in beliebige Applikationen einfügen.

1 – Der Installationsschlüssel wird in einer Datei gespeichert. Der Anwender kann diese Datei später an den Entwickler übermitteln. Der Name der Datei wird in der Eigenschaft *cParamFile* hinterlegt.

2 – Der Installationsschlüssel wird in einer Datei gespeichert und an den Entwickler als E-Mail-Anhang gesendet. Der Name der Datei wird in der Eigenschaft *cParamFile* hinterlegt. Die E-Mail-Adresse des Entwicklers, an die der Installationsschlüssel gesendet wird, wird in der Eigenschaft *cRegEMail* eingetragen.

## **Das Funktionsprinzip**

Die Aktivierungsinformationen werden auf dem PC des Kunden in einer INI-Datei gespeichert. Der Name dieser INI-Datei wird in der Eigenschaft *cINIFileName* der Klasse *cVFXAcvtivation* (Appl.vcx) eingetragen. Der Standardwert ist *VFX.ini*.

Der Entwickler kann wählen, ob die einfache Produktaktivierung verwendet werden soll oder ob zusätzlich die Datei „FirstInstall.txt“ benutzt werden soll, um den ersten Start der Applikation zu protokollieren. Der Name dieser Datei kann in der Eigenschaft *cFirstInstall* der Klasse *cVFXAcvtivation* (Appl.vcx) eingetragen werden. Der Standardwert ist *FirstInstall.ini*.

Wenn die Datei „FirstInstall.txt“ verwendet werden soll, muss diese Datei mit der Applikation vertrieben werden. Das Installationsprogramm muss diese Datei im Windows-Ordner speichern. Das Aktivierungsobjekt wird diese Datei beim ersten Start der Applikation löschen. In diesem Moment wird das Installationsdatum in der INI-Datei gespeichert. Später wird bei jedem Start der Applikation in der INI-Datei geprüft, ob das Installationsdatum vorhanden ist. Wenn das Datum fehlt und wenn die Datei „FirstInstall.txt“ nicht vorhanden ist, wird davon ausgegangen, dass an der Installation manipuliert wurde und die Ausführung der Applikation wird beendet.

Wenn die Datei „FirstInstall.txt“ nicht verwendet wird, wird die INI-Datei neu erstellt, falls sie nicht vorhanden ist.

Das Installationsdatum kann auf zwei Arten ermittelt werden. Entweder wird das Systemdatum verwendet oder es wird das Erstellungsdatum einer bestimmten Datei verwendet. Wenn das Erstellungsdatum einer Datei verwendet werden soll, muss der Name dieser Datei in der Eigenschaft *cRegFileName* der Klasse *cVFXActivation* gespeichert werden.

## **Die Definition der Aktivierungsregeln**

*Defining Security key pattern* – Wenn der Application Security Key and Rights-Assistent das erste Mal für ein Projekt gestartet wird, muss ein neues Muster für dieses Projekt angelegt werden.

Nach der Eingabe eines Namens für das Muster wird der Application Security Key and Rights-Assistent gestartet.

Auf der Seite *Security Key* des Assistenten befindet sich eine Combobox aus der ein Muster für das aktuelle Projekt ausgewählt werden kann. In dem darunterliegenden Grid können so viele Zeilen hinzugefügt werden, wie benötigt werden. Jede Zeile definiert einen Teil des Installationsschlüssels. Der Installationsschlüssel stellt sicher, dass die Applikation nur auf dem Computer ausgeführt wird, für den der Aktivierungsschlüssel erstellt wurde.

In der ersten Spalte des Grid kann ein systemspezifischer Wert ausgewählt werden. In einer Combobox sind hier alle möglichen Hardware-Parameter aufgeführt, die zur Erstellung des Installationsschlüssels verwendet werden können. Zusätzlich können Zeichenkettenfunktionen angewendet werden, um den Wert zu verändern.

Zum Beispiel sollen anstelle der vollständigen Seriennummer einer Festplatte nur die letzten vier Stellen zur Erstellung des Installationsschlüssels verwendet werden. Aus der Combobox in der ersten Spalte wird „HDD Factory Serial Number“ ausgewählt. Die Systemvariable, die diesem Parameter entspricht „HDDFactoryNumber“, erscheint daraufhin in der zweiten Spalte. Um nur die letzten vier Stellen zu verwenden, muss der folgende Ausdruck in dieser Spalte eingetragen werden: `RIGHT(ALLTRIM(HDDFactoryNumber),4)`.

Wenn einer der systemspezifischen Werte Dateierstellungsdatum oder Windows-Registrierungsschlüssel verwendet werden soll, müssen weitere Parameter angegeben werden. Wenn das Erstellungsdatum einer Datei verwendet werden soll, muss der Name der Datei angegeben werden. Um einen Windows-Registrierungsschlüssel verwenden zu können, muss die Bezeichnung des Schlüssels eingegeben werden.

Alle Hardware-Parameter, bzw. die Ausdrücke, die davon verwendet werden sollen, werden zu einem Installationsschlüssel zusammengesetzt. Dieser Schlüssel wird benutzt, wenn der Anwender seine Applikation aktivieren will. Wenn nur ein Hardware-Parameter verändert wird, wird die Installation ungültig und der Anwender muss einen neuen Aktivierungsschlüssel anfordern, entsprechend seiner geänderten Hardware.

Es können so viele Zeilen dem Grid hinzugefügt werden, wie benötigt werden. Die Zeilen im Grid können mit den Pfeiltasten am rechten Rand des Assistenten in eine andere Reihenfolge gebracht werden.

**Achtung:** Je mehr Zeilen dem Grid hinzugefügt werden, desto länger werden die Aktivierungsschlüssel!

Nach der Definition der Aktivierungsregeln wird das Muster in der Eigenschaft *cActPattern* der Klasse *cVFXActivation* (Appl.vcx) gespeichert. Zur Laufzeit der Applikation wird der Installationsschlüssel verwendet. **DER INSTALLATIONSSCHLÜSEL DARF NIEMALS GELÖSCHT WERDEN!**

Auf der Seite *Rights* können bis zu 32 verschiedene Benutzerrechte angelegt werden. Damit kann der Zugriff auf bis zu 32 Module einer Applikation gesteuert werden. Beispielsweise können Rechte angelegt werden, die es dem Anwender erlauben Formulare zu starten (*RunDataForms*), Berichte zu drucken (*RunReports*), Daten zu bearbeiten (*EditData*), Daten anzusehen (*ViewData*) usw. Zur Laufzeit der Applikation können die einzelnen Berechtigungen überprüft werden und ggf. wird die entsprechende Aktion ausgeführt.

Alle Benutzerrechte stehen zur Laufzeit als Eigenschaften des global sichtbaren Objekts *goProgram.SecurityRights* zur Verfügung, sodass an jeder Stelle der Applikation darauf zugegriffen werden kann.

Wenn die Applikation nicht aktiviert ist, haben alle Benutzerrechte den Wert -1. Wenn die Applikation aktiviert ist, hat ein Benutzerrecht den Wert 1, wenn die Aktion erlaubt ist und 0, wenn die Aktion nicht erlaubt ist.

Wenn dem Anwender entsprechend dem obigen Beispiel alle Rechte zur Datenbearbeitung gegeben wurden, er aber nicht das Recht hat Berichte zu drucken, sehen die Eigenschaften zur Laufzeit so aus:

```
goProgram.SecurityRights.RunDataForms = 1
goProgram.SecurityRights.RunReports = 0
goProgram.SecurityRights.EditData = 1
goProgram.SecurityRights.ViewData = 1
```

Um im Assistenten ein Recht einzutragen muss zuerst das Kontrollkästchen in der ersten Spalte markiert werden. Dann wird ein Name für das Recht eingetragen. Zur Laufzeit der Applikation wird eine Eigenschaft des *SecurityRights*-Objekts mit diesem Namen angelegt. Daher müssen bei der Eingabe des Namens die Konventionen zur Namensgebung von VFP beachtet werden!

**Anmerkung:** Applikationsrechte sind für jede Applikation unterschiedlich. Die Rechte, die für eine andere Applikation erstellt wurden, können nicht verwendet werden. Auch wenn ähnliche Rechte benötigt werden, müssen diese neu erstellt werden. Die Applikationsrechte werden in der Tabelle Vfxapprights.dbf im Projektordner gespeichert.

### **Definition der Rechte für einen Anwender**

Wenn der Anwender seinen Installationsschlüssel sendet, muss ein Aktivierungsschlüssel erstellt werden. Dieser Aktivierungsschlüssel teilt der Applikation mit, ob der Anwender eine bestimmte Aktion ausführen darf. Für jede Aktion muss das entsprechende Recht auf 1 gesetzt werden.

Wenn aus dem VFX 8.0-Menü „Create Activation Key“ aufgerufen wird, erscheint der Dialog mit dem Benutzerrechten für das aktive Projekt.

Mit der Schaltfläche „*Read User Hardware Details*“ öffnet sich ein Dialog, in den der Installationsschlüssel des Anwenders eingegeben wird. Der Installationsschlüssel kann über die Zwischenablage eingefügt werden oder aus einer Datei gelesen werden.

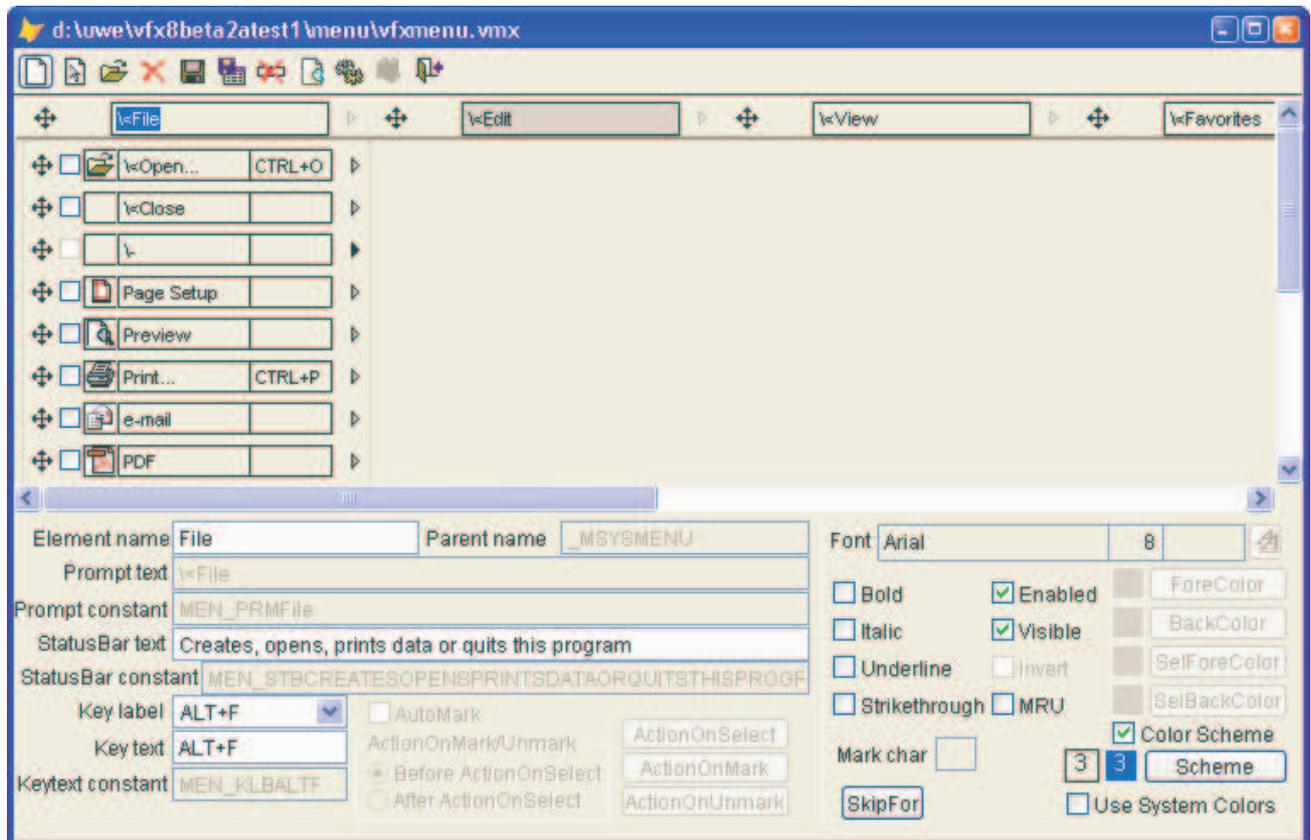
Nachdem jedes für den Anwender erlaubte Recht markiert ist, wird mit einem Klick auf OK der Aktivierungsschlüssel generiert. Der erstellte Aktivierungsschlüssel wird in der Datei <Projektname>.xak im Projektordner gespeichert. Der Aktivierungsschlüssel oder die Datei muss an den Anwender zur Aktivierung der Applikation gesendet werden.

Wenn der Anwender eine Applikation startet, die eine Aktivierung erfordert (und wenn die Applikation noch nicht aktiviert wurde), wird automatisch der Installationsschlüssel erzeugt. Abhängig vom Wert der Eigenschaft *nRegWay* wird der Installationsschlüssel entweder angezeigt oder in einer Datei gespeichert, die per E-Mail versendet werden kann. Nachdem der Anwender den Aktivierungsschlüssel erhalten hat, kann er ihn im Aktivierungsfenster eingeben. Damit ist die Applikation auf diesem Computer aktiviert.

Wenn der Anwender später den Menüpunkt Extras, Aktivierung auswählt, wird der Installationsschlüssel angezeigt, unabhängig von der Einstellung der Eigenschaft *nRegWay*.

## Der VFX Menü-Designer

Der VFX Menü-Designer (VMD) ist ein Werkzeug zur schnellen Entwicklung von Menüs. Der VMD ist ein visueller Designer, in dem das Menü schon während der Entwicklung so angezeigt wird, wie es zur Laufzeit aussehen wird. Der VMD macht die Entwicklung einfacher und ermöglicht die schnelle Einstellung aller Menü-Eigenschaften im Gegensatz vom VFP Menü-Designer, der nicht alle Eigenschaften von Menüs unterstützt. Es können mehrsprachige Menüs erstellt werden, indem auf die entsprechende Schaltfläche in der Symbolleiste geklickt wird.



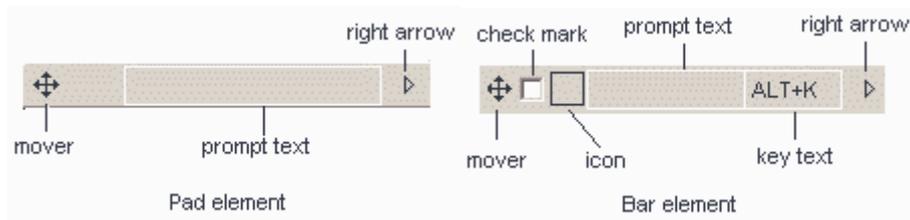
Ein in einem VFX-Projekt enthaltenes Menü kann direkt aus dem VFP-Projekt-Manager mit dem VMD geöffnet werden. Wahlweise können Menüs auch aus dem VMD heraus über das Öffnen-Symbol in der Symbolleiste oder über den entsprechenden Menüpunkt geöffnet werden. Im Öffnen-Dialog kann zwischen den Menütypen .mnx und .vmx gewechselt werden. Wenn ein Menü geöffnet wird, das noch nie mit dem VMD bearbeitet wurde, wird es automatisch in das .vmx-Format konvertiert.

Das geöffnete Menü kann visuell bearbeitet werden. Es können Einträge hinzugefügt und gelöscht werden und es können die Eigenschaften der einzelnen Einträge bearbeitet werden.

Neue Menü-Pads können durch einen Klick auf den Rechtspfeil, der sich rechts neben jedem Menüeintrag befindet, angelegt werden. Dadurch wird ein Untermenü angelegt. Einem Menü können neue Einträge hinzugefügt werden, indem auf den Pfeil nach unten unterhalb des Eintrags geklickt wird.

Ein Menüeintrag oder ein Menü-Pad können gelöscht werden, wenn sich der Fokus darauf befindet. Über den Menüpunkt löschen oder mit der Tastenkombination *Strg+Entf* wird der markierte Eintrag gelöscht.

Einige der Eigenschaften eines Menüeintrags können visuell eingestellt werden:



#### - Prompt text

Der angezeigte Text kann direkt eingetragen werden, wenn sich der Fokus auf dem jeweiligen Eintrag befindet. Die Textbox „*Prompt text*“ im unteren Teil des VMD dient nur zur Anzeige des aktiven Eintrags im visuellen Teil des Designers.

#### - Key text

Die Bezeichnung des Tastenschlüssels zeigt dem Anwender die Zugriffstaste oder Tastenkombination an, mit der der Eintrag ausgewählt werden kann. Die Bezeichnung sollte dem im unteren Teil des VMD gewählten Tastenschlüssels entsprechen.

#### - Check mark

Damit sich ein Menüeintrag wie ein Kontrollkästchen verhält, muss bei „*AutoMark*“ eine Markierung gesetzt werden. Wenn zusätzlich eine Markierung bei „*Check mark*“ gesetzt wird, ist der Menüeintrag bereits beim Laden des Menüs markiert. Für Menüeinträge, die sich wie ein Kontrollkästchen verhalten, können zusätzliche Code-Teile ausgeführt werden, wenn das entsprechende Kontrollkästchen markiert wird („*ActionOnMark*“) bzw. wenn die Markierung aufgehoben wird („*ActionOnUnmark*“). Im Standard-VFP-Editorfenster kann der jeweilige Code bearbeitet werden.

Der Code, der bei *ActionOnMark* oder *ActionOnUnmark* eingegeben wird, kann wahlweise vor oder nach der *ActionOnSelect* ausgeführt werden. Um dieses Verhalten einzustellen, ist die entsprechende Option „*Before ActionOnSelect*“ oder „*After ActionOnSelect*“ auszuwählen.

#### - Icon

Jedem Eintrag in einem Menü kann ein Symbol zugeordnet werden. Diese Symbol kann aus den in VFP integrierten Systemressourcen ausgewählt werden oder es kann eine Datei verwendet werden. Durch einen Klick auf das schwarzumrandete Kästchen kann ein Symbol mithilfe des „*Get a picture from*“-Dialogs ausgewählt werden. In diesem Dialog kann zwischen einer Datei und einem Symbol aus den VFP Systemressourcen gewählt werden. Wenn einem Menüeintrag ein Symbol zugeordnet ist und sich dieser Menüeintrag wie ein Kontrollkästchen verhalten soll, dient das Symbol als Markierung. Wenn der Eintrag markiert wird, erscheint das Symbol eingedrückt. Wenn die Markierung aufgehoben wird, erscheint das Symbol normal.

Die Position der einzelnen Einträge innerhalb des Menüs kann per Drag & Drop verändert werden. Für diesen Vorgang ist der Vierwegepfeil , der sich links neben allen Einträgen befindet festzuhalten. In einigen Fällen sind Verschiebeoperationen nicht möglich. Ein Menü-Pad kann nicht in einen Menüeintrag umgewandelt werden und umgekehrt. Außerdem ist es nicht möglich einen Menüeintrag in ein Untermenü zu verschieben.

Weitere Eigenschaften der Menüeinträge können im unteren Teil des Designers eingestellt werden. Dazu gehören der Zeichensatz, die Vordergrund- und Hintergrundfarbe, eine Meldung, die in der VFP-Statusbar angezeigt wird sowie der Name einer Konstanten, die verwendet wird, wenn ein mehrsprachiges Menü erstellt wird. Alle Änderungen werden unmittelbar im aktiven Element angezeigt.

Mit der Schaltfläche „*ActionOnSelect*“ kann in einem Editor-Fenster die auszuführende Aktion eingegeben werden. Über die Schaltfläche „*SkipFor*“ kann eine Bedingung eingegeben werden. Wenn diese Bedingung .T. liefert, kann der dazugehörige Menüeintrag nicht ausgewählt werden.

Die eingestellten Eigenschaften beziehen sich immer auf den aktiven Menüeintrag. Neue Menüeinträge erben die Eigenschaften des zuvor ausgewählten Eintrags.

Der Zeichensatz kann über die Schaltfläche „*Font*“ ausgewählt werden. Der Standard-Windows-Dialog zur Auswahl eines Zeichensatzes erscheint. In diesem Dialog können insbesondere die Schriftart und die Schriftgröße sowie der Schriftschnitt ausgewählt werden.

Zu jeder Zeit kann eine Vorschau des Menüs angezeigt werden, indem in der Symbolleiste oder im VMD-Menü „*Preview*“ gewählt wird.